Designing smart ITS services through innovative data analysis modeling

Thesis document

PhD Student: Jamie Arjona Martínez
Supervised by: Dr. Josep Casanovas and Dr. MªPaz Linares

Thesis Presented to Obtain the Qualification of Doctor from the Universitat Politècnica de Catalunya

December 18, 2020

Universitat Politècnica de Catalunya
Department of Statistics and Operations Research

Acknowledgements

A lot has happened during the work realized during this thesis and I'm very grateful to all the people that helped me in one way or another because they have changed my vision about the world... BUT, of all the people that has been involved, there are ones that deserve a special distinction.

I want to sincerely thank the inLab FIB, the Department of Statistics and Operational Research of the UPC and the company Worldsensing S.L. for bringing to me this opportunity and the trust placed in me.

To the Department of Civil Engineering at the National University of Athens, specially to Professors Eleni Vlahogianni and George Yannis for the opportunity and the guidance. Also I want to thank the PhD students of the department: Elena, Foteini, Maria, Dimitris, Vasilis, Dimitrios, Panagiotis, Emmanouli, etc. because you made my stay in Greece wonderful. I miss you all a lot!

To Matt Elmore and the external referees, I very much appreciate the time and effort you have invested in reviewing my work.

A mi director Josep Casanovas por los consejos, su visión y mano izquierda en las muchas reuniones con la empresa, equilibrando la balanza de la dedicación de la tesis entre el mundo académico y la empresa.

Quiero agradecer a mis compañeros de aventuras del inLab por esos ratos tan divertidos que hemos pasado juntos: Carla, Vanessa, Samuel, Didac, Marc, Ton, Ester, Pol, Pau, Josefran, Marta y un largo etcétera. Pero concretamente a Germán, Juan y Juanjo ya que siempre que les he necesitado han estado ahí para lo que fuera, especialmente en los malos momentos.

A Pablo y a Alba por ser esa lucecita que siempre brilla incluso en los momentos más difíciles y oscuros. Muchas gracias por estar siempre ahí.

A mi mejor amigo Cristian, por todo su apoyo no solamente durante la realización de esta tesis si no por la infinitud de momentos juntos. Recuerdo con especial cariño cuando me viniste a visitar a Grecia, sabes que eres un hermano para mí.

A mis padres y a mi hermano por ser la base de mi persona y ser el apoyo para conseguir mis objetivos.

A M^{a}Paz Linares por todo. Doblaría el número de página de la tesis si tuviera que escribir toda su implicación, dedicación y apoyo (tanto personal como profesional) durante los

años de realización de esta tesis (incluso antes de empezar!). Sin ella nada de esto hubiese sido posible. Por más difíciles que sean las cosas siempre encontraremos el camino para ser felices.

Jamie Arjona Martínez

Barcelona, December 2020

Abstract

Nowadays, one of the most important problems in urban areas concerns traffic congestion. This, in turn, has an impact on the economy, nature, human health, city architecture, and many other facets of life. Part of the vehicular traffic in cities is caused by parking space availability. The drivers of private vehicles usually want to leave their vehicles as close as possible to their destination. However, the parking slots are limited and may not be enough to sustain the demand, especially when the destination pertains to an attractive area. Thus, individuals looking for a place to park their vehicles contribute to increasing traffic flow density on roads where the parking demand cannot be satisfied.

An Internet of Things (IoT) approach allows us to know the state of the parking system (availability of the parking slots) in real time through wireless networks of sensor devices. An intelligent treatment of this data could generate forecasted information that may be useful in improving management of on-street parking, thus having a notable effect on urban traffic. Smart parking systems first appeared in 2015, with IoT platforms in Santander, San Francisco and Melbourne. That is the year when those cities began to provide on-street real-time parking data in order to offer new services to their citizens. One of the most interesting services that these kinds of platforms can offer is parking availability forecasting, for which the first works in this field studied the temporal and spatial correlations of parking occupancy to support short-term forecasts (no more than 30 minutes). Those short-term forecasts are not useful at all to the end user of this service; thus, the necessary prediction intervals should be at the order of magnitude of hours.

In this context, this thesis focuses on using parking and other sources of data to characterize and model different parking systems. The methodology used employs novel techniques for providing real-time forecasts of parking availability based on data from sensors with certain inaccuracies due to their mechanical nature. The models are developed from four different methodologies: ARIMA, multilayer perceptron (MLP), long-short term memory (LSTM) and gated recurrent unit (GRU). The first has been the standard approach to forecasting in the ITS literature, while the latter ones have proven to be the best neural network (NN) architectures for solving a wide set of sequential data problems, such as those presented in this work. As far as we know, LSTM and GRU methods (recurrent neural network approaches) have been used recently with good results in traffic forecasting, but not for parking. In addition, we propose using exogenous data such as weather conditions and calendar effects, thereby converting the problem from univariate to multivariate. It is shown here how NN methods naturally handle the

increased complexity in the problem. The reason for using exogenous variables is that they can offer relevant information that cannot be inferred from the sensor measurements.

The proposed methods have been intensively compared by creating parking models for parking sectors in five cities around the world. The results have been analysed in order to identify and provide exhaustive guidelines and insights into the inner mechanisms of parking systems while also ascertaining how the idiosyncrasies of each method are reflected in the model forecasts.

When comparing the results according to their disciplines of origin (ARIMA from statistics and NN methods from machine learning), neither of the proposed methodologies is clearly better than the other, as both can provide forecasts with low error but by different means. ARIMA has shown lower error rates in small-sized sectors where the more recent status of the parking system is more relevant; while the NN methods are more capable of providing forecasts for large-sized sectors where patterns are dependent on long time horizons.

Keywords: parking availability forecast; deep learning; smart cities, recurrent models, time series.

Resumen

En la actualidad uno de los mayores problemas de las zonas urbanas tiene origen en la congestión del tráfico con un alto impacto en la economía, el medio ambiente, la salud, la arquitectura de la ciudad y otras muchas facetas de la vida urbana. En muchas ocasiones parte de la congestión del tráficotiene origen en la disponibilidad de las plazas de aparcamiento debido a que los conductores de vehículos privados suelen querer aparcar sus vehículos lo más cerca posible de su destino pero las plazas de aparcamiento son limitadas y pueden no ser suficientes para mantener la demanda, especialmente cuando el destino corresponde a un área atractiva. Por lo tanto, las personas que buscan un lugar para aparcar sus vehículos contribuyen a aumentar la densidad del flujo de tráfico en las vías donde no se puede satisfacer la demanda de estacionamiento.

Un enfoque basado en el *Internet of Things (IoT)* nos permite en tiempo real conocer la disponibilidad de plazas de estacionamiento a través de redes inalámbricas de sensores. Un tratamiento inteligente sobre estos datos puede generar información que ayude a predecir la futura demanda de estacionamiento en las zonas sensorizadas mejorando así la gestión del estacionamiento en la calle y teniendo un efecto notable en el tráfico urbano. Sistemas de aparcamiento inteligentes aparecieron por primera vez en el año 2005, con plataformas de IoT en Santander, San Francisco y Melbourne proporcionando datos de estacionamiento en la vía pública en tiempo real y ofreciendo nuevos servicios a sus ciudadanos basados en éstos. Uno de los servicios ofertados más interesante es el de la predicción de la disponibilidad de estacionamiento en el cuallos primeros trabajos académicos se centraron en estudiar las correlaciones temporales y espaciales de la ocupación del estacionamiento para proveer pronósticos a corto plazo (predicciones a tiempo máximo de 30 minutos) y que en muchas ocasiones no son de utilidad ya que para el usuario final es preferible tener estimaciones de la disponibilidad de estacionamiento en el order de magnitud de horas.

Con este contexto, esta tesis se centra en el uso de datos de aparcamientos y otras fuentes de datos para caracterizar y modelizar diferentes sistemas de aparcamiento. La metodología utilizada emplea técnicas innovadoras para proporcionar predicciones en tiempo real sobre la disponibilidad de aparcamiento basadas en datos de sensores que debido a su propia naturaleza presentan ciertas inexactitudes de captación. Los modelos se desarrollan a partir de cuatro metodologías diferentes: *Autoregressive Integrated Moving Averag*e (ARIMA), *Multilayer Perceptron* (MLP), *Long-Short Term Memory* (LSTM) y *Gated Recurrent Unit* (GRU). La primera ha sido el enfoque estándar de predicción en la literatura sobre Sistemas de Transporte Inteligentes, mientras que las otras tres han demostrado ser las mejores arquitecturas de redes neuronales para re-

solver un amplio conjunto de problemas de datos de naturaleza secuencial, como los que se tratan en este trabajo. Hasta donde sabemos, los métodos LSTM y GRU (enfoques de redes neuronales recurrentes) se han utilizado recientemente para la predicción de tráfico, obteniendo buenos resultados, pero no para aparcamiento. Además, proponemos utilizar datos exógenos como las condiciones meteorológicas y los efectos del calendario, transformando el problema de univariante a multivariante y demostramos como los métodos de redes neuronales gestionan de forma natural esta mayor complejidad del problema. El motivo para incluir variables exógenas es el de reducir la incertidumbre dada por las mediciones de los sensores ya que el uso de los sistemas de aparcamiento está condicionado por procesos no medibles por los sensores.

Los métodos propuestos se han comparado intensamente mediante la creación de modelos de estacionamiento para sectores de aparcamiento en cinco ciudades del mundo. Los resultados se han analizado con el fin de identificar y proporcionar pautas exhaustivas y conocimientos sobre los mecanismos internos de los sistemas de estacionamiento y, al mismo tiempo, determinar cómo se reflejan las idiosincrasias de cada método y de cada sector en los pronósticos del modelo.

Al comparar los resultados según sus disciplinas de origen (ARIMA de estadística y métodos de redes neuronales de aprendizaje automático), ninguna de las metodologías propuestas es claramente mejor que las otras, ya que ambas pueden proporcionar predicciones con bajo error. ARIMA ha demostrado tener tasas de error más bajas en sectores de aparcamiento de menor dimensión donde el estado más reciente del sistema de aparcamiento es más relevante; mientras que los métodos de redes neuronales has demostrado ser mejores capaces de proporcionar mejores predicciones para sectores de gran tamaño donde los patrones tienen mayores dependencias temporales.

Palabras clave: predicción de disponibilidad de aparcamiento; aprendizaje profundo; *smart cities*; modelos recurrentes; series temporales.

Contents

List of Figures

List of Tables

1 Introduction

This chapter describes the context for this thesis , the background of the problem to be solved, the objectives and an outline of how this document is organized.

1.1 Thesis context

The work was carried out under the program Doctorats Industrials de Catalunya, a research program that aims to cultivate relationships between research centers and industry. This thesis is a collaborative project between the Department of Statistics and Operations Research at the Universitat Politècnica de Catalunya (DEIO-UPC) in cooperation with the inLab FIB innovation and research laboratory in the Barcelona School of Informatics and with the startup company Worldsensing S.L. The goal of this project is to provide innovative solutions to problems generated by parking in urban areas.

1.1.1 DEIO-UPC

This UPC department works in statistics and operations research. Among its many areas of focus, it collaborates with teachers in: the School of Industrial Engineering of Barcelona; the School of Industrial, Aerospace and Audiovisual Engineering of Terrassa; the Barcelona School of Informatics; and the UPC School of Mathematics and Statistics. The department is largely involved in research activities in the field of industrial engineering, intelligent interfaces, automatic data processing, optimization and simulation.

1.1.2 Worldsensing S.L.

This company manufactures their own sensor devices, networks and software in order to accomplish their mission: provide the tools for making the right operational decisions based on real-time intelligence. As a company, Worldsensing has achieved recognition as a global Internet of Things pioneer, with customers from more than 50 countries across five continents.

They have recently placed their focus on providing services that help cities become smarter, specifically by assisting city operators in their decision-making processes using

data provided by the city. With that data, they apply statistics and machine learning techniques to generate the most useful visualizations for taking decisions.

The company considers research and innovation to be crucial in achieving their main goals. For this reason, they have incorporated PhD and doctoral students into their Department of Innovation under the leadership of Dr Andrea Bartoli, PhD (UPC), who is also the industry advisor for this doctoral thesis. In addition, Worldsensing collaborates intensely with the academic world, in particular with universities such as the Universitat Oberta de Catalunya and the Università di Padova.

1.2 Background

Urban areas today face an increasing number of problems with one common origin: traffic congestion due to road transport. Urban traffic affects the quality of life in cities and metropolitan areas around the world, with consequences for widely different scopes: the economy, environment, public health, and city landscapes, among others.

According to the yearly statistical book, *Energy, transport and environment indicators,* published by the statistical office of the European Union, the most used mode of passenger transportation in 2016 was the car, which accounted for nearly 83% of use compared to other modes (trains, motor coaches, buses and trolleybuses) (see Figure 1.1). Companies like INRIX and TomTom have made online dashboards available for sharing information about traffic in capital cities. According to a Forbes article (McCarthy [2019]) using data from the TomTom traffic index, *"Over the past decade, traffic congestion has continued to increase globally with nearly 75% of the cities in the index reporting increased or stable congestion levels between 2017 and 2018".* Figure 1.2 shows the ranking of the world's capital cities that have lost the most time due to traffic congestion.

Given these facts, it is imperative to develop new strategies and policies for sustainable urban transport, thus making it equally essential to invest not only in the improvement of infrastructures, but in their planning and management as well.

1.2.1 The parking problem

Parking in dense urban areas can have a positive feedback effect on traffic congestion, in other words, by increasing it. Drivers of private vehicles usually want to leave their transportation mode as close as possible to their destination. However, parking slots are limited and may not be sufficient for sustaining demand, especially when the destination pertains to an attractive area. This forces individuals to continue driving in search of a parking space within their destination's neighborhood. These vehicles contribute to increasing traffic density on roads where parking demand cannot be satisfied. In major areas, like the city of Barcelona, 17% of the traffic density results from vehicles looking for a parking slot (El Periodico [2012]).

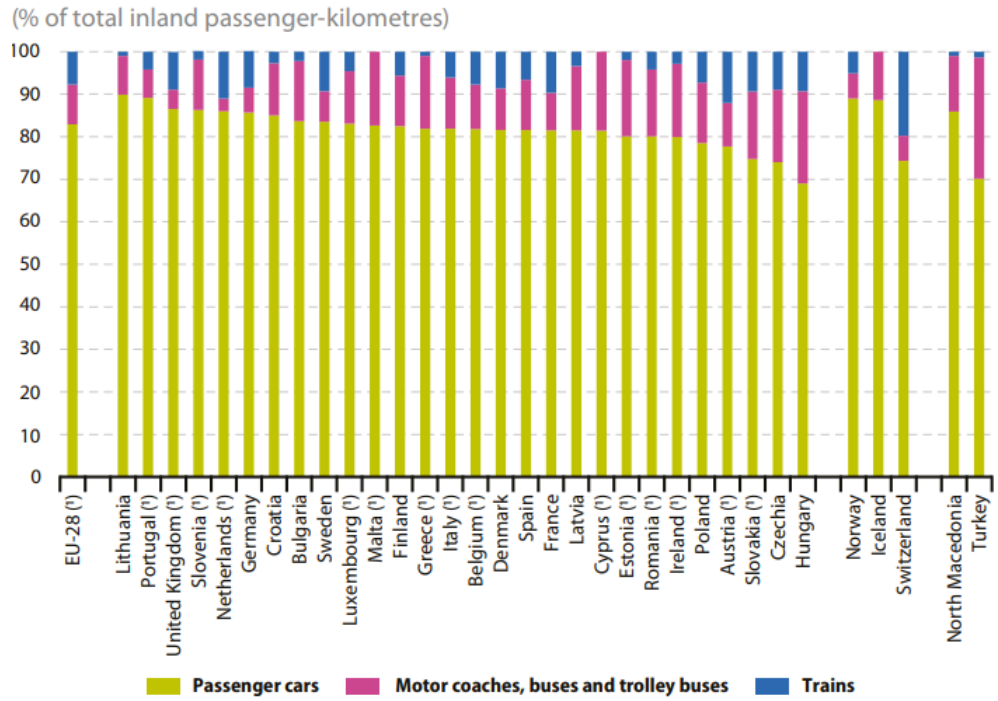


Figure 1.1: Transportation modes of EU countries: Percentage of total inland transport by passenger-kilometer. Source: Eurostat [2018].

In recent years, parking has become a strategic issue for economic interests and has attracted a lot of attention from researchers because it has been identified as one of the main contributors to urban traffic congestion. The increase in urban traffic congestion has an impact on the environment, health and the economy. Donald Shoup, a pioneer in studying the economic impact of parking in urban areas, has demonstrated through several studies that part of the traffic congestion in cities comes from cruising traffic: vehicles that continue to drive in search of a place to park after arriving at their destination (Shoup [1997], Shoup [2006] and Shoup [2018]).

The authors of a recent work (Hampshire and Shoup [2018]) present a summary table of cruising studies in different cities around the world from 1927 to 2015, in which they show the percentage of traffic caused by cruising and/or the average time spent searching for a parking slot. This table can be seen in Figure 1.3. Another work conducted in the city of Bristol found that cruising traffic caused 790 metric tons of CO_2 in one year at a total cost of US$467,000 in wasted fuel (Jones et al. [2018]). In 2006, a study in France (Gantelet and Lefauconnier [2008]) estimated that 70 million hours are spent every year searching for parking, with an imputed cost of 700M€. One IBM survey stated that drivers spend an average of 20 minutes searching for a parking spot in the world's major capital cities (IBM [2011]).

Cruising for parking affects not only the city by increasing traffic and resulting in its derivative effects (congestion, accidents, climate impact, etc.), it also poses a major problem for urban planning. City planning must strike a balance between on-street and

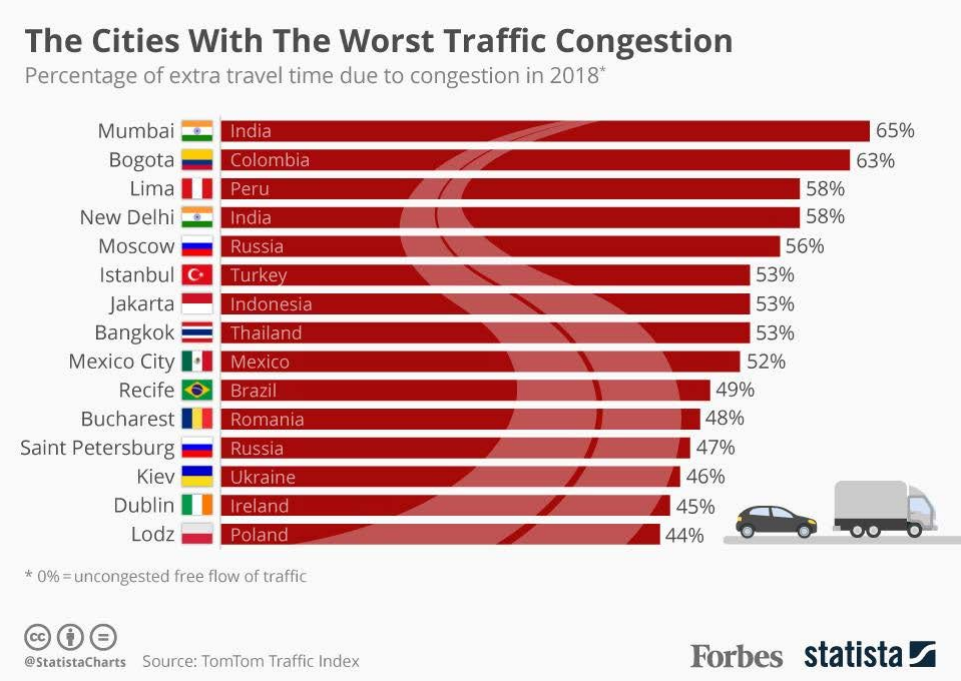


Figure 1.2: Ranking of cities in time lost to congestion. Information extracted from TomTom Traffic Index. Source: Forbes / statista.

off-street parking. When a small number of on-street parking places are available, the most common solution is to create off-street places: parking facilities like garages and lots. The increase in off-street places causes more people to take private vehicle to work instead of using public transportation, leading once again to an increase in urban traffic flow. Some authors (Shoup [1999]) propose pricing policies for on-street parking instead of creating off-street facilities, but this needs to be measured meticulously because an area with higher prices will penalize shops and leisure areas, as people will park in cheaper zones. On the other hand, if the prices are too low, vehicles will cruise for parking until a spot becomes available.

In the end, all problems related to parking have a common trigger: when the last free place is taken in a block. Figure 1.4 represents the cascade of effects when the last parking spot is occupied. From top to bottom: at the moment that the last free on-street parking slot on a block is occupied, vehicles arriving with a desire to park in that area begin cruising in search of a free slot while competing with other vehicles that have the same objective. This causes an increase in traffic flow for that block and its surrounding neighborhoods, resulting in wasted time for the vehicle users. During rush hours, the problems are exacerbated as the increase in traffic flow creates congestion in the streets as the vehicles cruise for parking together with vehicles that do not. The congestion imposes a stop-and-go flow of traffic due to competing vehicles being operated by distracted drivers looking for parking. All of this causes an increase in CO_2emissions, fuel consumption and the possibility of accidents.

Cruising for Parking

Year	City	Share of traffic cruising (%)	Average search time (minutes)
1927	Detroit (1)	19	
1927	Detroit (2)	34	
1933	Washington		8.0
1960	New Haven	17	
1965	London (1)		6.1
1965	London (2)		3.5
1965	London (3)		3.6
1977	Freiburg	74	6.0
1984	Jerusalem		9.0
1985	Cambridge	30	11.5
1993	Cape Town		12.2
1993	New York (1)	8	7.9
1993	New York (2)		10.2
1993	New York (3)		13.9
1997	San Francisco		6.5
2001	Sydney		6.5
2005	Los Angeles	68	3.3
2006	New York	28	
2007	New York	45	
2008	New York		3.8
2011	Barcelona	18	
2015	Brisbane		15.4
Average		**34**	**8.0**

Source: Schaller (2006), Transportation Alternatives (2007, 2008), Shoup (2011), Lee *et al.* (2017), and Ruh (2017).

Figure 1.3: Cruising for parking summary table. Source: Hampshire and Shoup [2018].

Different solutions have been proposed for improving parking management. The so-called smart parking systems are becoming one of the most interesting areas of research within the context of the smart cities ecosystem. Some proposed solutions are: creating specific parking zones, depending on the type of vehicle; new on-street parking payment policies; and promoting park & ride areas.

One of the issues surrounding the topic of smart parkingis managing parking slot availability. An Internet of Things approach allows knowing the state of the parking system (availability of parking slots) in real time through wireless networks of sensor devices. Intelligent treatment of this data can generate useful information for improving the management of on-street parking blocks, thus having a notable effect on urban traffic.

1.2.2 Company motivation

In 2016, the European Commission's *Horizon 2020* (H2020) program granted Worldsensing funding for their project *Fastprk2: Enhanced on-street parking management system.* This project is an initiative for developing next-generation parking detection systems equipped with Intelligent Transport Services (ITS) for cities and citizens, all merged into a single mobility software platform. This new generation of sensors will combine both magnetic and infrared detection technologies while providing highly accurate data based on their measurements, thus ultimately making it possible to offer parking predic-

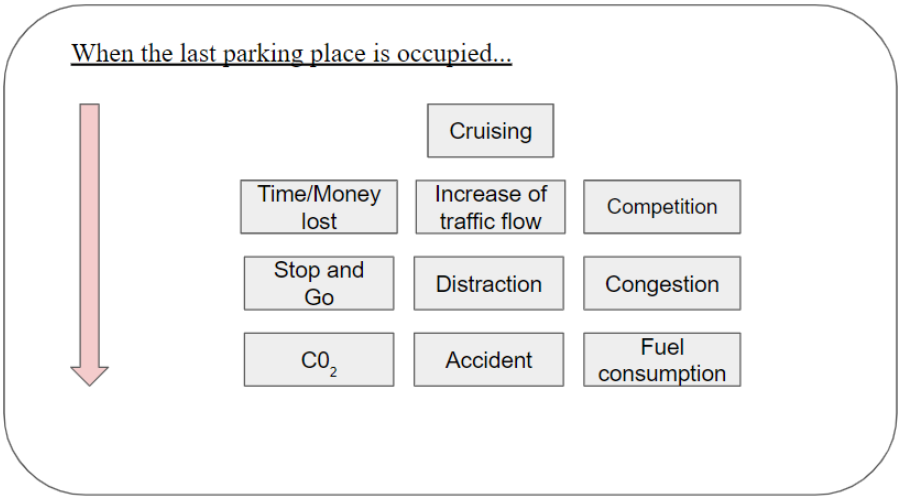


Figure 1.4: The consequences and possibilities of the last on-street space occupied in a block.

tion services. In order to provide acceptably accurate predictions, these solutions must be based on state-of-the-art computational intelligence.

As part of the project, the technology developed were deployed in different cities: Antwerp (Belgium), Barcelona (Spain), Paris (France), Wattens (Austria) and Grudziadz (Poland). Two additional cities out of this project's scope have been included in the present work: Los Angeles (USA) and Riyadh (Saudi Arabia). The motivation for this is that studying parking behaviour in other cities of the world can provide new information about parking processes.

1.3 Motivations and thesis objectives

The main objective of this thesis is to use parking and other sources of data to characterize and model different parking systems in order to provide accurate short and long term on street parking occupancy forecasts. The methodology employs novel techniques for providing real time forecasts of parking availability based on data from sensors that is accuired in real time and can have certain inaccuracies due to the sensors mechanical nature. The created models need to fulfill the following requirements: highly accurate forecasts, predictions in real time and robustness to sensor noise.

In order to correctly execute, validate and evaluate the developed forecasting models, this thesis develops an integrated software framework, which includes:

- Handling (extraction, transformation, and storage) of data from real sensor devices.
- A core component containing the developed models.
- The appropriate architecture for conducting online training in the use of the developed forecasting models and for storing the obtained output data.

- A proper validation methodology for testing the accuracy of the global procedure and developed models.
- An environment for evaluation that allows treating the obtained results and quantitatively comparing different forecasting model proposals in order to select the most suitable one for each scenario.

This objective is motivated mainly by the need to obtain accurate short/long-term forecasts of parking availability in order to improve parking management. Consequently, this will reduce the urban traffic caused by vehicles searching for free parking slots. In this context, "long-term forecast" refers to a prediction of more than one hour. It would also be very interesting to predict parking availability at a minimum of six hours while maintaining the same levels of accuracy over different time horizons.

At the moment of beginning this thesis, the literature on ITS techniques for forecasting traffic and parking had started to consider deep learning models as good candidates for forecasting. We also consider these to be of great interest to this area of research and thus explore further in this direction.

Another motivation for the research proposed in this thesis is worth mentioning. For some time now, our research group has addressed other smart city problems through traffic simulation and optimization techniques. However, approaches based on machine learning techniques had not been intensively considered. Currently, it is our belief that the data explosion from increasing Information and Communication Technology (ICT) penetration can provide a suitable framework for exploiting the potential of these forecasting methods.

Finally, as the availability of on-street parking data is scarce, this work opens up the possibilities for exploring those systems while making them available to other researchers.

Aside from the primary objective, this thesis focuses on the following research questions that, in the current literature, have not been properly solved as is explained in Chapter 4:

- Forecasting the availability of parking places is not a new problem in the literature. Over the past 20 years, efforts have been made to provide solutions, although without much progress due to the lack of data. However, because we currently have real-time parking status made available by advances in IoT and increasing computational capacity, we can train machine learning models that are capable of learning and identifying complex patterns in the data. Considering these advantages, is it now possible to provide a better solution to the problem of forecasting parking availability?
- Creating models is affordable using two different but related disciplines: classical statistical methods and machine learning. Both try to solve the same problems with different philosophies. Specifically, both provide algorithms that enable models to forecast time-dependent data. Which approach would be the most suitable in fulfilling our requirements?

- Intuition leads us to question whether sensor data is enough for inferring patterns that govern a parking system. Thus, it is natural to consider other data sources. What other sources can be exploited to improve parking forecasts and what effect do they have?
- Models are trained with historical data, but a new installation in a new system (city) lacks this fundamental prerequisite. In order to use the forecast service for new installations, the proposed algorithm must provide models with some sort of transfer learning in order to share knowledge of patterns between models in similar situations. Is this feasible?
- Which of the current NN methods is more suitable for handling data with temporal dependencies?

1.4 Thesis Outline

This thesis is organized according to the scheme presented in Figure 1.5.

Chapter 2 introduces the data pipeline from the moment the data is generated up to the point when it is stored in different data warehouses. It further details the database structures, the services provided, and the platform developed for delivering the forecasts. Chapter 3 presents an exploratory analysis of the data with the goal of finding patterns and relevant information that can help in the decision-making process. It also describes what kind of data-driven models in the literature are better suited to the problem at hand. Chapter 4 introduces the reader to the state of the art within the framework of this thesis: forecasting in the ITS area. This chapter provides an exhaustive literature review about data-driven models for forecasting traffic and parking. Chapter 5 presents the theoretical aspects of the methods considered in this work. Chapter 6 is focused in the characterization and modeling of the parking system through the ARIMA methodology. Chapter 7 modelizes the different parking systems by exploring and developing deep learning models. Previous chapters made use only of the endogenous variable, that is the usage of previous realizations of the same variable to provide its outcomes in the future, in Chapter 8 exogenous information it is introduced, namely calendar and weather, and used to create models to forecast parking availability. Finally, Chapter 9 summarizes the major contributions of this thesis and suggests proposals for advancing the present work further, other interesting lines for future research and some concluding remarks.

1.5 Methodology outline

The following methodology has been used during the development of this thesis:

1. Characterization of the studied systems (different parking sites) through a data analysis of different sources that are endogenous or exogenous to the system. This corresponds to Chapter 3 and first sections of Chapter 8.
2. Development of different experiments with the objective to answer different questions related to the modeled process and the methodology/modeling used for each system that consists in:
   a) Identification of a subset of possible models in accordance of the methodology used.
   b) Models creation (fit/training).
   c) Models evaluation through goodness of fit and visualizations on unseen data (validation/testing).
   d) Choice of the model that, in accordance to the methodology used and the evaluations, provided the best results from the subset of identified models. This model will be the output of the experiment altogether with the observations and information collected that will be used to answer the raised questions.

   The experiments can be found in chapters 4, 5 and 8.
3. Discussion about the lessons learnt through the development of the experiments, answering of the raised questions and conclusions about the parking processes, the methodologies used and the chosen models.

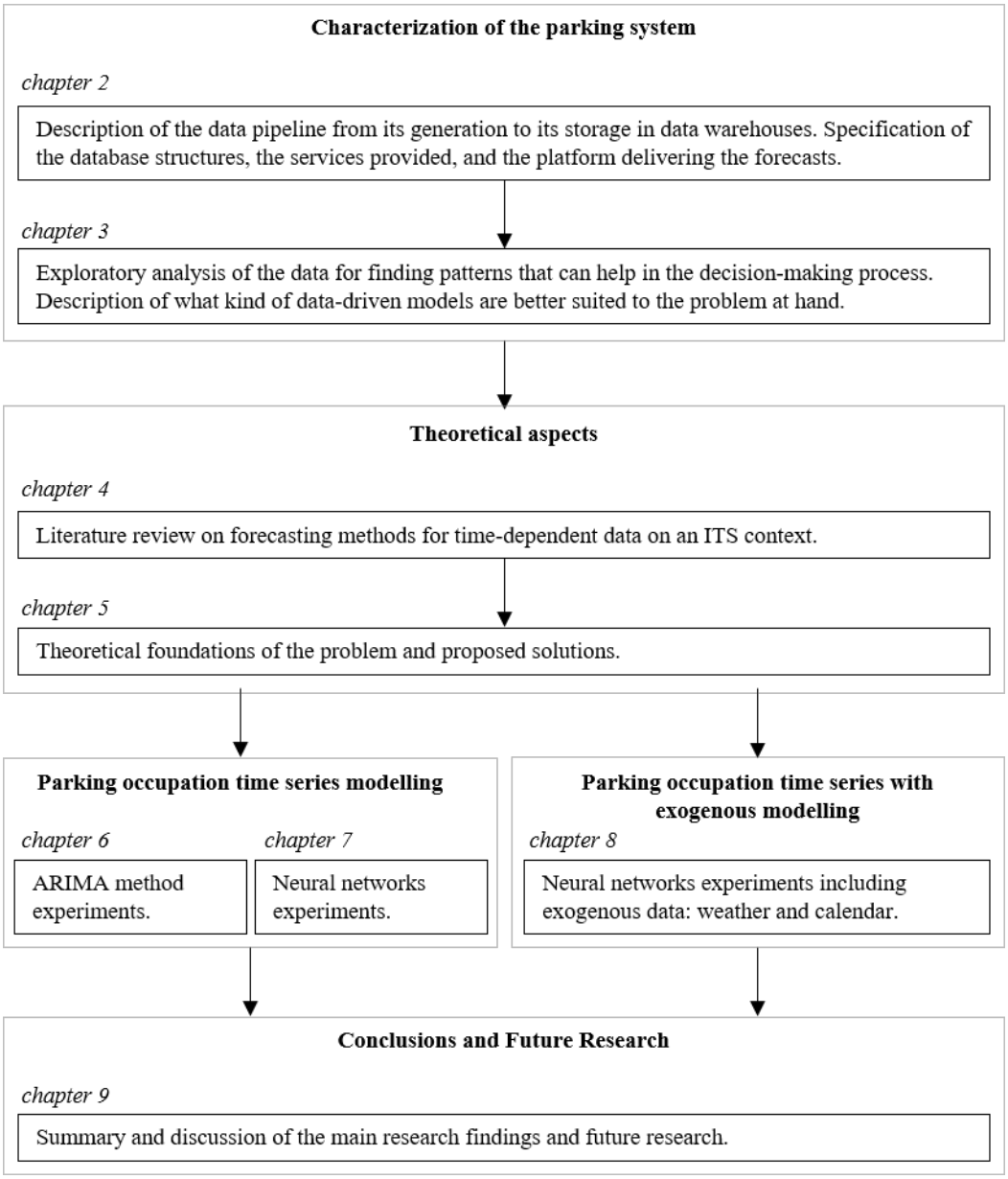


Figure 1.5: Schematic presentation of the thesis outline.

2 Problem overview based on data flow characterization

This chapter introduces the data pipeline, that is, the transfers and processes that the data undergo from their inception to storage in various data warehouses. The database structures are also described. Next, we begin with a short description of the different services provided by the data warehouses, followed by a description of the transformations needed for ascertaining the occupancy of parking spots. Finally, we present the structure that loads the models for delivering forecasts in real time. Altogether, this will serve as a reference for the following chapters.

2.1 Data pipeline

Data flow can be viewed as two sides of the same coin: one requiring physical devices, and another involving various processes and means for storage.

2.1.1 Physical components

The basic components of the system are individual nodes installed under the ground of each parking spot, namely parking sensors. Once a parking event is sensed as a change in the sensor measurements, the nodes wirelessly communicate it to its assigned gateway, which will forward the information to the cloud. The node box is composed of the following modules: sensors, micro-controller, communications module and power stage.

Sensors are composed of two different sensor technologies working in tandem: a triple-axis magnetic device and an infrared sensor. These are governed by an energy-efficient detection algorithm that outputs the parking state based on sensor measurements. This innovative approach allows using these nodes under strong magnetic noise conditions, whereas this was not possible with the previous generation of nodes (magnetic only). After installation, the sensors automatically calibrate themselves in order to establish the magnetic levels for operating.

The micro-controller runs the device drivers and firmware via a real-time operating system while guaranteeing low power consumption. This is done by a properly integrated power stage that regulates the power coming from the primary battery. The main

design objective is to achieve the lowest power consumption during sleep mode. The communications module has the objective of sending the generated information to the assigned gateway.

Gateways can be located within a radius of up to 500 m and are reached by the sensor node in a single hop. The underlying forwarding protocol is LoRa, as it can use extremely low transmit power levels and fulfill many of the requirements for long battery life. It is configured via a web tool in order to establish the sensor network. Gateways forward all data to the data warehouse servers in the cloud.

Figure 2.1 shows a representation of the data traveling through the different components.

Figure 2.1: Physical components of the data pipeline. Once a node detects a change in their measurements (a car over it), the data is sent to a gateway (the component in the traffic light) that also sends the data to the cloud. Source https://www.worldsensing.com/product/fastprk/

2.1.2 Logical components

Raw data from the gateways is stored in MongoDB databases in the cloud, where each database maintains the sensor measurements as well as other information like sensor battery, gateway status, network malfunctions, etc. All this together forms a network, that is, a sensor installation for a city. This data is used by different Extract Transform Loading (ETL) processes in order to provide different services. The schema in Figure 2.2 shows the different components that form the data pipeline from the gateways to the different services and data warehouses.

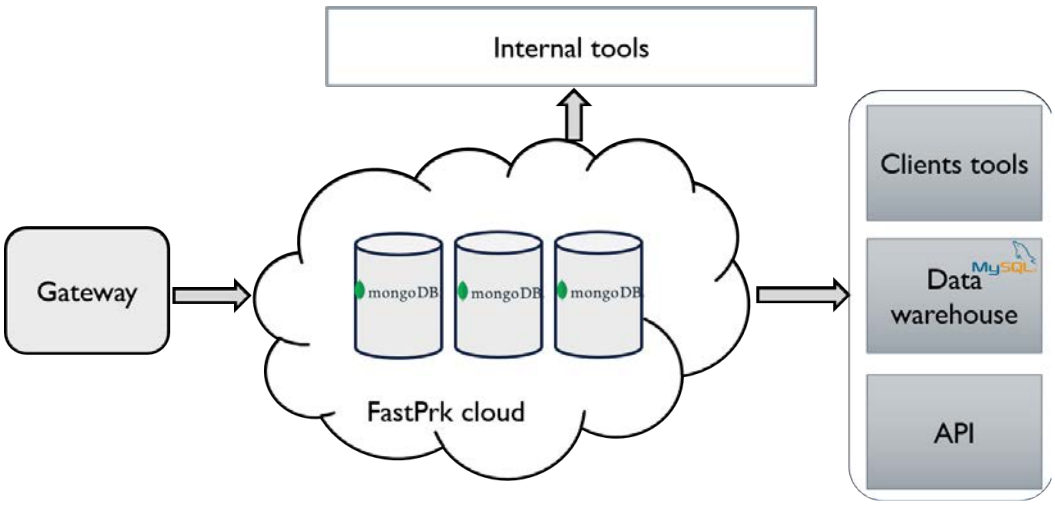


Figure 2.2: Pipeline from gateways to data warehouses and services.

One of the ETL processes extracts information from the MongoDB instances on a daily basis and transforms it for storage in relational databases (MySQL), where data is stored to provide historical aggregated and real-time information through web services. Similarly to the MongoDB, each of the databases stores information for a specific network, and all of them follow the same schema of tables. The information stored in the tables results from different data processing on the raw data and on some extra information needed for different services. Tables related to parking usage are divided according to different aggregations in the temporal and spatial dimensions. Based on spatial aggregation, there are tables with sensor information and others with sector information. In the latter, a sector serves as a logical aggregation of sensors communicating with the same gateway and which are installed in the same zone. Furthermore, sector and sensor tables are divided according to temporal aggregations, thus forming new tables based on hourly, daily and weekly aggregations. The information that can be found in these tables is:

- Sensor_id:The ID of the sensor (only in per-sensor tables)
- Sector_id: The ID of the sector
- Hour_of_day: The hour of the day, from 0 to 24.
- Day_of_year: The day of the year
- Week_of_year: The week of the year.
- Year_for_week: This complements the week_of_year column to unambiguously identify a week.
- Datetime: The date and time, representing the start of the period covered by the row (start of the hour, start of the day, start of the week)
- Total_occupancy_events: The number of occupancy events occurring over the period, i.e., how many times a car parked.

- Total_free_events: The number of free events occurring over the period, i.e., how many times a parked car left.
- Total_occupancy_hour: The number of hours that a space has been occupied during the period. For example, in the sensor per hour table, $occ \leq 1$ indicates that the space was occupied during a fraction occ of the hour (if $occ = 0.75$, the space was occupied 45 minutes). In the sensor per week table, $occ \leq 168$ indicates that the space was occupied for occ hours during the week. In the sector per hour table, the value is the sum of occupied spaces multiplied by the time they were occupied. If a sector has 3 sensors and each was occupied half an hour, $occ = 1.5$.
- Total_free_hour: The number of hours that a space has been free during the period. Complementary to the previous value "total_occupancy_hour".
- Total_full_hour (sectors only): The number of hours that the sector remains completely full. Only intervals when the sector and at least one of its sensors are active count for this metric.
- Total_full_events (sectors only): The number of times that the sector has become completely full. Only events occurring when the sector and at least one of its sensors are active count for this metric.
- Total_disconnected_hour: The number of hours that a space has been disconnected (inactive). For a sector, it is the number of disabled spaces multiplied by the time they have been so (equivalent to aggregation of free and occupancy hours).
- Avg_occupancy_length_min: The average of all the occupancy lengths in this period, in minutes
- Std_occupancy_length_min: The standard deviation of all the occupancy lengths in this period, in minutes
- Avg_free_length_min: The average of all the free lengths in this period, in minutes
- Std_free_length_min: The standard deviation of all the free lengths in this period, in minutes
- Total_occupancy_intervals : The number of occupancy intervals ended during this period.
- Occupancy_lengths_sum: The sum of occupancy intervals ended during this period.
- Free_lengths_sum: The sum of free intervals ended during this period.
- Occupancy_lengths_square_sum: The square sum of occupancy intervals ended during this period.
- Free_lengths_square_sum: The square sum of free intervals ended during this period.

- Occupancy_length_0_5_min: The number of occupancy lengths that lasted between 0 and 5 minutes
- Occupancy_length_5_10_min: The number of occupancy lengths that lasted between 5 and 10 minutes
- Occupancy_length_10_30_min: The number of occupancy lengths that lasted between 10 and 30 minutes
- Occupancy_length_30_60_min: The number of occupancy lengths that lasted between 30 and 60 minutes
- Occupancy_length_1_2_hour: The number of occupancy lengths that lasted between 1 and 2 hours
- Occupancy_length_2_4_hour: The number of occupancy lengths that lasted between 2 and 4 hours
- Occupancy_length_4_8_hour: The number of occupancy lengths that lasted between 4 and 8 hours
- Occupancy_length_8_24_hour: The number of occupancy lengths that lasted between 8 and 24 hours
- Occupancy_length_1_7_day: The number of occupancy lengths that lasted between 1 and 7 days
- Occupancy_length_weeks: The number of occupancy lengths that lasted more than a week
- Free_length_0_3_min: The number of free lengths that lasted between 0 and 3 minutes
- Free_length_3_5_min: The number of free lengths that lasted between 3 and 5 minutes
- Free_length_5_10_min: The number of free lengths that lasted between 5 and 10 minutes
- Free_length_10_30_min: The number of free lengths that lasted between 10 and 30 minutes
- Free_length_30_60_min: The number of free lengths that lasted between 30 and 60 minutes
- Free_length_1_2_hour: The number of free lengths that lasted between 1 and 2 hours
- Free_length_2_4_hour: The number of free lengths that lasted between 2 and 4 hours
- Free_length_4_8_hour: The number of free lengths that lasted between 4 and 8 hours

- Free_length_8_24_hour: The number of free lengths that lasted between 8 and 24 hours
- Free_length_days: The number of free lengths that lasted more than a day.

There are other tables that are self-explanatory, like streets, areas, day of year and hour of day. Two tables also serve as descriptors of sensors and sectors, where their ID and localization are stored.

Another ETL extracts the data as is from the MongoDB and stores it in a Data Lake (PostgreSQL), where data from other sensors is also stored.

Finally, part of the information in the MySQL and MongoDB databases is transferred by another ETL process to a time series database (InfluxDB). The information is stored in two databases, one for sensor information and the other for sector information. Each database has tables identified by a network name. For sectors, the information stored comes from the MySQL databases and is:

- Occupancy_lengths_square_sum.
- Occupancy_lengths_sum.
- Std_occupancy_length_min.
- Total_disconnected_hour.
- Total_free_events.
- Total_free_hour.
- Total_full_events.
- Total_full_hour.
- Total_occupancy_events.
- Total_occupancy_hour.
- Day.
- DayOfWeek.
- Hour.
- Month.
- Season.
- Year.
- Latitude.
- Longitude.
- Occupancy_length_0_5_min.
- Occupancy_length_10_30_min.
- Occupancy_length_1_2_hour.

- Occupancy_length_1_7_day.
- Occupancy_length_2_4_hour.
- Occupancy_length_30_60_min.
- Occupancy_length_4_8_hour.
- Occupancy_length_5_10_min.
- Occupancy_length_8_24_hour.
- Occupancy_length_weeks.
- Sector_id.
- Sector_name.
- Sectortype_id.
- Sectortype_name.
- Total_occupancy_intervals.

For the table of sensors, the information comes from the MongoDB and is:

- Amtype: Message type.
- Fpver: Sensor version.
- Id: Sensor ID.
- Lat: Latitude.
- Lon: Longitude
- Network_id: Network ID.
- Sector_id: Sector ID where the sensor belongs.
- Bat: Percentage of battery.
- Rssi: Infrared sensor state. This field is 1 if the infrared detects that the spot is occupied, 0 otherwise.
- S0x: Magnetic axis X values for magnetic sensor 0.
- S0y: Magnetic axis Y values for magnetic sensor 0.
- S0z: Magnetic axis Z values for magnetic sensor 0.
- S1x: Magnetic axis X values for magnetic sensor 1.
- S1y: Magnetic axis Y values for magnetic sensor 1.
- S1z: Magnetic axis Z values for magnetic sensor 1.
- S2x: Magnetic axis X values for magnetic sensor 2.
- S2y: Magnetic axis Y values for magnetic sensor 2.
- S2z: Magnetic axis Z values for magnetic sensor 2.

- Seqno: Sequence number of the message of the sensor.
- Signal: Signal value.
- Snr: Signal to noise ratio value.
- Temp: Temperature in celsius.

This is the data warehouse used primarily throughout this work. Its contents are updated daily.

2.2 Services

The services that use the data stored in the data warehouses are the following:

1. A REST API allows real-time access to the status of the networks, sectors and sensors. This tool was developed for a mobile app that provides users with the real-time status of parking spots.
2. Three web pages: Configuration, Monitoring and Information. As their names imply, these configure the sectors, monitor the status of network devices and indicate the occupancy status of the sensors in real time. They also provide other relevant statistics.
3. A cloud model service that provides: information about the recent historical data from the sectors; parking occupancy forecasts for upcoming hours; and a visualization tool. These services and tools were developed as one of the contributions of this thesis, and detailed information is given in Section 2.4.

2.3 Sector and sensor data

In order to get the occupancy value from the data, it must be manipulated. The way it is manipulated depends on the data source.

2.3.1 Sector data

When using data from the MySQL data warehouse, the occupancy (occ) is calculated by Equation 2.1.

$$occ = \frac{total\,occupancy\,hour}{total\,occupancy\,hour + total\,free\,hour + total\,disconnected\,hour} \tag{2.1}$$

Because sector data in the time series database comes from the hourly sector table, each of the rows is indexed by a timestamp that matches the start of the hour period

for the date where the data was generated. This means that each entry represents the occupancy of a sector for a specific hour of a day. Thus, it is an aggregation of occupancy events during a full hour. A value of 1 for a specific hour indicates that the sector was full the whole hour; analogously, a 0 indicates that the sector was empty for the whole hour.

2.3.2 Sensor data

When using data that comes from MongoDB databases, the sensor occupancy is inferred from sensor status, which is calculated according to Table 2.1. Sensor occupancy coincides with the infrared value, except when infrared status has a "not determined" value, in which case the occupancy is given by the magnetic values.

Sensor status	*Infrared*	*Magnetic*	*Magnetic threshold proximity*	*Sensor occupancy*
0	Free	Free	Very Far	Free
1	Free	Free	Far	Free
2	Free	Free	Near	Free
3	Free	Free	Bottom Line	Free
4	Free	Occupied	Bottom Line	Free
5	Free	Occupied	Near	Free
6	Free	Occupied	Far	Free
7	Free	Occupied	Very Far	Free
8	Occupied	Free	Very Far	Occupied
9	Occupied	Free	Far	Occupied
10	Occupied	Free	Near	Occupied
11	Occupied	Free	Bottom Line	Occupied
12	Occupied	Occupied	Bottom Line	Occupied
13	Occupied	Occupied	Near	Occupied
14	Occupied	Occupied	Far	Occupied
15	Occupied	Occupied	Very Far	Occupied
16	Not determined	Free	Very Far	Free
17	Not determined	Free	Far	Free
18	Not determined	Free	Near	Free
19	Not determined	Free	Bottom Line	Free
20	Not determined	Occupied	Bottom Line	Occupied
21	Not determined	Occupied	Near	Occupied
22	Not determined	Occupied	Far	Occupied
23	Not determined	Occupied	Very Far	Occupied
255	Not determined	Not determined	Not determined	Not determined

Table 2.1: Inferred sensor occupancy.

Sensor data in the time series database comes from the MongoDB data warehouses;

thus, it is indexed by the time stamp of when the event occurred in the sensor.

2.4 Model serving

In order to provide historical information and forecasts for the different parking sector occupancies, we developed a cloud-based tool comprising software architecture of "virtual machines" using Docker. The architecture proposal is shown in Figure 2.3.

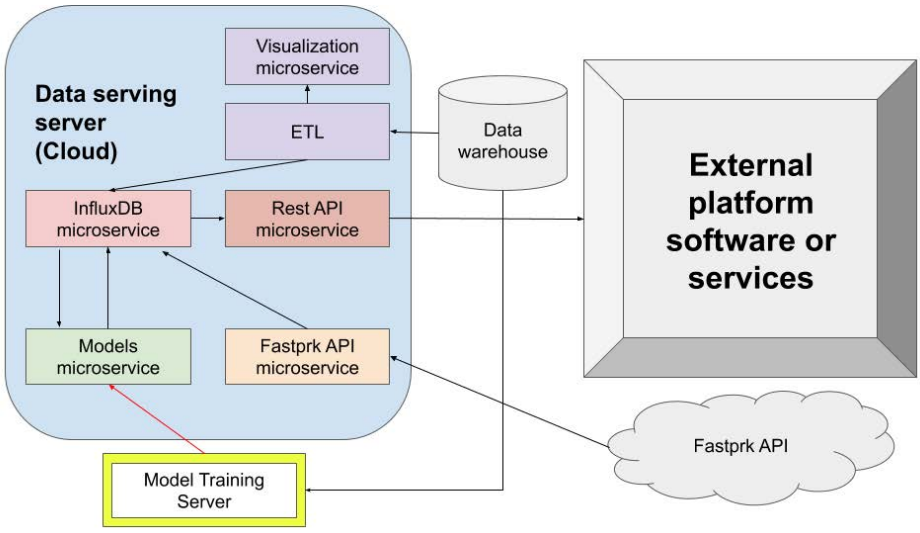


Figure 2.3: Solution architecture. Black arrows represent data flow.

The data server is hosted on the *Google Cloud Platform* and comprises 6 modules (Docker containers - similar to individual virtual machines), i.e., microservices. Each microservice has only one function:

- InfluxDB machine: This stores the data from the ETL process (which itself extracts and receives data from the data warehouse), the model predictions and the Fastprk API.
- ETL: This process extracts the last 24 h of data from the data warehouse to use it for predictions.
- Fastprk API microservice: This module connects to the fastprk API every 10 minutes in order to retrieve the state of parking in real time, and it computes the mean of occupancy for one hour. This information is then stored in the database.
- Model microservice: This module stores the models, then loads and uses them for forecasting. The forecasts are stored again in the InfluxDB database.
- Visualization microservice: This delivers a graphic representation of the historic and forecasted occupancy as a time series. An actual example can be seen in Figure 2.4.

- REST API: This module provides external petitions of the historical and prediction data via a RESTful API.

Finally, forecasting models are created on a dedicated server, namely a computer built for conducting the experiments and training the models offline. They are generated from the data in the data warehouses.

Figure 2.4: Visualization microservice.

3 Data analysis

The last chapter introduced how parking data is generated, transformed and then stored in different data warehouses. This chapter aims to explore the data throughout the different stages of the data pipeline, with special focus on the sector data from the InfluxDB data warehouse (Section 2.1.2). The objective is to find patterns and relevant information that can help in the decision-making process, specifically regarding which kind of data-driven models from the literature are best suited to the problem at hand.

3.1 The nature of sensor data

Sensor data realizations differ from other kinds of data, mainly in that their context must be taken into account. In other words, these data samples have an implicit order that is orchestrated by time. Because of that, our data constitute mere realizations of a random variable over time. If the context given by time is subtracted, the samples become powerless and hardly useful for extracting information or learning patterns in the source system. The correlations that arise from ordered samples in time violate one of the main assumptions from classical statistical methods, namely that data realizations are independent and identically distributed (i.i.d). Thankfully, one area of research has developed mathematics and statistics tools for learning patterns and gathering information from time correlations: this area is known as time series analysis.

The goal of a time series analysis is to identify key properties of the data that provides useful information about the process that generated it that is useful for interpretation and/or characterization. Some of the properties studied are linearity (linear or non-linear relations), stationarity, recurrence (cycles or seasonalities), identification of trends, outliers or noise.

When facing such kinds of data, the first steps always involve generating different plots in which the X axis usually indicates the interval of time when the realizations occur and the Y axis represents the magnitude of the measured effect. Following the data representations, mathematical and statistical tools are used to analyze and extract information from the series. At this point, two different types of methods can be used: those with a time domain approach and those with a frequency domain approach. While the former focus on correlations between lagged samples, the latter study cycles in the series.

Time series are related to the concept of sequence data, although the former is constrained by being ordered in time, the second requires only that the order of the data

has meaning. Examples of sequence data are text, speech or DNA. The methods developed for working with sequence data can also be applied to time series.

As explained in Section 2.3.1, the data in the InfluxDB data warehouse consists of parking occupancy in the different sectors across each hour of the day, with time series indexed by hour being a natural choice of representation. This does not happen in the data from MongoDB data warehouse (sensor data), because it consists of parking spot status based on an external event: a change in the device's measurements, typically when a vehicle parks or leaves the sensor's area of influence. The consequences are that data is not evenly distributed in time, an assumption that is required for some methodologies to develop an analysis or a model. Logically, it is possible to transform the sensor data to the averaged sector occupancy/hour data, but not the other way around.

3.2 Theoretical time series properties

Time series data have a set of mathematical properties that are very useful in order to interpret, identify and characterize the stochastic process that realized them. With these properties it is possible to classify the process in one of the following categories: random walks, Markov processes, Gaussian (or other statistical distribution) processes, random fields, martingales, Lévy processes, renewal processes and branching processes. The most common processes behind time series are random walks, Markov processes and Gaussian processes. When working with time series data the following measurements of dependence are of great importance:

- Mean value function: It is the expected value for each of the random variables that forms the stochastic process. By marginalization of the realizations of the random variables it is possible to find this value. Its expression is shown in equation 3.1. In time series it is of great importance that this value to be the same for all the time points (all the random variables) because that is one of the necessary conditions in order to have regularity in the time series.

$$\mu_t = \mathbb{E}(x_t) = \int_{-\infty}^{+\infty} x f(x)\, dx \tag{3.1}$$

  where $\mathbb{E}$ is the expectation operator, x_t a random variable, x the probability of the observed realization of x_t and $f(x)$ is the marginal density function.

- Autocovariance function: The autocovariance measures the linear relationship between two random variables. In time series analysis it is a necessary condition that the autocovariance function have a dependency on the distance (time lags) between the two random variables considered meaning that the value of the covariance is reduced as further are the realizations in time. Note that the autocovariance function of a random variable with itself equals to the variance of the random variable. Its formula is presented in equation 3.2.

$$\gamma_x(s,t) = cov(x_s, x_t) = \mathbb{E}[(x_t - \mathbb{E}[x_t])(x_s - \mathbb{E}[x_s]) \tag{3.2}$$

where x_s, x_t are the random variables at the time points $s, t \in \mathbb{N}$.

- Autocorrelation function: It is the scaled autocovariance function. Easier to be interpreted because its output value belongs to the interval $[-1, 1]$. The 1 means direct linear dependency (one point increase in one of the random variables equals to one point increase in the other) and -1 means inverse linear dependency (one point increase on one of the random variables equals to one point decrease in the other). A 0 means no correlation between the random variables but this does not mean that the random variables do not have another type of realationship (for example a non linear relation). The expression for the autocorrelation is shown in equation 3.3.

$$\rho_x(s,t) = \frac{\gamma_x(s,t)}{\sqrt{\gamma_x(s,s)\gamma_x(t,t)}} \tag{3.3}$$

where x_s, x_t are the random variables at the time points $s, t \in \mathbb{N}$ and γ is the autocovariance function.

Also of great importance are the following properties:

- Linearity (in probabilistic context): A time series is linear if the realizations of a random variables that generates it is dependent on another random variable through the expression shown in equation 3.4.

$$Y = \alpha X + \beta \tag{3.4}$$

X, Y are random variables and α, β are constants. In fact, α is used as a measure of the linear relationship between both random variables (correlation if the random variables have been standardized) and β is refered as the stochastic term. A geometrical interpretation of linearity can be understood from the point of view of a vector space is if considering that the realizations of two random variables form two different vectors that have been centered then the cosine of the angle between the two vectors equals to its correlation and so, measures the linear dependence between the random variables.

- Stationarity : Related to the concept of regularity over time in the behavour of a time series. A time series is said to be strictly stationary when its probabilistc behaviour (their joint distribution) of every collection of values is identical to that of the collection shifted in time (mean and variance to be constant for all time index value). Because this is hard to accomplish with a real data set, in order to model a time series it is enough that the series to be weakly stationary (also known as second-order stationary) meaning that the mean value function is constant and does not depend on time and that the autocovariance function depends on the time points t and s only through their difference and so they are independent of time. In other words, weak stationarity is accomplished when the mean value function and the autocovariance function are regular. Take note that a strict stationary time series is also weakly stationary and if the distribution of the time series is

Gaussian then a weakly stationary series is also a strict stationary series.

Putting all together, one can use the following definitions in order to model a time series:

- Linear process: It is a linear combination (shown in equation 3.5) of white noise random variables. White noise random variables are defined by $w_t \sim wn(0, \sigma_w^2)$. A great number of time series are generated by a linear process and they can be modeled if the process is causal, meaning that the coefficients are 0 when $j < 0$ or in other words, that the process is independent of the future.

$$x_t = \mu + \sum_{-\infty}^{\infty} \psi_j w_{t-j} \tag{3.5}$$

  where ψ are the coefficients with $\sum_{j=-\infty}^{\infty} |\psi| < \infty$, w_t are white noise random variables and μ is the mean value of the process.

- Gaussian process: A process formed by a n-dimensional random vector where each random variable is indexed by a time index and that all together forms a multivariate normal density function. A Gaussian process is defined by a vector of means (one mean for each random variable of the n-dimensional random vector) and by a covariance matrix which is assumed to be positive definite. Gaussian process have the following properties:

  1. Weakly stationary and strict stationary are equivalent. If a time series is generated by a Gaussian process then it is enough to know the first and second moments in order to know its behaviour.

  2. By the Wold Decomposition theorem a Gaussian process is a causal linear process with $w_t \sim iid\, N(0, \sigma_w^2)$ ($iid\,\mathcal{N}$ stands for independent and identical distributed normal random variables of mean 0 and variance σ_w^2.

All together makes posible to model a time series by using finite sampled points in order to estimate the mean, the autocovariance and autocorrelation functions because when the series is weakly stationary then the population mean, autocovariance and autocorrelation can be estimated by means of averaging the sampled data. Once the measurements of dependence (mean and autocovariance) and the nature of the process that generates the data are known, then it is possible to identify the stationarity of the series and be able to determine a model. For example, if the process that generates the data is a Gaussian process, well-known and interpretable linear models are perfect candidates for the modeling of the series, but if only weak stationarity can be identified/assumed, then a more general family of models can be used at the cost of increased complexity and lack of interpretation.

When a time series is weakly stationary, then the sample mean can be estimated using equation 3.6 with an standard error shown in equation 3.7 that is reduced to the expression σ_x^2/n if the process is white noise (notice that if there is dependence between the random variables then it is possible that $var(\bar{x}) \neq \sigma_x^2/n$). The sample autocovariance function can be estimated through the equation 3.8. Finally, the sample autocorrelation

can be computed by means of the autocovariance function as expressed in equation 3.9 and permits to identify those lags where the correlation is statistically significant.

$$\bar{x} = n^{-1} \sum_{t=1}^{n} x_t \tag{3.6}$$

where n is the total number of realizations and x_t is a realization at time $t \in \mathbb{N}$.

$$var(\bar{x}) = var(n^{-1} \sum_{t=1}^{n} x_t) = n^{-2} cov(\sum_{t=1}^{n} x_t, \sum_{s=1}^{n} x_s) = n^{-1} \sum_{h=-n}^{n} (1 - \frac{|h|}{n}) \gamma_x(h, 0) \tag{3.7}$$

where n is the number of realizations, h the lag or time difference between t and s and γ_x is the covariance function.

$$\hat{\gamma}(h) = n^{-1} \sum_{t=1}^{n-h} (x_{t+h} - \bar{x})(x_t - \bar{x}) \tag{3.8}$$

where n is the number of realizations, h the lag between two time index t and s, x_t a realization at time index t and $\bar{x}$ is the sample mean as defined in 3.6.

$$\hat{\rho}(h) = \frac{\hat{\gamma}(h, 0)}{\hat{\gamma}(0, 0)} \tag{3.9}$$

where $\hat{\gamma}$ is the sample autocovariance function as defined in 3.8 and h is the difference of the time lags t and s.

If x_t is white noise, the distribution of the autocorrelation function for a large n is approximately normally distributed with zero mean and standard deviation $\sigma_{\rho_x(\hat{h},0)} = \frac{1}{\sqrt{n}}$ under general conditions (x_t is iid with finite fourth moment), this permits to assess that observed peaks in the autocorrelation function outside the interval $\pm 2\sigma$ are significant (in other words, 95% of the sampled data lies inside the interval).

All of the above (single time series case) is also applicable when dealing with two time series if the series are jointly stationary (each one of them is weakly stationary) and they fullfil that their cross-covariance function (presented in equation 3.10) only depends on the lag h. The cross-correlation function can be computed by using the cross-covariance function as shown in equation 3.11. When n is large, then the cross-correlation function is normal with mean zero and standard deviation $\sigma_{\hat{\rho}_{x,y}=\frac{1}{\sqrt{n}}}$ if at least one of the processes is independent white noise. Similar to the single time series case, one can estimate the cross-covariance and cross-correlation by using averages of the time series realizations. The cross-correlation function provides information about the leading or lagging relation between two series and its proportion (that is in which lag a peak occurs in the cross-correlation function), making possible to predict the outcomes of one of the series as a function of the other and provide interpretations about the processes.

$$\gamma_{x,y}(h, 0) = \mathbb{E}(x_{t+h} - \mu_x, y_t - \mu_y) \tag{3.10}$$

where x, y are time series and $h, t \in \mathbb{N}$ are time indexes.

$$\rho_{x,y}(h,0) = \frac{\gamma_{x,y}(h,0)}{\sqrt{\gamma_x(0,0)\gamma_y(0,0)}} \tag{3.11}$$

3.3 Data preprocessing

Depending on the source of the data, it will need some basic transformations. Specifically, sector data require some basic preprocessing validations and actions.

First, we must check if there are missing values in the data. Missing values are prone to happen in fresh installations because of the need to calibrate the sensors or caused by gateway failures. In this case, the data is examined and pruned until we obtain data realization for each hour in a continuous fashion, that is, at least one day of realizations without missing values. In long-established installations, missing data is uncommon but can still occur. In those cases, a linear interpolation of the data is made.

The second step is to check that the data have no duplicates. Duplicates can usually happen in new installations, where a sector ID is associated to two different sector names (human error when creating the sector). The duplicate data is removed while keeping the realization with the correct sector name.

Further steps of preprocessing may still be necessary if they are required by the proposed methods. In those cases, the modifications made on the data are further discussed in the corresponding sections.

3.4 Visualizations, methods and descriptive tools

During the analysis the following visualizations, methods and descriptive tools were used to find patterns and provide explanations about the data. As far as this author knows, there are no defined methodologies for analyzing parking time series; I propose the following methodology as a guideline for other researchers in this field:

- First, describe the sector zones through:
  - Tables containing sector information such as name, number of sensors (that can be considered parking spots) and deployment date.
  - Map of the sector locations.
- Generate the first visualization of the data by plotting all the data available for all sectors up to the year 2020, where the abscissa axis represents time and the ordinate axis represents occupancy/hour. These visualizations provide information about possible discontinuities or anomalies in the data.
- Create quantile and descriptive statistics tables for all sectors in order to characterize their statistics and the shape of their density plots.

- In order to visualize patterns at small scales, generate monthly and weekly data visualizations by choosing a random month and a random week from that same month together with the same information under different smoothing methods. This can help visualize the patterns. The applied smoothing methods are the moving average (Equation 3.12) and the exponential weighted average (Equation 3.13).

$$\hat{y} = \frac{1}{k} \sum_{n=0}^{k-1} y_{t-n} \tag{3.12}$$

where $\hat{y}$ is the estimated mean using a window with size k comprised of data realizations $y_.$:

$$\hat{y}_t = \alpha * y_t + (1 - \alpha) * \hat{y}_{t-1} \tag{3.13}$$

where $\hat{y}_t$ is the estimated moving average, α is the decay factor parameter and $\hat{y}_{t-1}$is the previous value of the estimated mean, where distant values become less significant.

- The use of seasonal plots constitute one way to gain an intuitive understanding of whether or not the seasonal patterns detected in the monthly and weekly visualizations are repeated throughout the entire interval for which data is available. In these plots, data is averaged at different periods, and this work has averaged them for day, month, week and year. This indicates how occupancy/hour behaves on average for each different period being considered, and it further characterizes the changing patterns within those periods.
- In order to take a closer look at the time component of the data, it is necessary to use specialized tools and visualizations for the time series:
  - Box plots. These visualizations allow detecting a desired property called stationarity in the time series modeling. In order to model the time series, the patterns must be regular in time, meaning that at some point in time the values of the series will be the same as some past values; thus, the only difference is that the realizations are shifted in time. This does not happen with real data, so a relaxation of this constraint is assumed. This relaxation implies that the mean value function $\mu_t$of the series is constant in time and that the autocovariance function $\gamma(s, t)$ depends only on s and t through their difference $\mid s - t \mid$. .
    Stationarity can be detected from a box plot, as this shows the evolution of the mean and variance over time.
  - Decompositional plots. A time series can be decomposed into three components: trend, seasonality and noise, which can be combined additively (Equation 3.14) or multiplicatively. In order to compute the trend component, the moving average (Equation 3.12) method is most commonly used. Once the trend is computed, the seasonality component can be easily computed by sub-

tracting the trend fromthe observed series and then computing the average for each period. Using yearly data with monthly seasonality as an example, each month is averaged using the same month as in other years (after detrending the series). Now, subtracting each component from the observed series, what is left is the random noise. Decomposition plots help visualize these components by presenting them one by one in order to distinguish the data realizations from those in a stochastic process (white noise), which allows us to identify whether there are patterns based on trends and/or seasonality. Another functionality of the decomposition plot is that it detects anomalies in the series, as their residual values will be higher.

$$x_t = m_t + s_t + e_t \tag{3.14}$$

where m_t is the trend component, s_t is the seasonal component and e_t is the residual component.
It is assumed that the series in this work are result of an additive composition based from the information learned on the other plots.

– ACF and PACF plots (also known as correlogram). One of the main assumptions from some time series methodologies is that the realization at time step t is directly affected by past realizations (causality) and, as time reaches further into the past, the influence of past values becomes zero (regularity). When the relationship between realizations is linear (the plus and stochastic components), the underlying process is identified as an autoregressive process. ACF and PACF plots help identify the number of lags (past time steps) that have a linear influence on the realizations at time t. These plots constitute visualizations of the autocorrelation and partial autocorrelation functions.
Recall that the autocorrelation function (Equation 3.3) is defined as a measurement of the linear relationship between the values of a series and that same series at different time steps. Essentially, it normalizes the autocovariance function to the interval [-1,1] (Equation 3.2).
The partial autocorrelation function (Equation 3.15) allows measuring the correlation of a time series between two time indexes while removing the influence (linear effect) of the intermediate values. In order to accomplish this, the random variables of the correlation are regressed against the random variables that lie at the intermediate lags between the two random variables. In other words, the correlation is made on the conditional expectation of the considered random variables ($\rho_{XY} = corr(X, Y|Z)$ where X, Y, Z are random variables and Z has a linear influence on X and Y). Partial autocorrelation assess the direct influence between two random variables removing the chain of dependences generated by the fact of causality and permits to identify which are the lags that generate the autoregression in the studied process.

$$\phi_{hh} = corr(x_{t+h} - \hat{x}_{t+h}, x_t - \hat{x}_t) \tag{3.15}$$

where $\hat{x}$ is the series of regressed values in a time series and its lagged values at different time steps are a consequence of the regression.

  - Power spectra. One approach that provides all the seasonalities in the data and their importance is plotting the time series in the frequency domain, where periods are highlighted for their importance. To accomplish this, the Fourier transformation is used to convert the data from the time domain to the frequency domain.

- Spatial and temporal correlations can be visualized using the cross-correlation function (equation 3.10). The cross-correlation function works similarly to the correlation function, but in this case the inputs are different time series and they are evaluated by also using lagged values. With this technique, how the occupancy of one sector affects other sectors can be highlighted.

3.5 Scenarios

Whereas the first chapter explained the Worldsensing's motivations (see 1.2.2) and presented the cities where pilot studies were conducted as part of the project, this chapter presents the different scenarios used in this thesis. These scenarios were chosen due to their heterogenic nature in order to, on the one hand, provide different characterizations of parking sectors and, on the other, indicate what must be taken into account when offering parking occupancy forecast services. The scenarios and motivations for them are:

- Antwerp. This scenario is characterized by sectors located in different parts of the city (albeit some are on the same street). Different environments condition their use, such as being at the city's main train station or on a street with shops and restaurants.
- Barcelona. Only one sector is represented in this scenario: a private parking facility at a Barcelona university, which is used for loading and unloading goods for different university services. This scenario can provide information about this kind of parking facility.
- Wattens. The two sectors presented are part of a private parking facility, but this time they belong to a private company rather than a university. We expect to find patterns linked to the company's working hours and provide extra information about private parking facilities.
- Los Angeles. This scenario provides sectors where parking places are destined to different types of users. These sectors are characterized by payment, parking permit, use by disabled people, and use by a carsharing company. Also, these sectors are next to a metro station and thus add environmental noise to the sensor measurements, which could affect the results of data-driven models.

- Riyadh. This scenario is studied because its large number of sensors (more than one thousand), sectors established next to each other that span all parking spots in a large zone, and the possibility of detecting patterns and behaviours that are not found in the Western world.

The Paris and Grudziadz scenarios have not been considered in this work, as they do not provide any new features compared to the other scenarios, although their sectors have also been modeled as part of the company's project.

This section presents an analysis of the studied scenarios, as well as their characteristics, in order to obtain a better understanding of the underlying processes that govern parking systems.

3.5.1 Antwerp (Belgium)

Characteristics

The city of Antwerp was one of the first where Worldsensing's sensors were deployed. Table 3.1 shows the sectors of the city and the number of sensors in each one, as well as the installation dates. Notice that the sectors on the streets Pelikaanstraat and Van Wesenbekestraat were the first to be installed.

The main characteristics of these sectors is that they are located on-street and near metro railways, making them vulnerable to magnetic noise. The images in Figure 3.1 illustrate the position and number of sensors in the different sectors of the city. At first sight of the point at which the Pelikaanstraat sector is far away from the other sectors, without connecting streets, it is natural to think that no spatial correlations exist in this specific case.

The Pelkaanstraat sensors constitute unique characteristics of the sector in that they are placed just outside the Antwerp Central Station, the central hub for trains and metro departures, thus making it a place used mainly by taxis.

The sectors in Van Wesenbekestraat are situated in what the residents of Antwerp commonly call Chinatown or Asiantown, a district full of restaurants and shops.

Another sector with a unique characteristic is Van Schoonhovestraat Zone 2, as its parking spots are intended for loading and unloading freight vehicles.

Analysis of all available data for the entire period

An overview of all historical data can help take a wider view of the data in order to detect trends and anomalies at a grand scale.

Figure 3.2 shows data realizations for the sector Pelikaanstraat. At first sight, it seems that in May 2019 an abrupt change occurred in the realizations, leading to an obvious reduction in the variance. This could be an indication that some of the sensors were

Sector name	*Number of sensors*	*Deployment date*
Pelikaanstraat	35	01/02/2018
Van Wesenbekestraat Zone 2	12	01/02/2018
Van Wesenbekestraat	10	01/02/2018
Van Schoonhovestraat Zone 3	8	25/02/2019
Van Schoonhovestraat Zone 1	8	25/02/2019
Van Schoonhovestraat Zone 2	3	25/02/2019

Table 3.1: Antwerp sectors ordered by number of sensors.

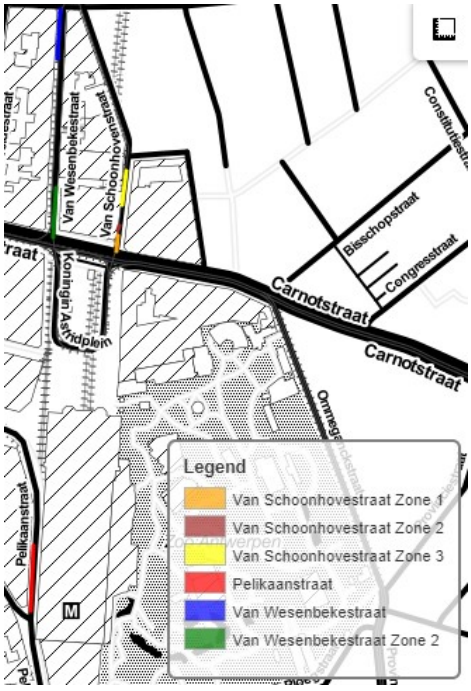


Figure 3.1: Antwerp map. The image shows sector locations, differentiated by color.

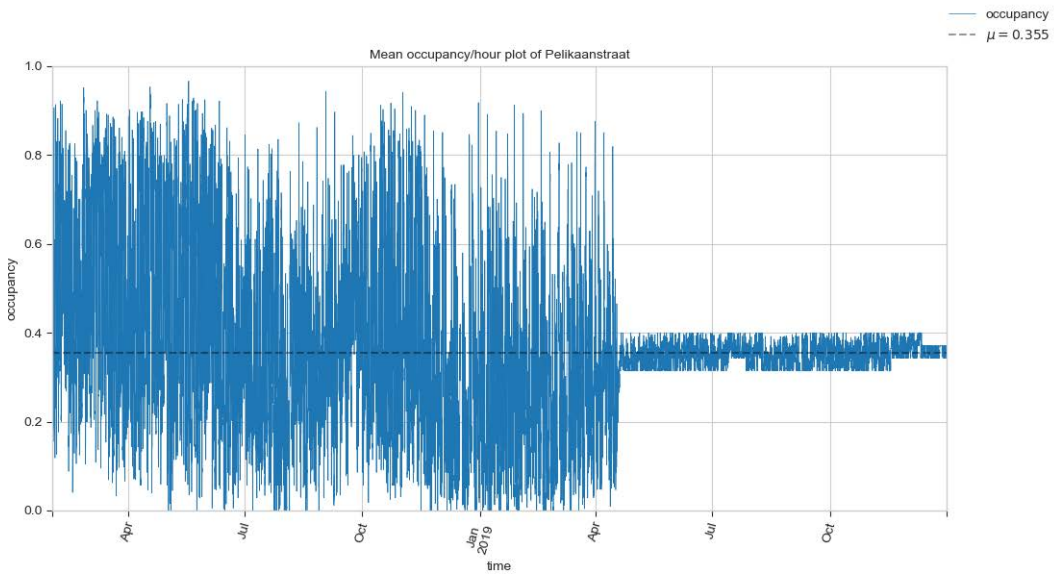


Figure 3.2: Pelikaanstraat data from 2018 and 2019.

removed, their batteries died, or street works affected them. This makes obvious that it is imperative to keep an eye on sudden changes in the data realizations.

Both sectors along the street Van Wesenbekestraat (see Figure 3.3) show similar anomalies from mid-2018 until the end of the year. In this case, it is known that the sensors were shut off for that period of time.

Data realizations known to originate from sensor malfunctions are removed in order to perform the analysis. For example, in the Pelikaanstraat sector, data realizations after 18 April 2019 are considered invalid and not taken into account. This is also the case for the Van Wesenbekestraat sectors, where data prior to 12 December 2018 is removed.

Figure 3.4 shows the occupancy for the sectors along the street Van Schoonhovestraat. In this case, no change in behaviour or anomaly is detected at first sight. Notice that in sectors with small numbers of parking spaces (Zone 2 has 3 sensors only), it is harder to detect periodic patterns because the series becomes non-smooth. This suggests that sectors with large numbers of sensors will have smoothed series in the transition from full to empty occupancy.

Statistics

The statistics of the Pelikaanstraat sector (Table 3.2) show that it has never been fully occupied for an hour and that its occupancy levels are usually below 50%. Of the 35 sensors (parking spots), it is rare for more than 28 to be occupied at once within a one-hour period. The density plots for each month of the year 2018 (Figure 3.5) confirm the statistics visually and also offer new insight: for some of the months, it is obvious

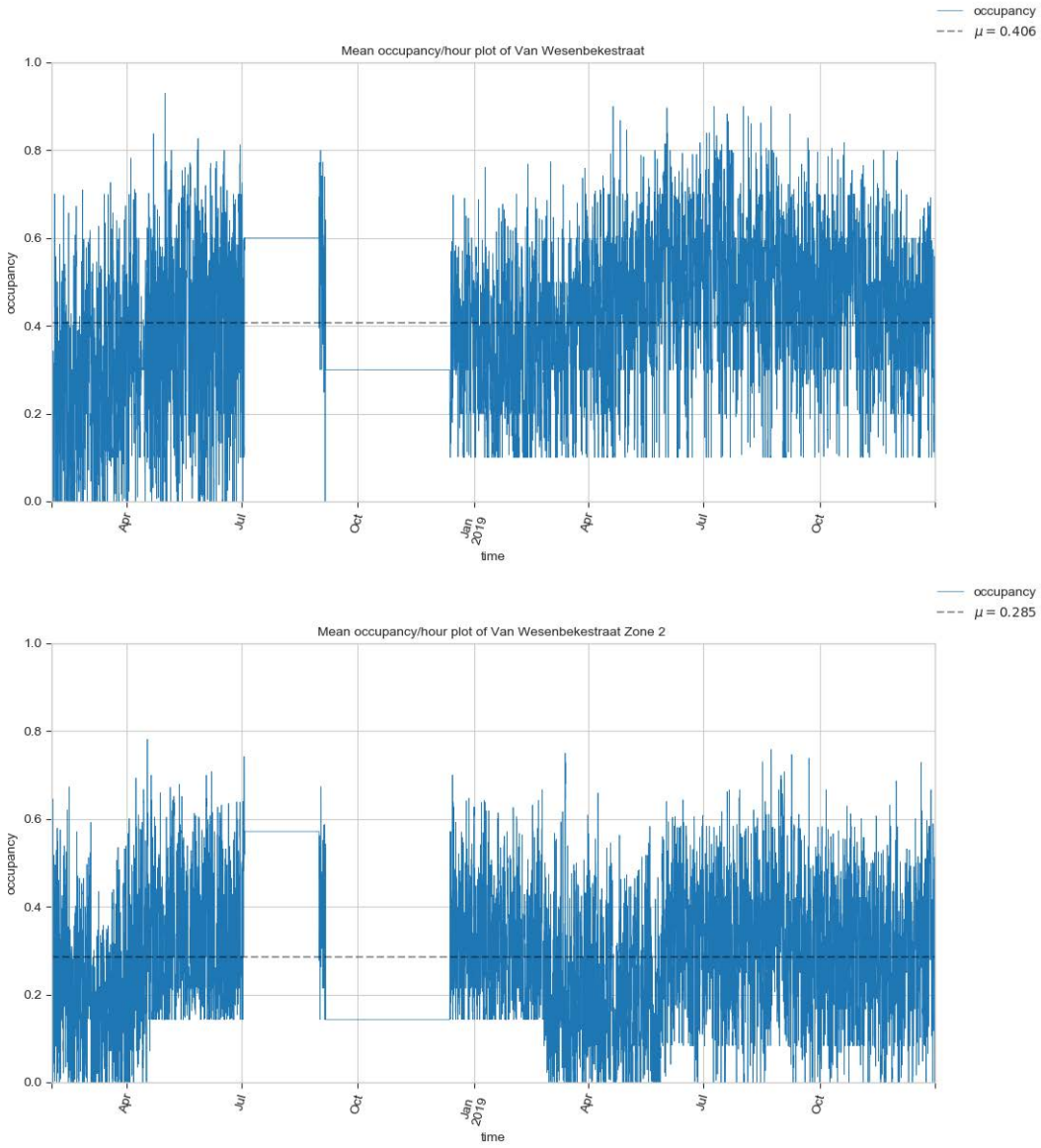


Figure 3.3: Data for Van Wesenbekestraat sectors, 2018 and 2019.

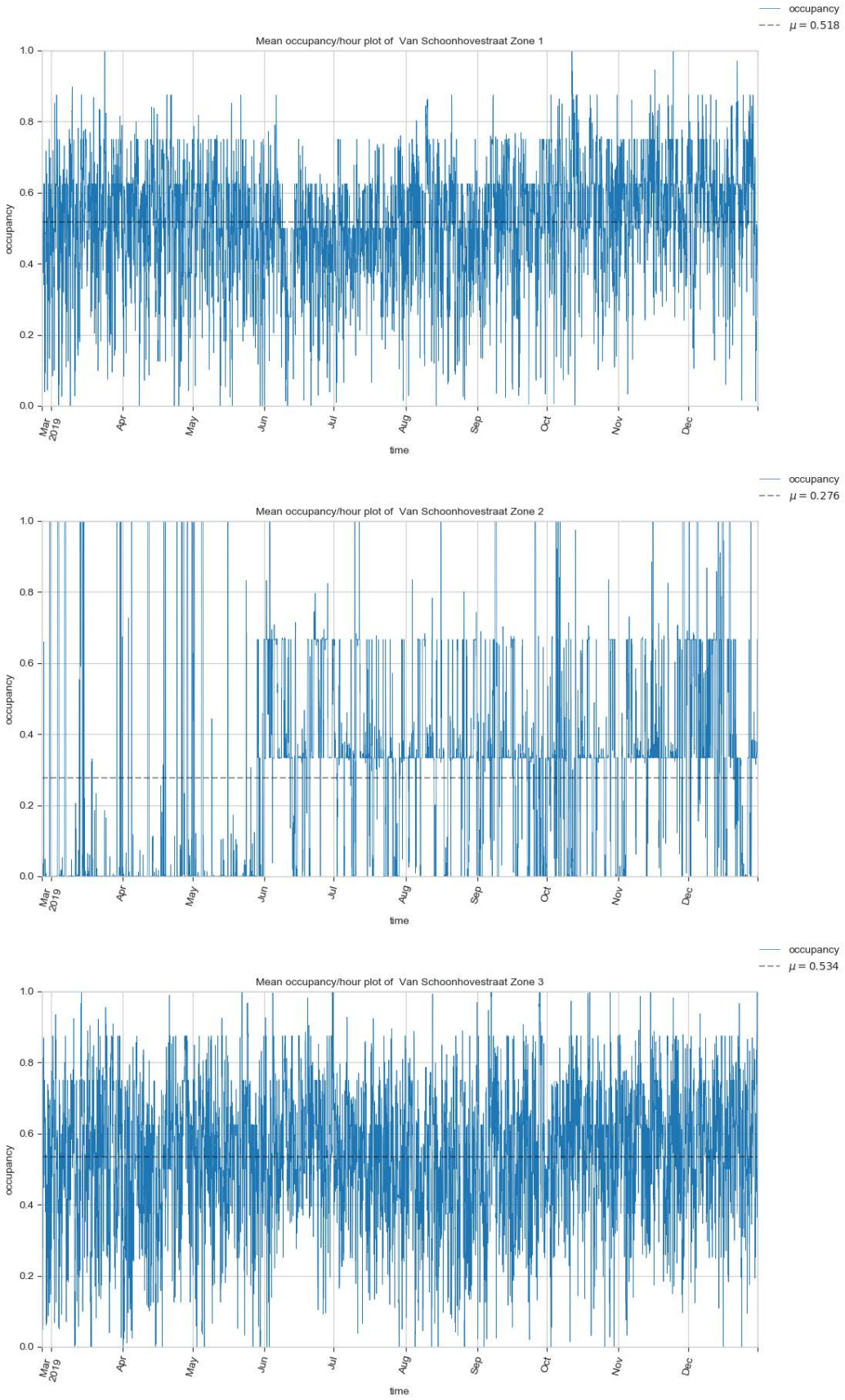


Figure 3.4: Data for Van Schoonhovestraat sectors, 2019.

Quantile statistics	
Minimum	0
5-th percentile	0.057
Q1	0.172
Median	0.314
Q3	0.514
95-th percentile	0.795
Maximum	0.967
Range	0.967
Interquartile range (IQR)	0.342

Descriptive statistics	
Standard deviation	0.229
Coefficient of variation	0.641
Kurtosis	-0.637
Mean	0.357
Median Absolute Deviation	0.191
Skewness	0.56
Variance	0.052

Table 3.2: Statistics for the Pelikaanstraat sector.

that the distributions have two peaks – one at 0.25 and the other at 0.75. When the two peaks are not present, the distribution has a positive skew. Another important fact is that the variance value is higher than for the other sectors.

For the Van Wesenbekestraat sectors (3.3), quantile statistics reveal that there is a particularity in Zone 1 and that its minimum and maximum values (0.1 and 0.9, respectively) reveal that the sector has never been empty or fully occupied. This could be an indicator of a possible problem with one of the sensors. Density plots for Zone 1 (Figure 3.6) indicate lower values for the first months of the year, but they maintain a very symmetric shape resembling a Gaussian curve.

The tables of statistics for Zone 2 (3.3) indicate lower occupancy/hour overall for the quantile statistics, with a mean value of 0.282. The descriptive statistics are similar to those from Zone 1, except for the skewness, which is negative in Zone 1 and positive in Zone 2. This is more notable for the first months of the year, as can be seen in Figure 3.6.

Lastly, we can see in the Van Schoonhovestraat sectors that Zones 1 and 3 behave similarly, as they are on the same street and have the same number of sensors. Their more notable difference is that Zone 1 has a leptokurtic shape (positive kurtosis), while Zone 3 has a platykurtic shape (negative kurtosis). On the other hand, the statistics for the sector in Zone 3 are very different from the other two, probably because of the sector's small number of sensors (3 sensors). The reason these parking spots are treated in a different sector rather than as being part of a sector in Zone 1 (since they are contiguous) is because Zone 3 parking places are marked as loading and unloading zones for truck deliveries.

Lastly, observe that the variance of Zone 2 is greater than that of any other sector in the city (the second greatest is Zone 2 in Pelikaanstraat). Both sectors coincide in that they have non-Gaussian shapes (for Pelikaanstraat, this happens over some months), with more than one peak. This could be indicative of the type of parking places, as both sectors pertain to places associated with short stays.

Quantile statistics	
Minimum	0.1
5-th percentile	0.139
Q1	0.304
Median	0.440
Q3	0.549
95-th percentile	0.698
Maximum	0.9
Range	0.8
Interquartile range (IQR)	0.244

Descriptive statistics	
Standard deviation	0.162
Coefficient of variation	0.374
Kurtosis	-0.48
Mean	0.433
Median Absolute Deviation	0.133
Skewness	-0.074
Variance	0.026

Quantile statistics	
Minimum	0
5-th percentile	0.04
Q1	0.168
Median	0.282
Q3	0.385
95-th percentile	0.519
Maximum	0.758
Range	0.758
Interquartile range (IQR)	0.217

Descriptive statistics	
Standard deviation	0.146
Coefficient of variation	0.517
Kurtosis	-0.495
Mean	0.282
Median Absolute Deviation	0.12
Skewness	0.106
Variance	0.021

Table 3.3: Statistics for the Van Wesenbekestraat sectors. Top, Zone 1; bottom, Zone 2.

Quantile statistics	
Minimum	0
5-th percentile	0.233
Q1	0.425
mMedian	0.526
Q3	0.625
95-th percentile	0.75
Maximum	1
Range	1
Interquartile range (IQR)	0.2

Descriptive statistics	
Standard deviation	0.163
Coefficient of variation	0.315
Kurtosis	0.249
Mean	0.518
Median Absolute Deviation	0.128
Skewness	-0.5
Variance	0.027

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0
Median	0.333
Q3	0.345
95-th percentile	0.667
Maximum	1
Range	1
Interquartile range (IQR)	0.345

Descriptive statistics	
Standard deviation	0.268
Coefficient of variation	0.97
Kurtosis	-0.064
Mean	0.276
Median Absolute Deviation	0.218
Skewness	0.733
Variance	0.072

Quantile statistics	
Minimum	0
5-th percentile	0.189
Q1	0.409
Median	0.544
Q3	0.663
95-th percentile	0.835
Maximum	1
Range	1
Interquartile range (IQR)	0.255

Descriptive statistics	
Standard deviation	0.192
Coefficient of variation	0.359
Kurtosis	-0.067
Mean	0.534
Median Absolute Deviation	0.153
Skewness	-0.355
Variance	0.037

Table 3.4: Statistics for the Van Schoonhovestraat sectors. From top to bottom: Zones 1, 2 and 3.

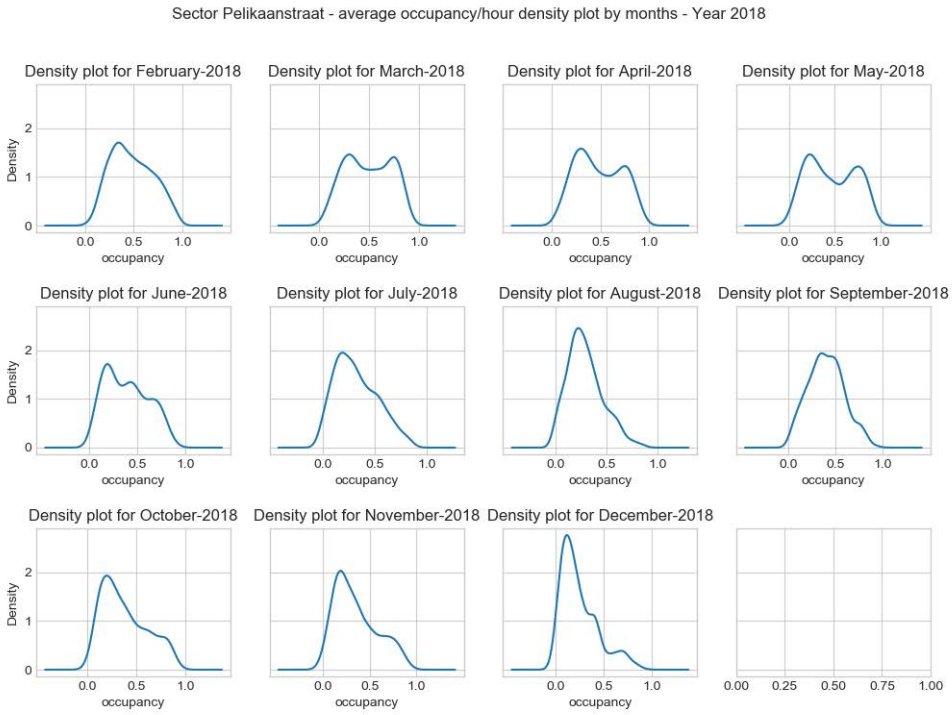


Figure 3.5: Density plot for the year 2018, Pelikaanstraat sector.

Monthly and weekly data analysis

When representing the data over shorter time intervals (like monthly or weekly), it is easier to visualize possible patterns. In Figure 3.8, we can see that there is a daily pattern throughout all February realizations as occupancy increases and decreases in a similar fashion every day. At 6 a.m., parking occupancy starts to increase until peaking at midday; and then the levels drop until 6 p.m., where it remains more or less constant until midnight, when it reaches the lowest occupancy of the series.

In the example of week 2 for February 2018, the early-morning levels of the series drop lower for Saturday and Sunday, indicating that maybe there are different patterns in the data realizations for weekdays and weekends.

Daily patterns can be seen more clearly when smoothing the series. Figure 3.9 illustrates the occupancy series of the second week of February 2018 for the Pelikaanstraat sector, together with the series after it has been processed by different smoothing algorithms. The algorithms used are a windowed moving average and an exponential weighted average (Equations 3.12 and 3.13).

Using a moving average with a window size of twelve hours, we can clearly see the daily periods of the data and how the levels change for weekends, where occupancy decreases and seems to stabilize at a value under the mean. This suggests that the data can have multiple periodicities in which one of them is determined by the hours of a day (period

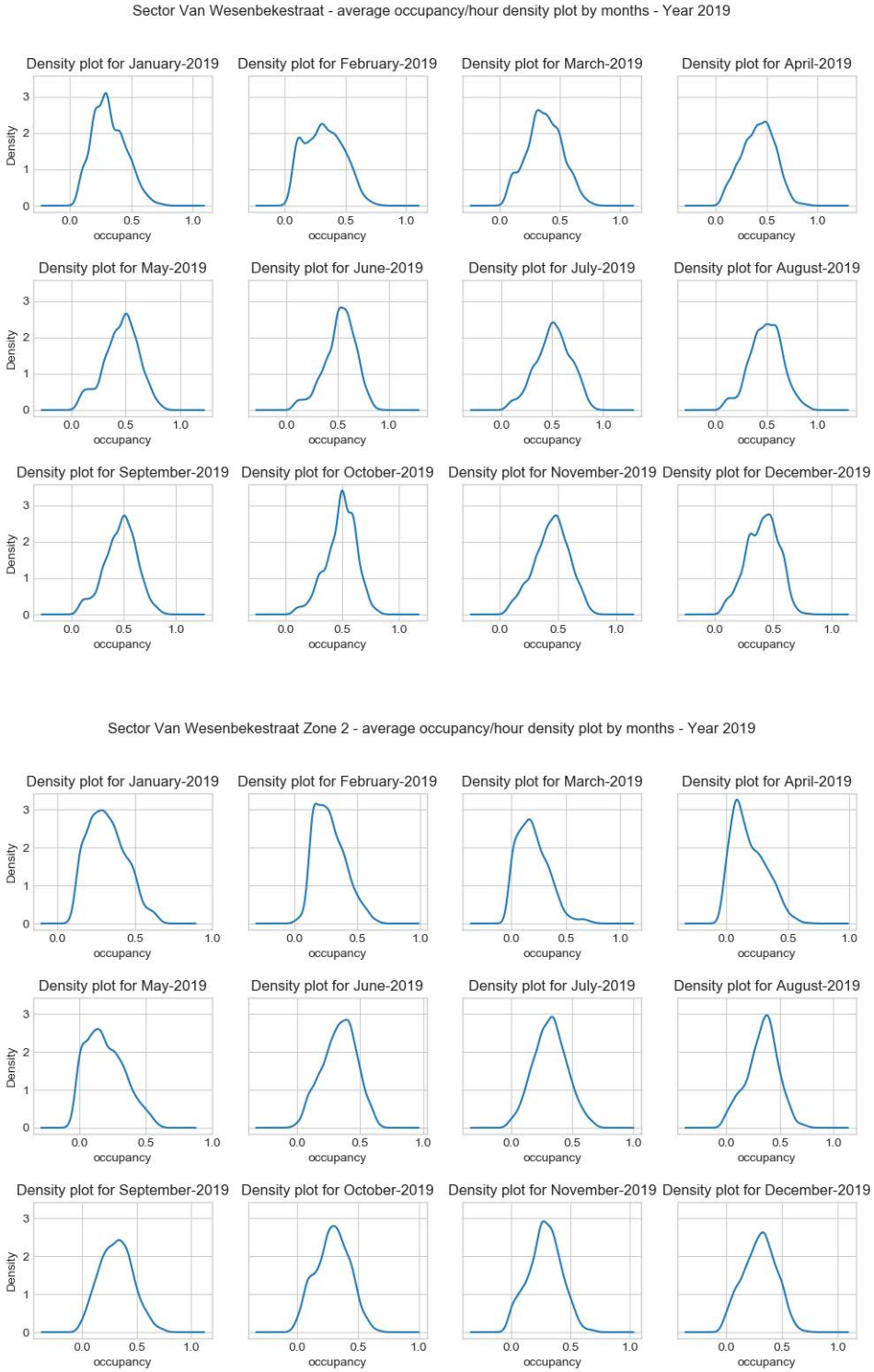


Figure 3.6: Density plot for the year 2019, Van Wesenbekestraat sectors.

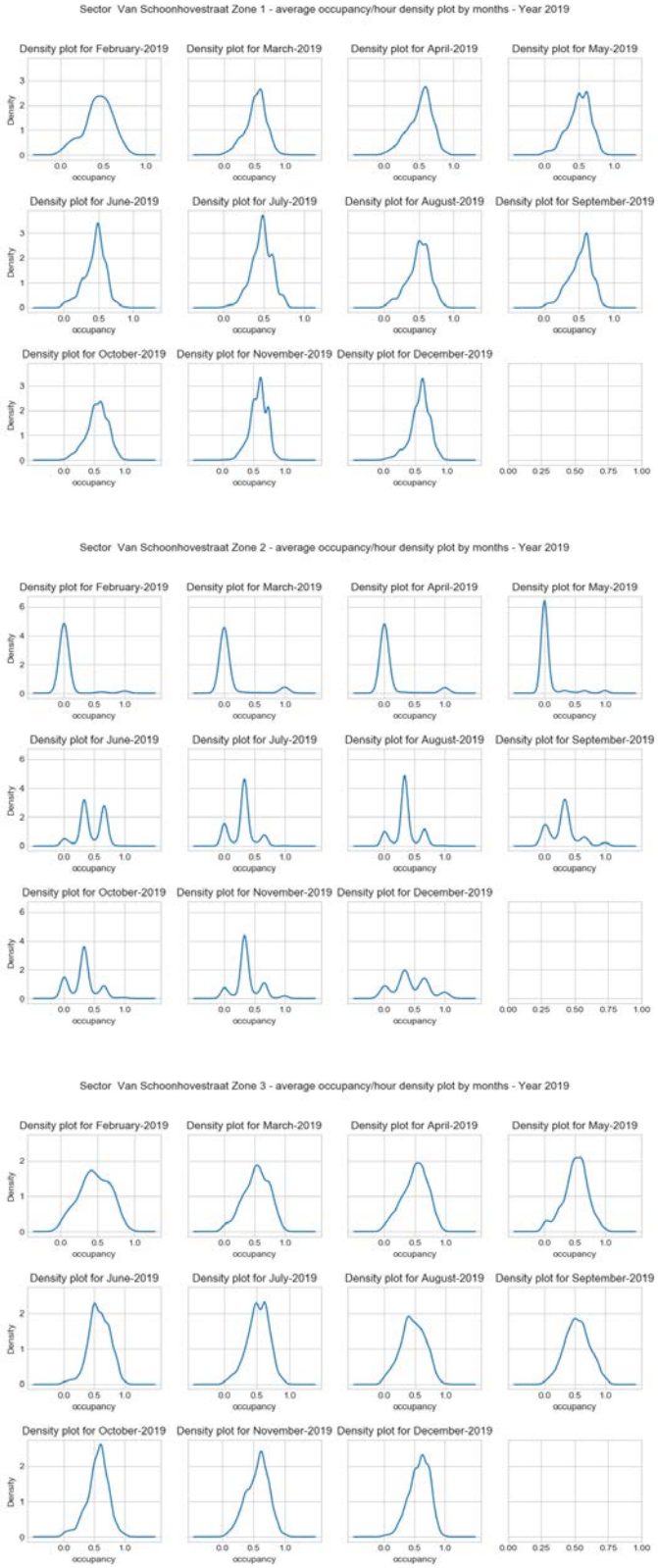


Figure 3.7: Density plot of year 2019 for Van Schoonhovestraat sectors.

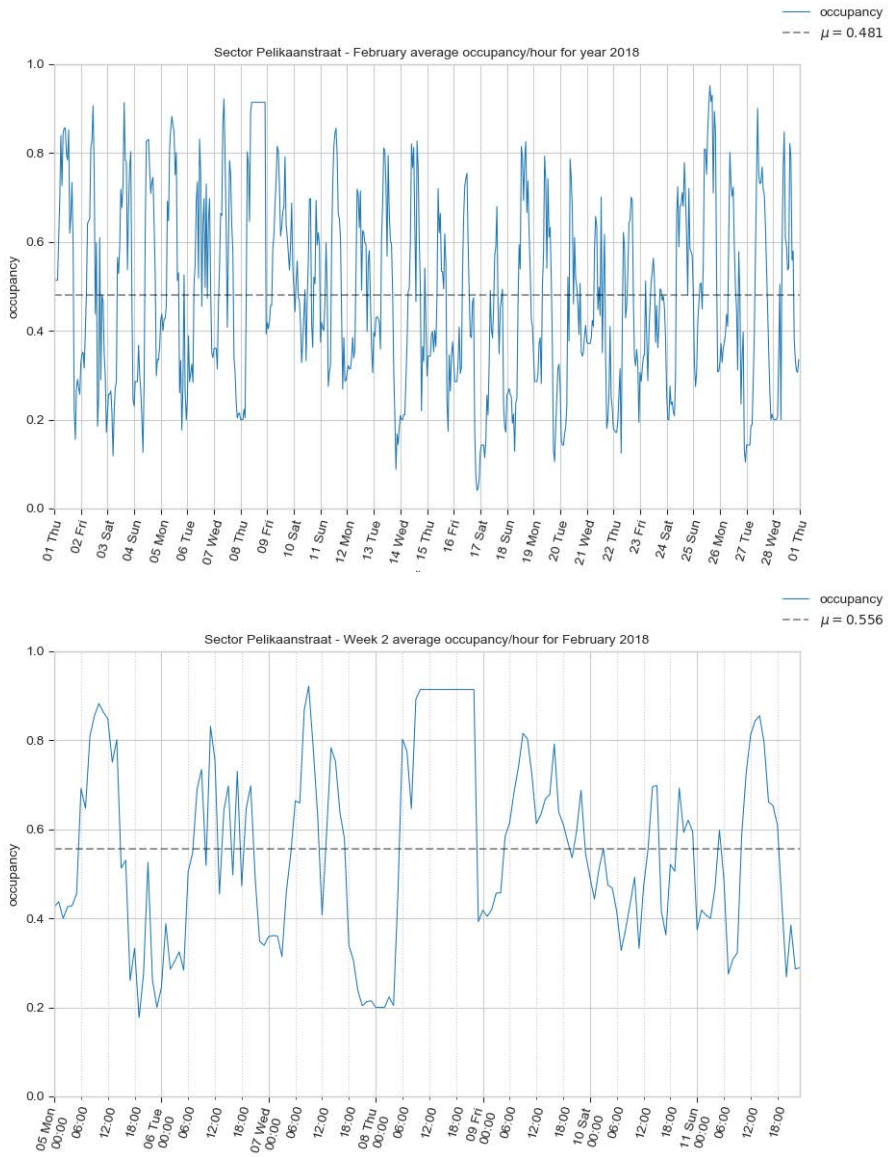


Figure 3.8: Antwerp time series from the Pelikaanstraat sector for February 2018 and the second week of February 2018.

of 24 hours) and another by the day of the week (period of 168 hours, that is, 7 days).

Figure 3.10 shows the occupancy/hour plotted for each day of the week, month, week and year, thus giving information about how occupancy behaves on average during the different time periods.

Overall, occupancy/hour behaves similarly, regardless of the period used for averaging the data (day, month or year). Its levels are higher during daytime hours and lower at night. When plotting for the day of the week, notice that occupancy on Saturday and Sunday has its maximum value at 15 p.m. while this happens at 10 a.m. on weekdays, thus indicating different behaviours for weekdays and weekends. It is interesting that Sunday is the day of the week with the greatest occupancy values; this could mean that parking stays are longer on this day due either to different train schedules or to different parking policies applied for that day. About the patterns throughout the day, one can conclude that the sector is not used at night (as the occupancy/hour levels are nearly 0) and its main usage coincides with train schedules (at the moment of writing this thesis, the train station's opening hours was Monday to Sunday, from 05:45 to 22:00). This could be the reason for the two peaks in the distributions of some months. Other assumptions are that the parking places allow only taxis or that, as the sector is in the very center of the city's downtown area, few residents leave their cars in on-street parking.

In the plot for the months of the year, the lowest occupancy levels correspond to the months of August, December and January; while the month with the most occupancy is May.

Interestingly, when plotting the weeks, weeks 1 and 6 behave differently than the other weeks, which resemble the Saturday and Sunday patterns (this could be an effect of when holidays occur on the calendar).

Lastly, the pattern is maintained independently of the year, but the occupancy levels in 2019 are lower than those of 2018. The reason for this could be that the data used for the 2019 yearly average takes into account only the first 5 months.

Seasonal plots for Van Wesenbekestraat's Zone 1 (Figure 3.11) show very different behaviour when compared to the sector on Pelikaanstraat. While the latter's occupancy levels follow the train station's opening hours and it is easy to differentiate between weekdays and weekends, occupancy in the former throughout the weekdays behaves similarly to weekends, with occupancy levels peaking at 11 a.m. and slowly declining until 5 a.m. Focusing more on weekdays and weekends, it is possible to detect a differential trait between weekdays and weekends (that can possibly be generalized to workdays vs. holidays), namely that on weekdays the occupancy levels fall until 15 p.m., then rise again until they fall again at 18 p.m. This may be because they are affected by the shops' opening hours and their closing at lunchtime. This pattern does not occur on weekdays, as the occupancy/hour levels fall from 13 p.m. to 5 a.m., only on Saturday, with some increase from 20:00 to 23:00, perhaps as a result of people visiting restaurants for dinner.

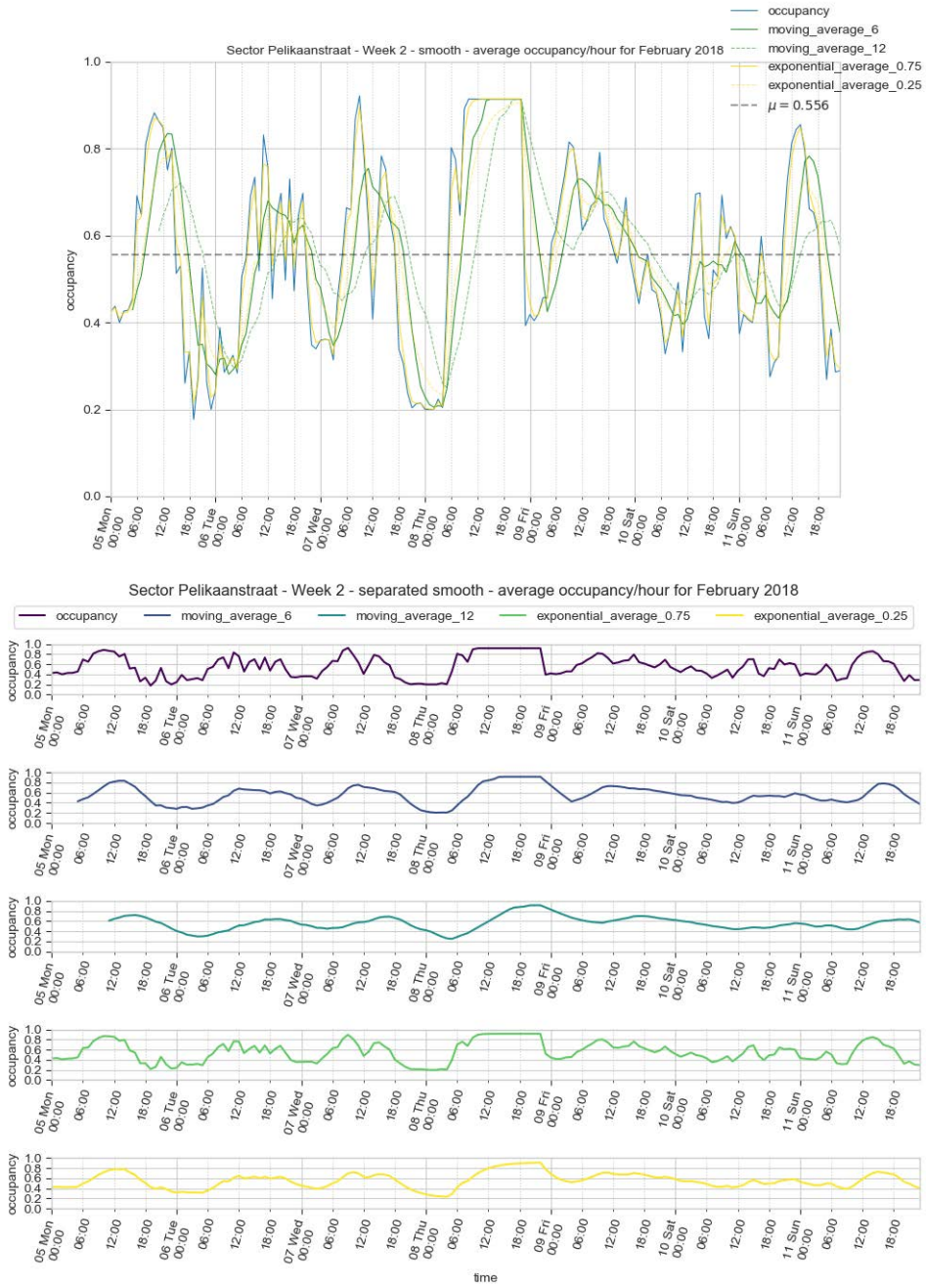


Figure 3.9: Example of smoothed series for the Antwerp sector along Pelikaanstraat street for February 2018. The figure shows the series layered one on top of the other, where green lines represent the window moving average series and the yellow ones the exponential weighted average results. The bottom figure decomposes the top plot and shows the smoothness of the series under different methods.

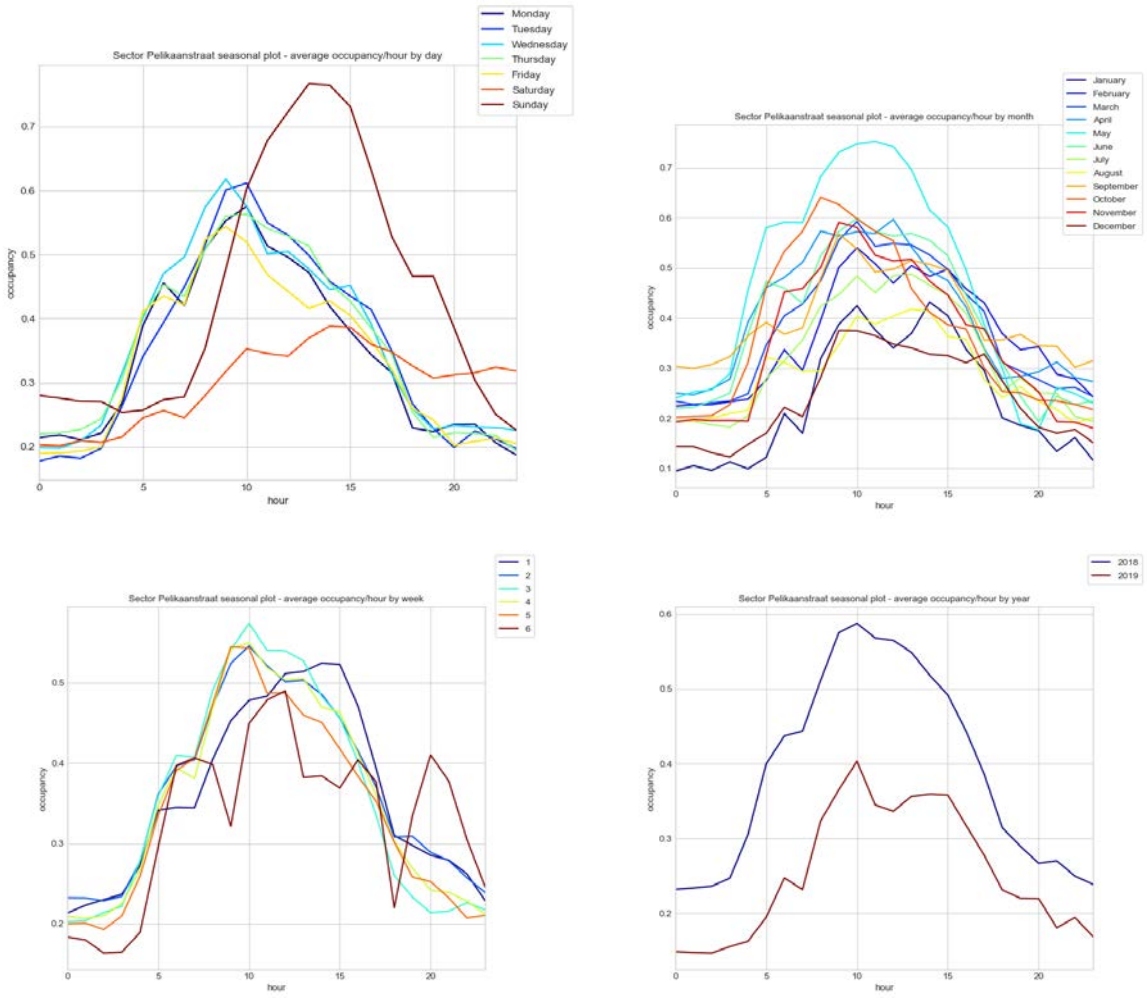


Figure 3.10: Seasonal plots for the Pelikaanstraat sector. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month, and year.

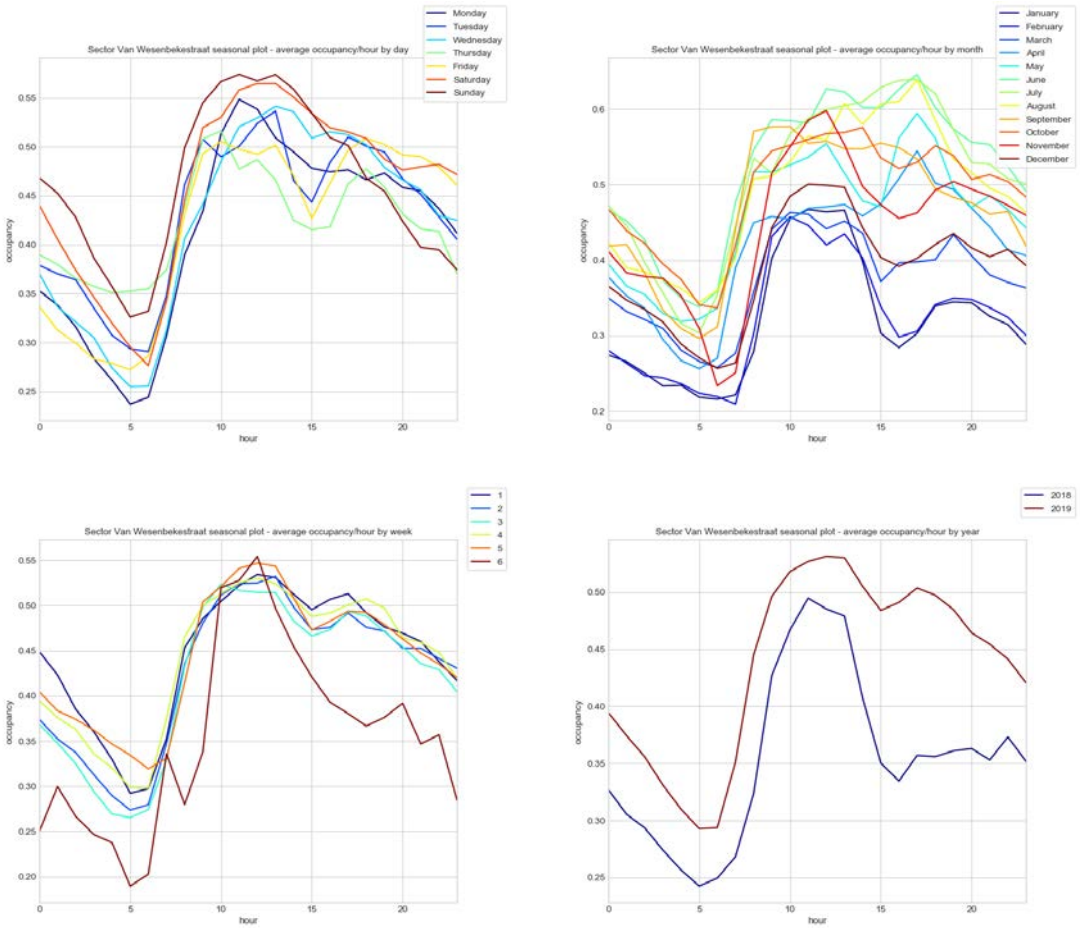


Figure 3.11: Seasonal plots for Van Wesenbekestraat's Zone 1 sector. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

Monthly aggregation shows that the occupancy is lower for the first 3 months of the year.

Weekly patterns (first plot of the second row in Figure 3.11) show similar behaviour to the daily aggregation throughout the weeks.

Lastly, the occupancy levels for 2018 are lower than those for 2019. In particular, parking behaviour at 15 p.m. is notably lower in 2018 than in 2019. Remember that the Van Wesenbekestraat data is mainly from all of 2019, while the 2018 data is from only December.

The seasonal plots for Van Wesenbekestraat's Zone 2 sector (Figure 3.12) show similar behaviour to Zone 1 (as expected, since they are on the same street), but with slightly lower occupancy levels. The behaviours that distinguished weekdays and weekends no longer occur. This is because the sector is right next to the restaurants along that street and therefore the behaviour is more pronounced on Friday and Saturday, when

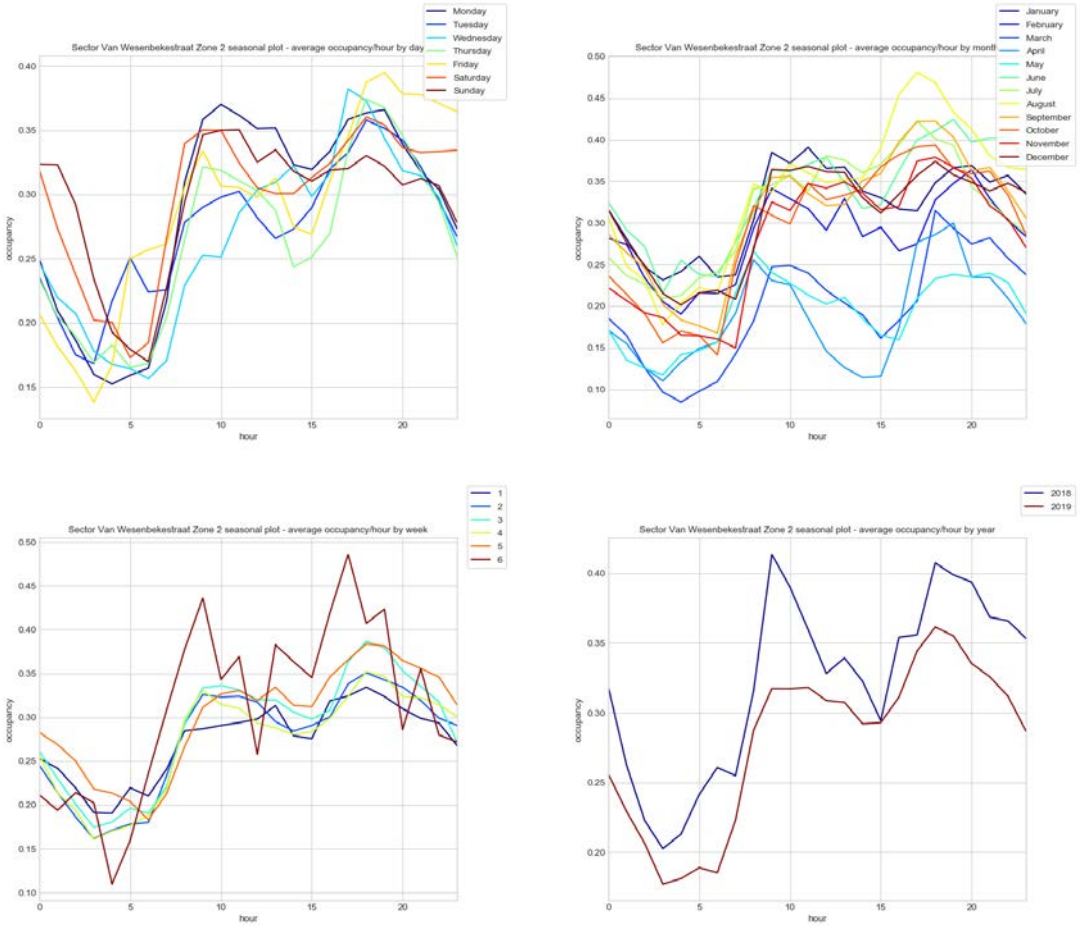


Figure 3.12: Seasonal plots for the Van Wesenbekestraat Zone 2 sector. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

the occupancy levels increase from 15 p.m. until 19 p.m. and remain high but decrease slowly into the night. Wednesday patterns are interesting, as the occupancy/hour levels increase throughout the day and peak at 16 p.m., which is different from other working days when levels peak at 10 a.m.

Monthly aggregation shows more or less the same patterns as the daily aggregation, but behaviour is more like the weekends, when occupancy peaks at around 19 p.m. to 20 p.m.. This happens especially on months with holidays (August and December).

Weekly aggregation shows similar patterns for all weeks when not taking into account week 6. The behaviour is similar to that seen in the daily aggregation.

Lastly, yearly aggregation shows slightly more occupancy for 2018, especially at 10 a.m.

The Van Schoonhovestraat Zone 1 seasonal plots show that the levels change little throughout the hours of the day, with a slight decrease at 5 a.m. Focusing on the

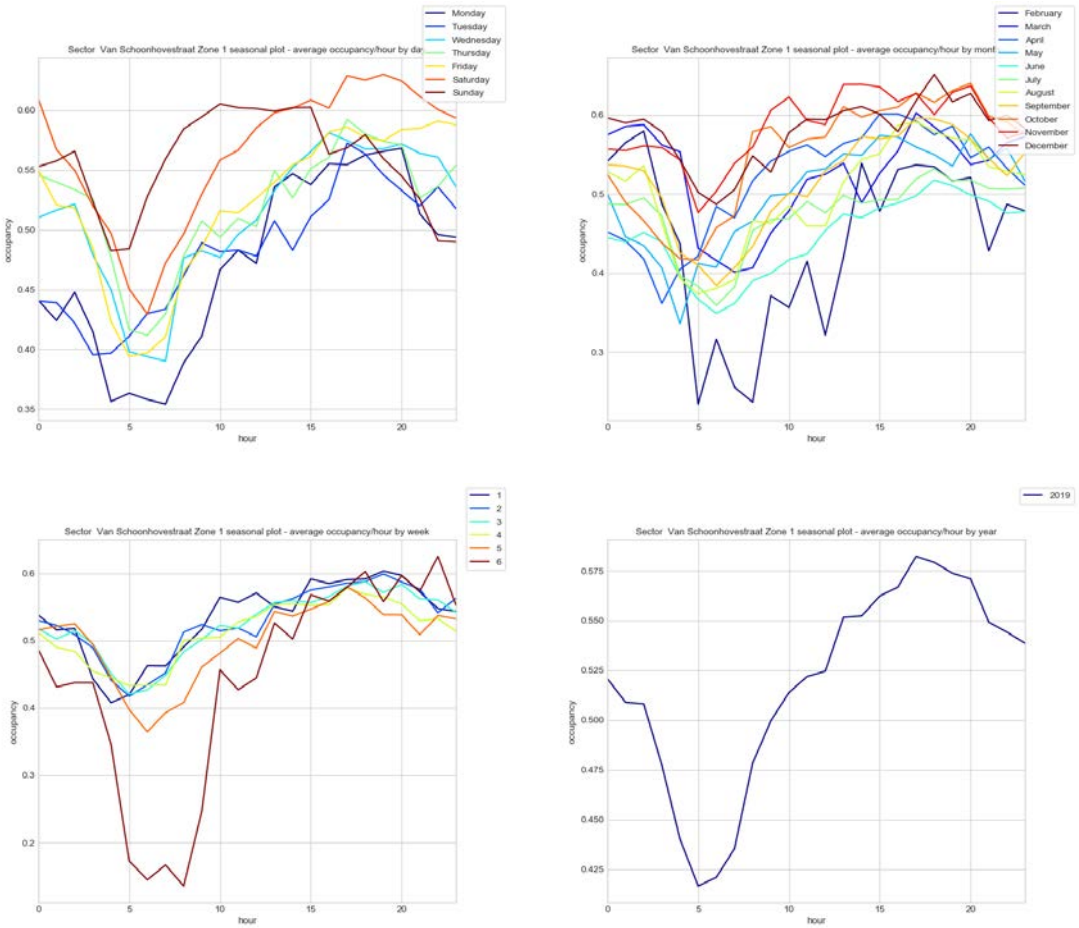


Figure 3.13: Seasonal plots for the Van Schoonhovestraat Zone 1 sector. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

first visualization in Figure 3.13, we can see that all days behave similarly, with slightly higher occupancy levels on Saturday and Sunday.

The second visualization shows the monthly aggregation in which all months behave similarly, with only February standing out with its comparatively lower levels.

Similarly to the previous sector, all weeks behave similarly (not taking into account week 6). The similarity between week 6 and the whole year 2019 is surprising.

It is difficult to extract any pattern from the seasonal plots in Figure 3.14 (the Van Schoonhovestraat Zone 2 sector). The occupancy levels seem to reach their peak usually between 6 a.m. and 10 a.m., which occurs throughout all the different components of each visualization.

The seasonal plots for the Van Schoonhovestraat Zone 3 sector (see Figure 3.15) show similar patterns to the Van Schoonhovestraat Zone 1, with slightly higher occupancy

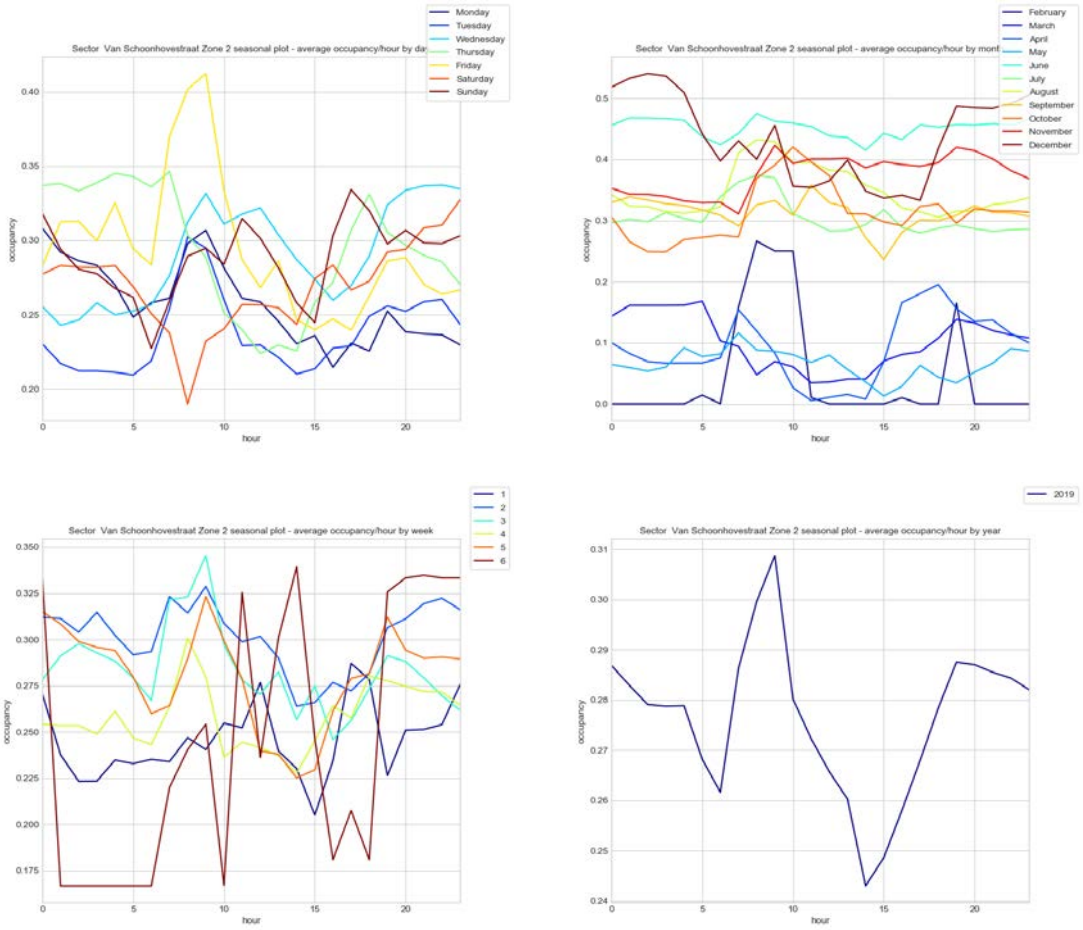


Figure 3.14: Seasonal plots for the Van Schoonhovestraat Zone 2 sector. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

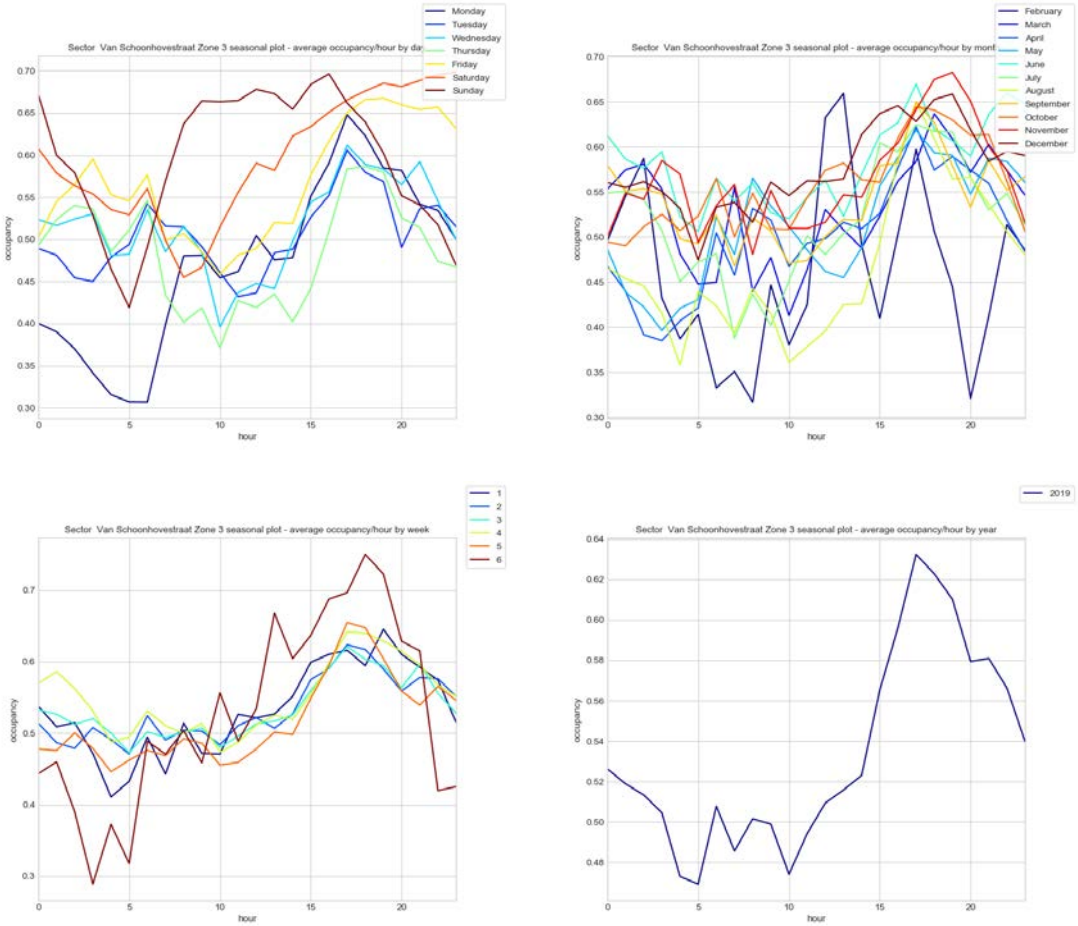


Figure 3.15: Seasonal plots for the Van Schoonhovestraat Zone 3 sector. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

levels. The daily aggregation behaviour is more or less identical throughout the days with Friday and Saturday with higher occupancy levels at latter hours of the day and Sunday with higher occupancy levels at daytime when compared to the other days. Monday indicates lower occupancy levels at 5 a.m. with respect to the other days of the week.

In the monthly aggregation, all months seem to behave more or less equally, with the highest occupancy occurring towards the end of the day. This is not the case for February, as it indicates abrupt changes in occupancy levels throughout the hours.

Weekly patterns are shared throughout the 6 weeks and are similar to the behaviours in the yearly aggregation.

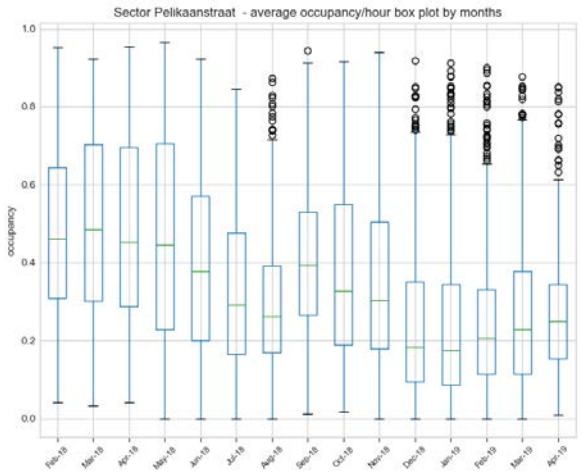


Figure 3.16: Box plot for the Pelikaanstraat sector.

Specialized time series plots

The box plot for the Pelikaanstraat sector (Figure 3.16) shows that the occupancy levels decrease over time, although the variance seems to be constant.

The box plots for Van Wesenbekestraat show similar behaviour for both sectors, where the variance remains constant through time while the mean oscillates between higher values in summer and lower values in winter.

Lastly, Van Schoonhovestraat's Zones 1 and 3 behave similarly, with more or less the same mean and variance over time. The sector in Zone 3 presents stochastic behaviour, evidencing that finding patterns is harder for smaller sectors.

The decomposed time series for the Pelikaanstraat sector (Figure 3.19) shows a decreasing trend (already observed in Figure 3.16) when taking into account the whole year of 2018 (using occupancy/day). However, when using occupancy/hour and focusing on a specific month (April), there is no trend. For occupancy/day, the seasonality is weekly, with two peaks that represent the start of the week and the weekend. In the occupancy/hour plot, the seasonality clearly follows a daily pattern. Lastly, residuals from the occupancy/day plot are lower than those of the occupancy/hour plot.

The decomposed components of the series for sectors on the street Van Wesenbekestraat indicate different trends and seasonal components. Zone 2 is more stable in its seasonality and trend levels, which is similar to the conclusions extracted from Figure 3.17.

The sectors on Van Schoonhovestraat show that Zone 2 behaves erratically in its trend values, with large residual values for some realizations. Zone 1's behaviour can be better explained by this approach compared to that of Zone 3, as residuals are enclosed in the [-1,1] interval.

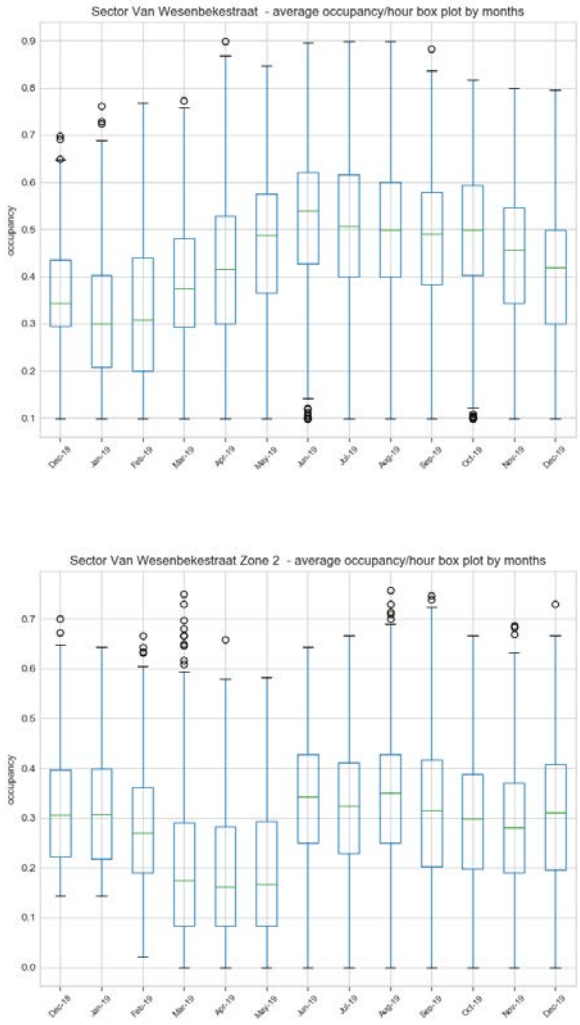


Figure 3.17: Box plots for sectors on the street Van Wesenbekestraat. From top to bottom: Zone 1 and Zone 2.

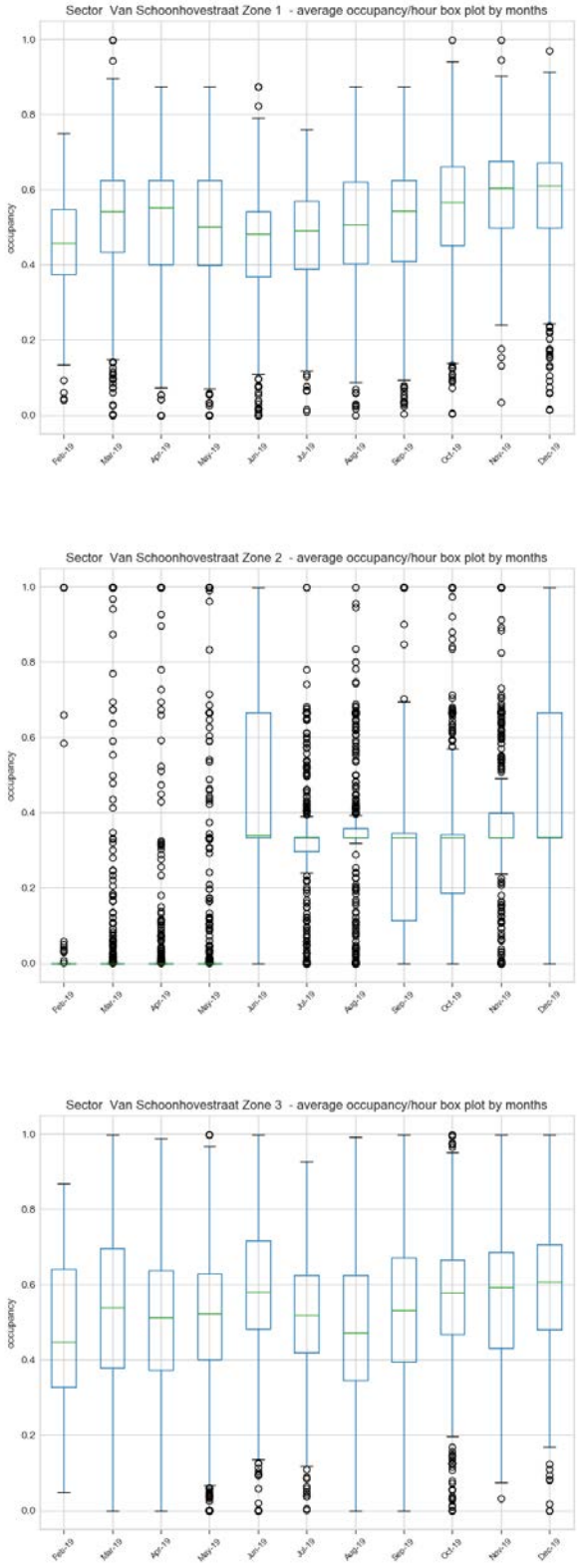


Figure 3.18: Van Schoonhovestraat box plots. From top to bottom: Zone 1, Zone 2 and Zone 3.

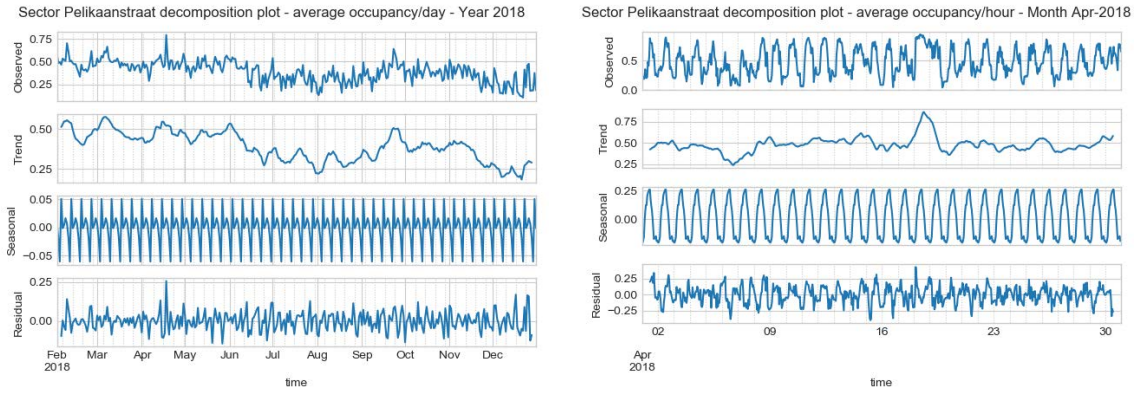


Figure 3.19: Decomposition of Pelikaanstraat time series. Left: the decomposition of the daily averaged series for the year 2018. Right: the hourly series for April 2018.

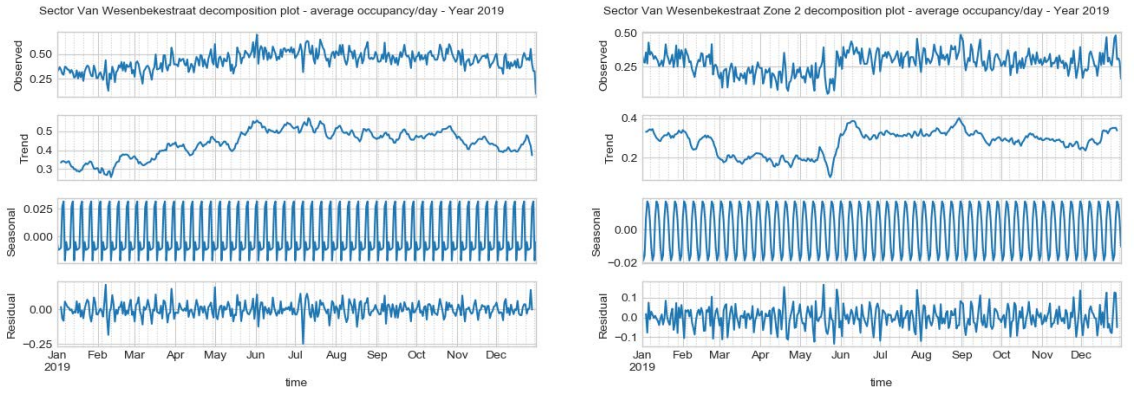


Figure 3.20: Decomposition of sectors on the street Van Wesenbekestraat. Occupancy/day for Zone 1 (left) and Zone 2 (right)..

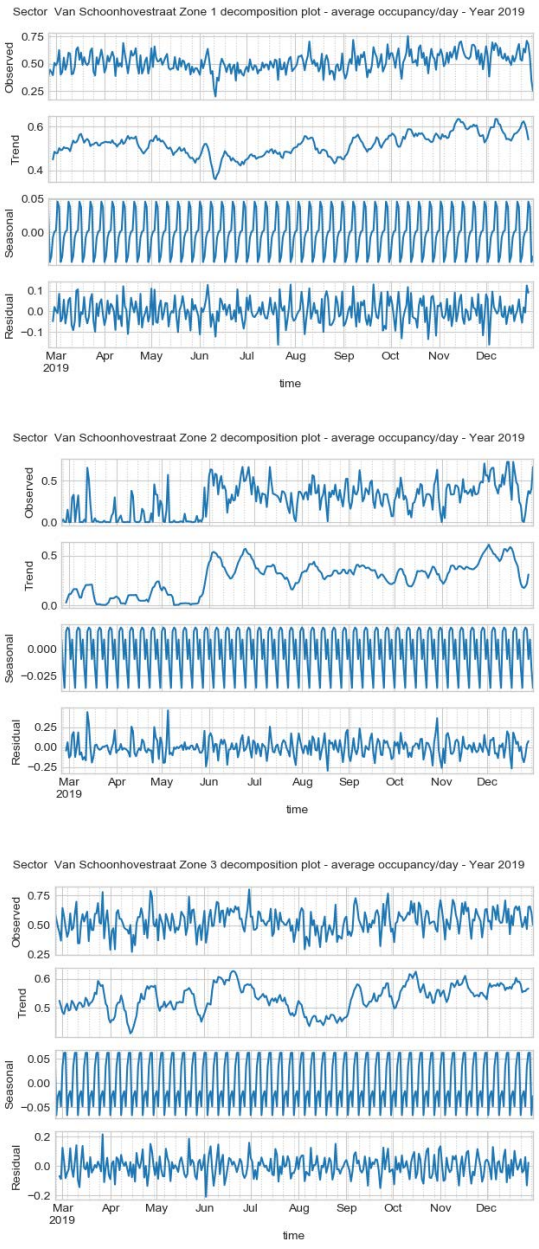


Figure 3.21: Decompositional plot for sectors on Van Schoonhovestraat. From top to bottom: Zone 1, Zone 2 and Zone 3.

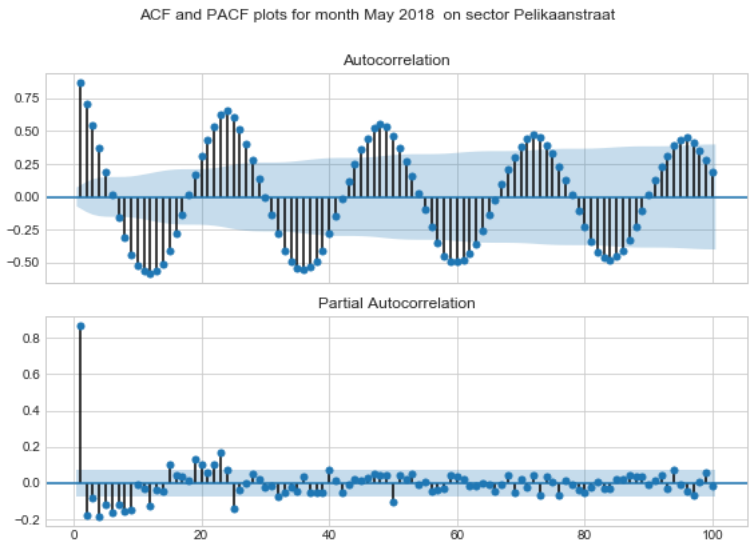


Figure 3.22: ACF and PACF plots for Pelikaanstraat, May 2018.

Figure 3.22 shows the ACF and PACF plot of the Pelikaanstraat sector for May 2018. The series indicates strong autoregressive behaviour as the autocorrelation function tails off and the partial autocorrelation cuts off at a significant lag. The most significant lag is the first one, but the subsequent lags up to lag 25 also have some significance. As the number of lags increases, the levels of the functions decreases, as expected; but lag 50 becomes significant again. This can be considered indicative of daily seasonality.

The ACF and PACF plots for the Van Wesenbekestraat sectors show that the significant lags for Zone 1 are 1 and 25, which is similar to the Pelikaanstraat sector. Zone 2, on the other hand, shows that only the first lag is significant. In both cases, the autocorrelation function tails off. The processes in both sectors are type AR(1) (autoregressive with 1 significant lag).

Lastly, for the sectors on Van Schoonhovestraat, Zones 1 and 3 show AR(1) behaviour without seasonality, while the first and second lags in Zone 2 are significant. It is not clear if there is daily seasonality in Zone 2, as lag 49 is also significant.

A lag plot is a scatter plot for realizations in the time series occurring between time t and time $t-1$. Figure 3.25 shows an example of this visualization for up to 9 lagged values. The scatter plot has a linear meaning for the first visualization in the figure; but as the considered lag increases, the linear relationship disappears.

The correlation between the time series and lagged series is shown in Figure 3.26, where a linear relationship can be seen for all sectors except for Van Schoonhovestraat Zone 2. This is due to the low number of sensors in the sector, which also plays a role in the high variance of the sector (compared with the other sectors).

Figure 3.27 presents the estimated spectrum of the occupancy/hour time series for the sector on Pelikaanstraat. The dominant period is 24, thus indicating daily seasonality.

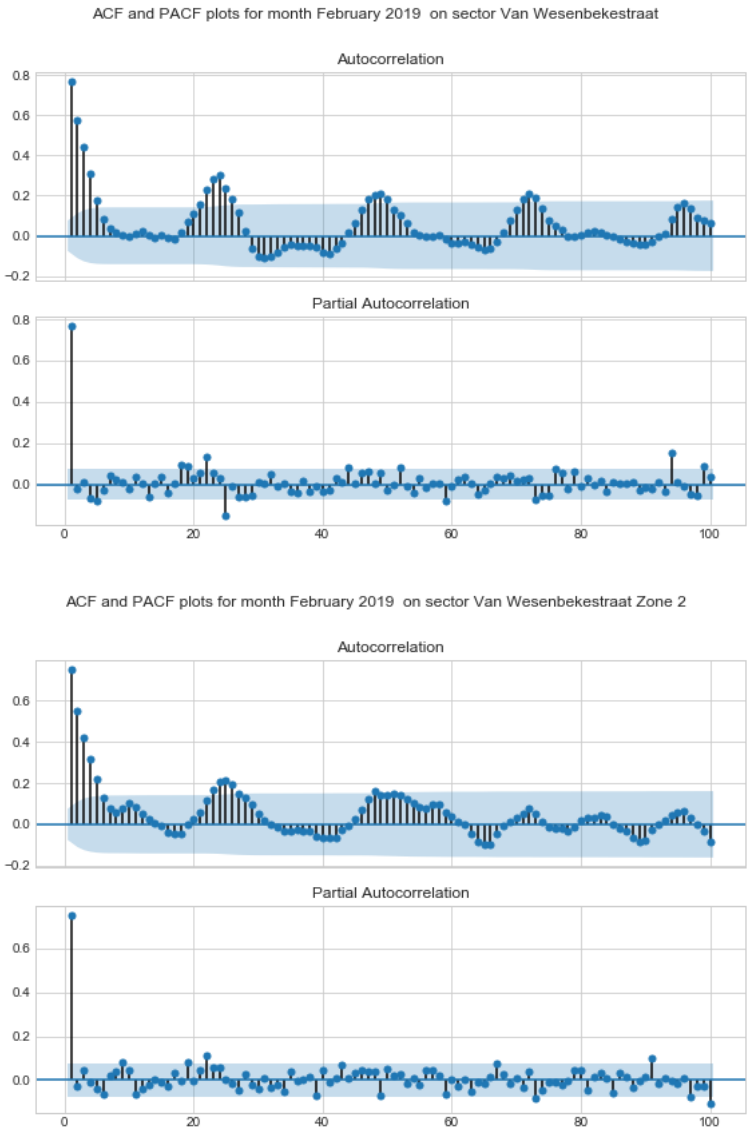


Figure 3.23: ACF and PACF plots for the Van Wesenbekestraat sectors.

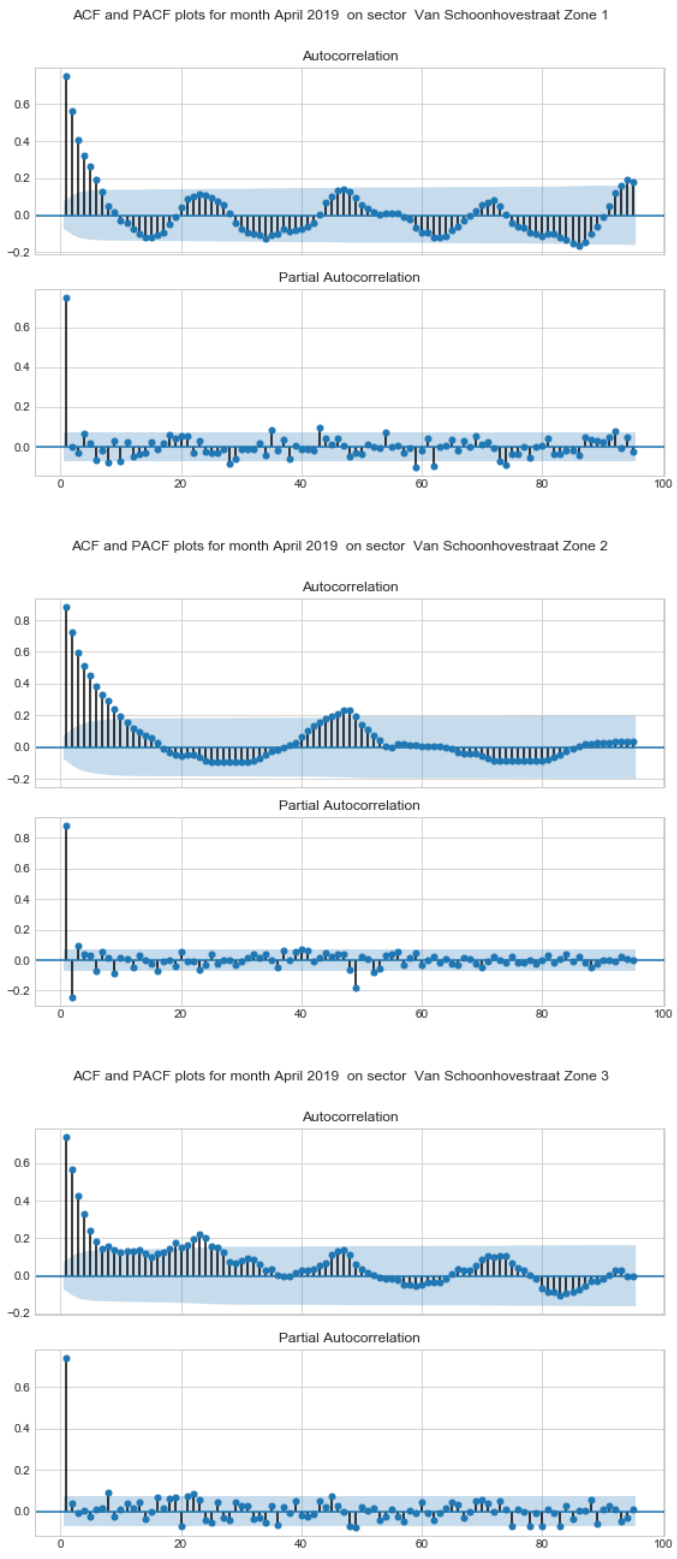


Figure 3.24: ACF and PACF for sectors on Van Schoonhovestraat.

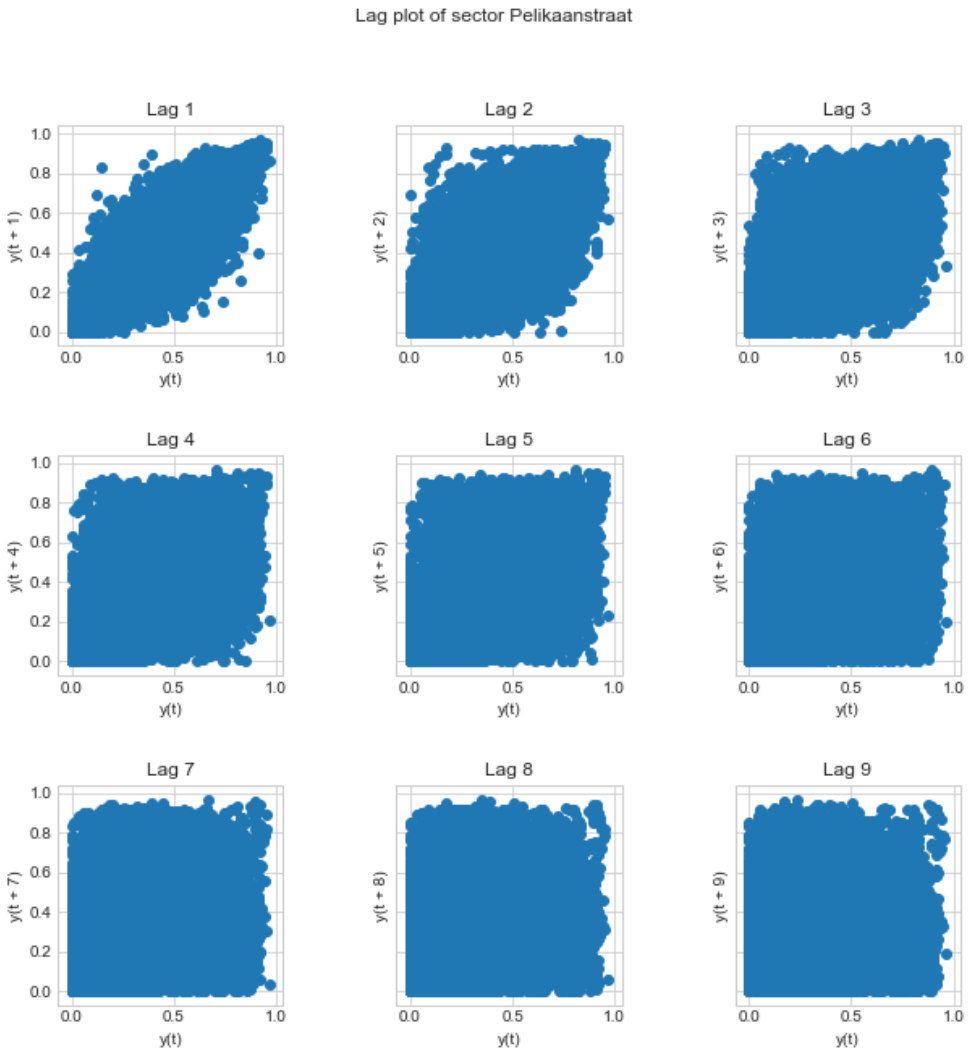


Figure 3.25: Lag plots for the Pelikaanstraat sector.

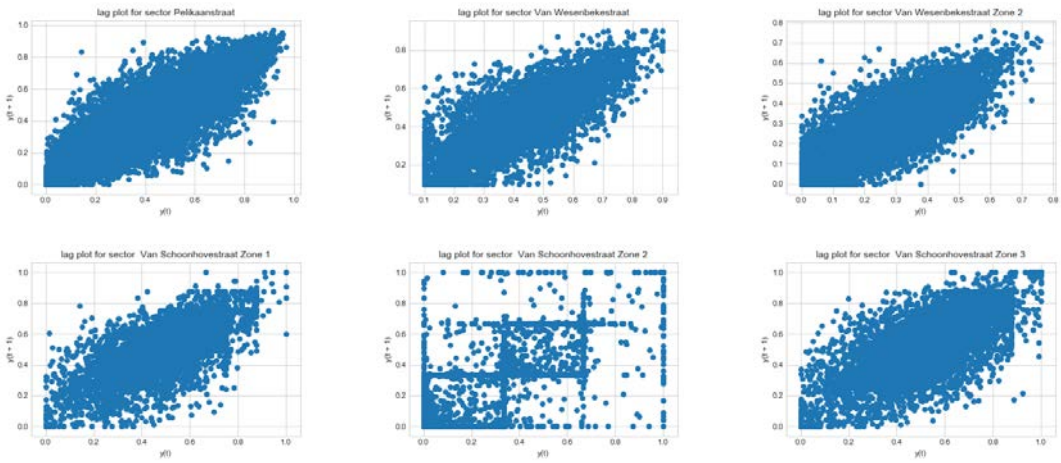


Figure 3.26: Lag plots for Antwerp sectors. First row from left to right: Pelikaanstraat, Van Wesenbekestraat Zone 1 and Van Wesenbekestraat Zone 2. Second row from left to right: Van Schoonhovestraat Zones 1, 2 and 3.

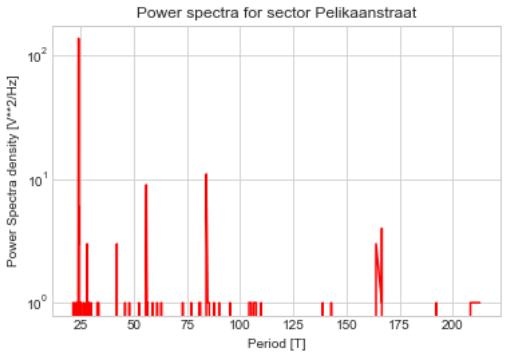


Figure 3.27: Pelikaanstraat occupancy/hour in the frequency domain.

Other interesting period values are 84, 56 and 168. The last one coincides with weekly seasonality.

The spectrum for sectors on Van Wesenbekestraat is similar to Pelikaanstraat, as the dominant period is 24. Other periods of interest can be seen in Zone 1, where they are 48 and 168 (weekly seasonality).

For sectors on Van Schoonhovestraat, the spectrum reveals that while Zones 1 and 3 have similar periods, such as 24 and 168, Zone 2 behaves erratically with no dominant period.

Time-space analysis

Figure 3.30 shows the cross-correlation for the Pelikaanstraat sector having few correlations with other sectors.

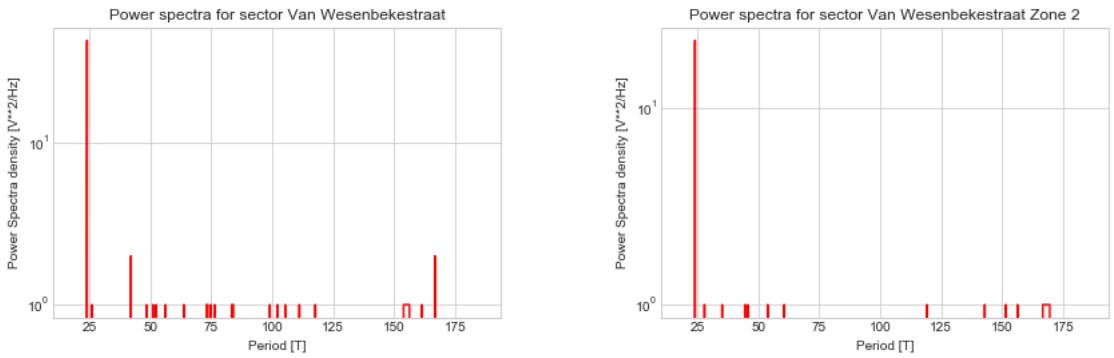


Figure 3.28: Van Wesenbekestraat occupancy/hour in the frequency domain.

The cross-correlation plot for the Van Wesenbekestraat sectors (Figure 3.31) shows high correlations between them and some correlation with Van Schoonhovestraat's Zones 1 and 3 sectors.

Lastly, the sectors on Van Schoonhovestraat show correlation (Figure 3.5) with the Van Wesenbekestraat sectors, especially at closer time lags. But they have no correlations with the Van Schoonhovestraat Zone 2 sector.

3.5.2 Barcelona (Spain)

Characteristics

Barcelona was the first scenario where Worldsensing installed their sensors, as it was easier for them to check and control the behaviour due to their being located in the same city. Table 3.6 shows the city sector and number of sensors. Later, a new sector was added next to the existing one, but it is not considered in this study, as it consists of a sector with a single parking spot for handicapped people. The reason for not including it is because the behaviour of very small sectors is already known (e.g., the Antwerp Van Schoonhovestraat Zone 2 sector, Figure 3.4).

The main characteristics of this sector is that the parking places belong to one Barcelona university and it is commonly used for short stays by supply transport.

Analysis of all available data for the entire period

When visualizing the plot for all the historical occupancy/hour data (Figure 3.33), a mean value of 0.295 is shown, with no irregular/anomalous realizations. Yet, somehow, it has a constant periodic pattern.

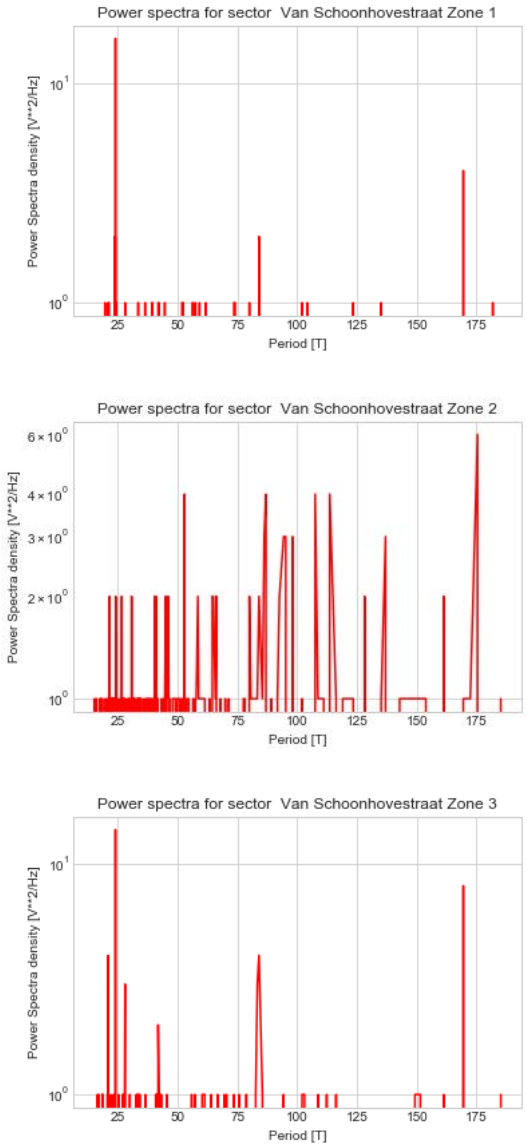


Figure 3.29: Van Schoonhovestraat occupancy/hour in the frequency domain.

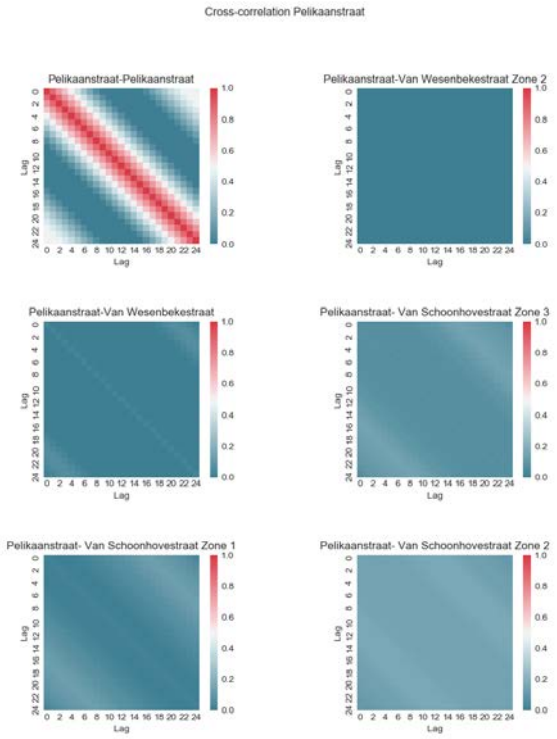


Figure 3.30: Cross-correlations of the Pelikaanstraat sector.

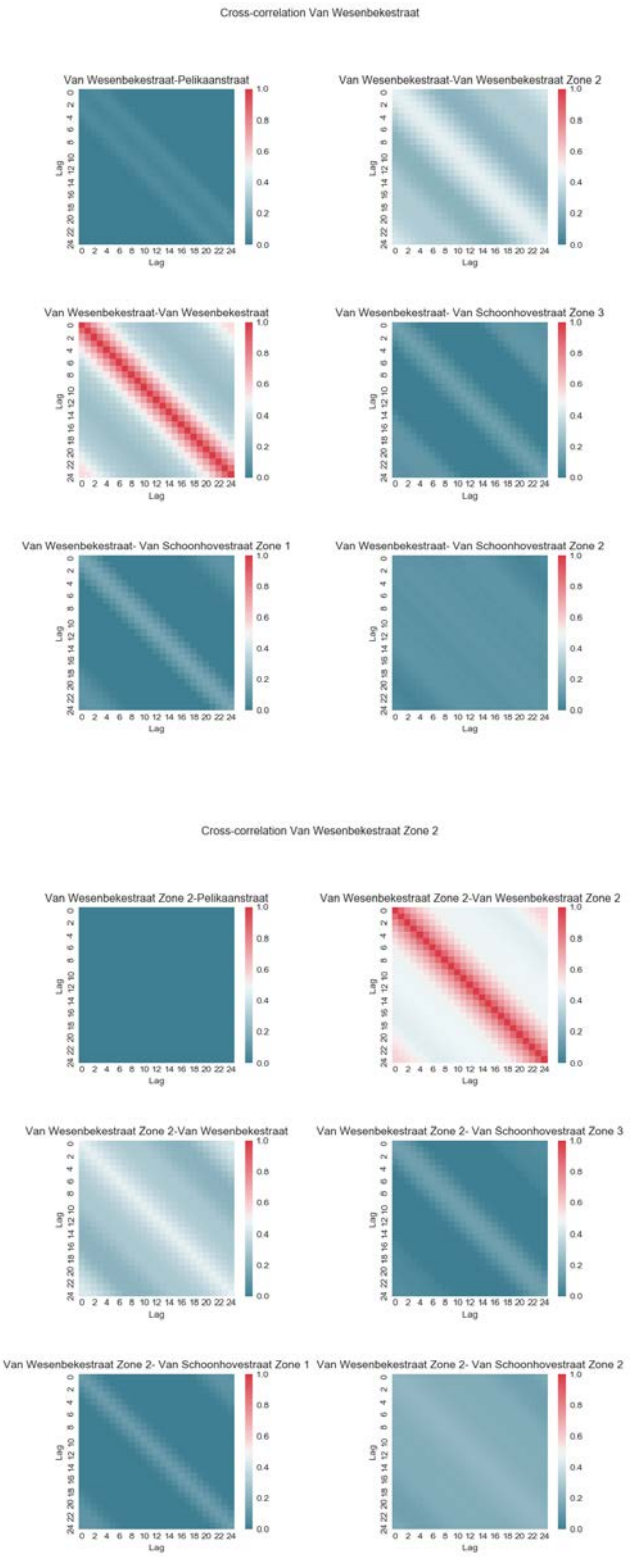


Figure 3.31: Cross-correlations of sectors on Van Wesenbekestraat. From top to bottom: Zone 1 and Zone 2.

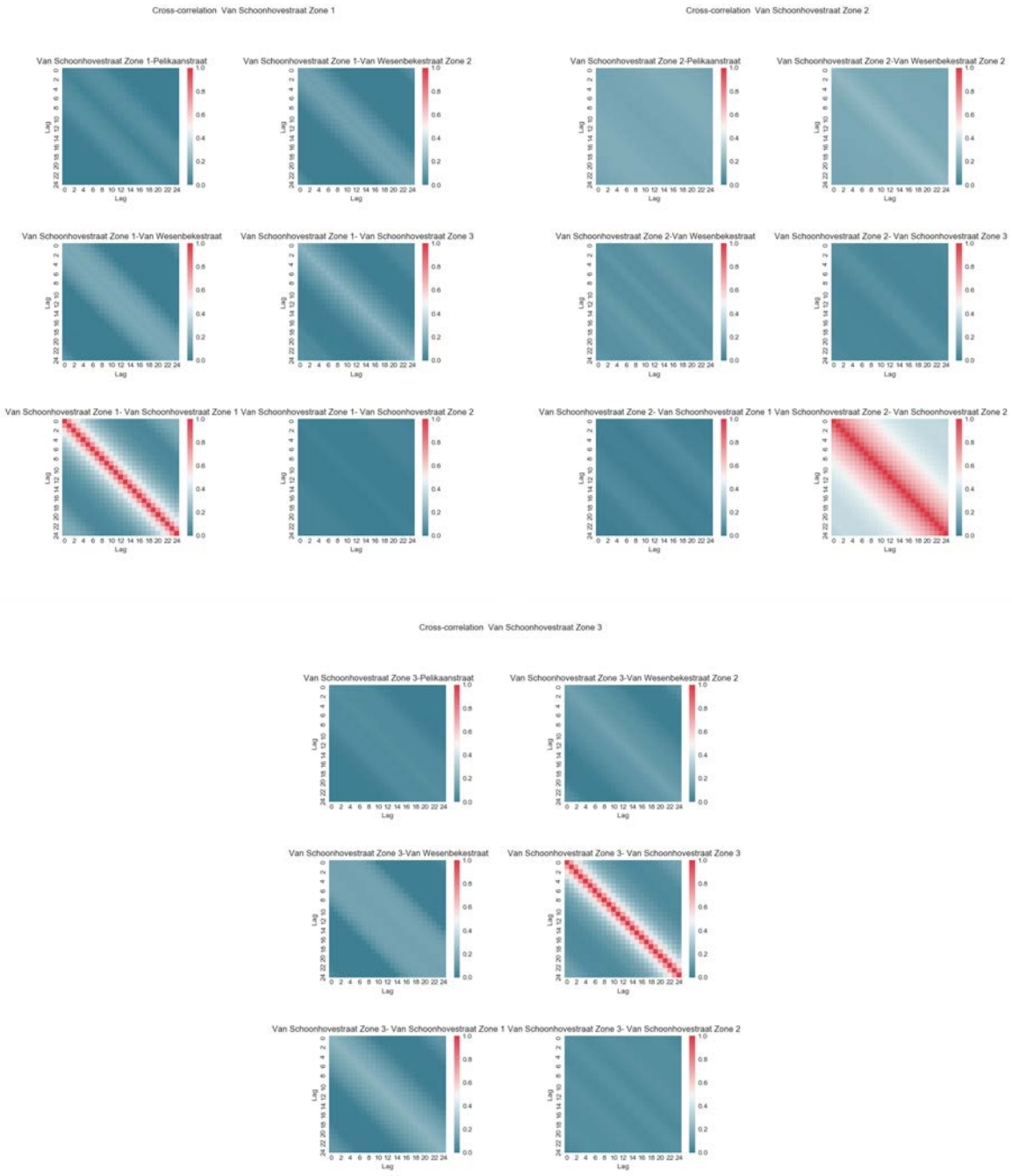


Table 3.5: Cross-correlations of Van Schoonhovestraat. From top (left, right) to bottom: Zone 1, Zone 2 and Zone 3.

Sector name	*Number of sensors*	*Deployment date*
Regular	11	08/01/2018

Table 3.6: The Barcelona sector at the Universitat Politècnica de Catalunya, Campus Nord.

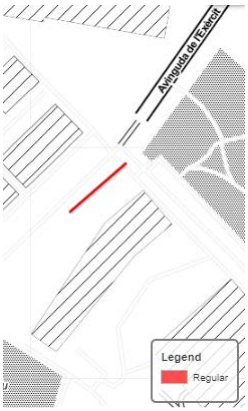


Figure 3.32: Barcelona map. The city sensors for Barcelona are installed in a private location.

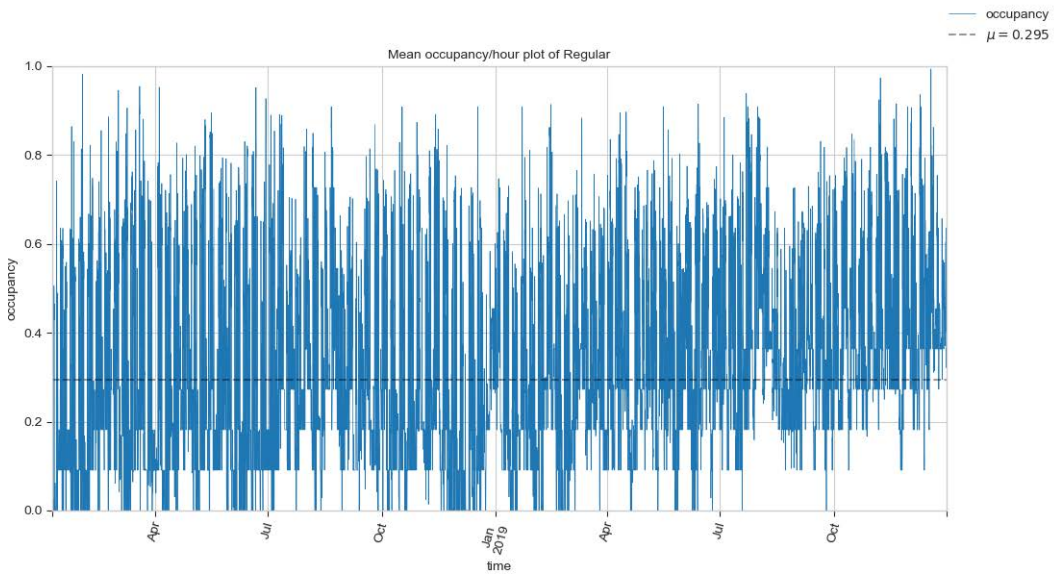


Figure 3.33: Regular sector occupancy/hour data for 2018 and 2019.

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0.125
Median	0.273
Q3	0.434
95-th percentile	0.7
Maximum	1
Range	1
Interquartile range (IQR)	0.31

Descriptive statistics	
Standard deviation	0.207
Coefficient of variation	0.702
Kurtosis	-0.254
Mean	0.295
Median Absolute Deviation	0.17
Skewness	0.657
Variance	0.043

Table 3.7: Statistics for the Pelikaanstraat sector.

Statistics

The statistics for the Barcelona sector are presented in Table 3.7. The mean occupancy is low and the variance is quite high. Similarly to the Pelikaanstraat sector (Table 3.2), it has a high positive skewness. These characteristics are a common pattern for short stay parking places. From the density plots (Figure 3.34), we can observe two peaks similar to those in the Pelikaanstraat sector, which are indicativeof two different behaviours. In this case, we suspect that the reason for this is the sector is nearly empty at night and on weekends and holidays, due to university regulations.

Monthly and weekly data analysis

Shorter time intervals, in the sense of calendar periods like months or weeks, help us interpret the data better. Figure 3.35 shows a daily pattern in the realizations throughout May 2018, as occupancy increases and decreases in a similar fashion throughout the days. We can also clearly see that the occupancy levels on weekends remain low while the university is closed. From the second week of May plot, we can see a pattern on some days as occupancy peaks two times: once at 6 a.m. to 7 a.m., and again at nearly 12 p.m.

Daily patterns can be seen more clearly when smoothing the series. Figure 3.36 illustrates the occupancy series of the second week of May 2018. A moving average with a window of 12 hours smooths the series, thus making the daily patterns very clear.

From the seasonal plots in Figure 3.37 it is easy to see, first, that the sector is governed by daily patterns that behave differently on weekdays and weekends and, second, that the overall patterns are maintained over time.

The seasonal plot for the average occupancy/hour by day (top left image in Figure 3.37) shows that weekdays have very similar patterns that differ from weekends, when the occupancy is much lower.

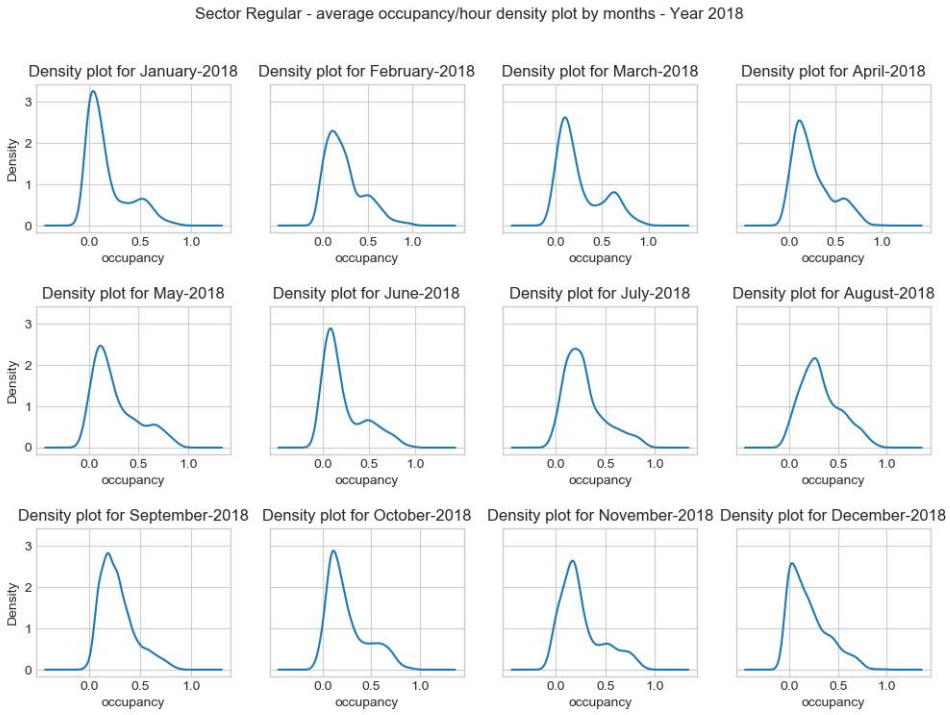


Figure 3.34: Density plot for the year 2018, Pelikaanstraat sector.

The top right image represents the average occupancy/hour by month, and all months behave very similarly.

The bottom left plot shows the average occupancy/hour by week, where we can see that all weeks are similar in shape.

Lastly, the plot on the bottom right shows that the yearly behaviour is similar for 2018 and 2019, although a slight increase in the occupancy/hour happened in 2019.

Specialized time series plots

Looking at the box plot in Figure 3.38, we see that the mean levels increased a little in the last part of 2019; but, overall, the mean and variance levels always maintain the same values over time.

In Figure 3.39 we can see the trend, seasonality and residuals of the data for the years 2018 and 2019, as well as for May in both years. The trend values do not increase or decrease in 2018, but they increase for the year 2019. Within the monthly period, independently of the year, trend and seasonality maintain the same values. Seasonal patterns stay the same for both years.

Figure 3.40 shows the ACF and PACF plots for the Regular sector in Barcelona for May 2018. Similarly to the Antwerp ACF-PACF plots (Figure 3.22), the plots show a strong

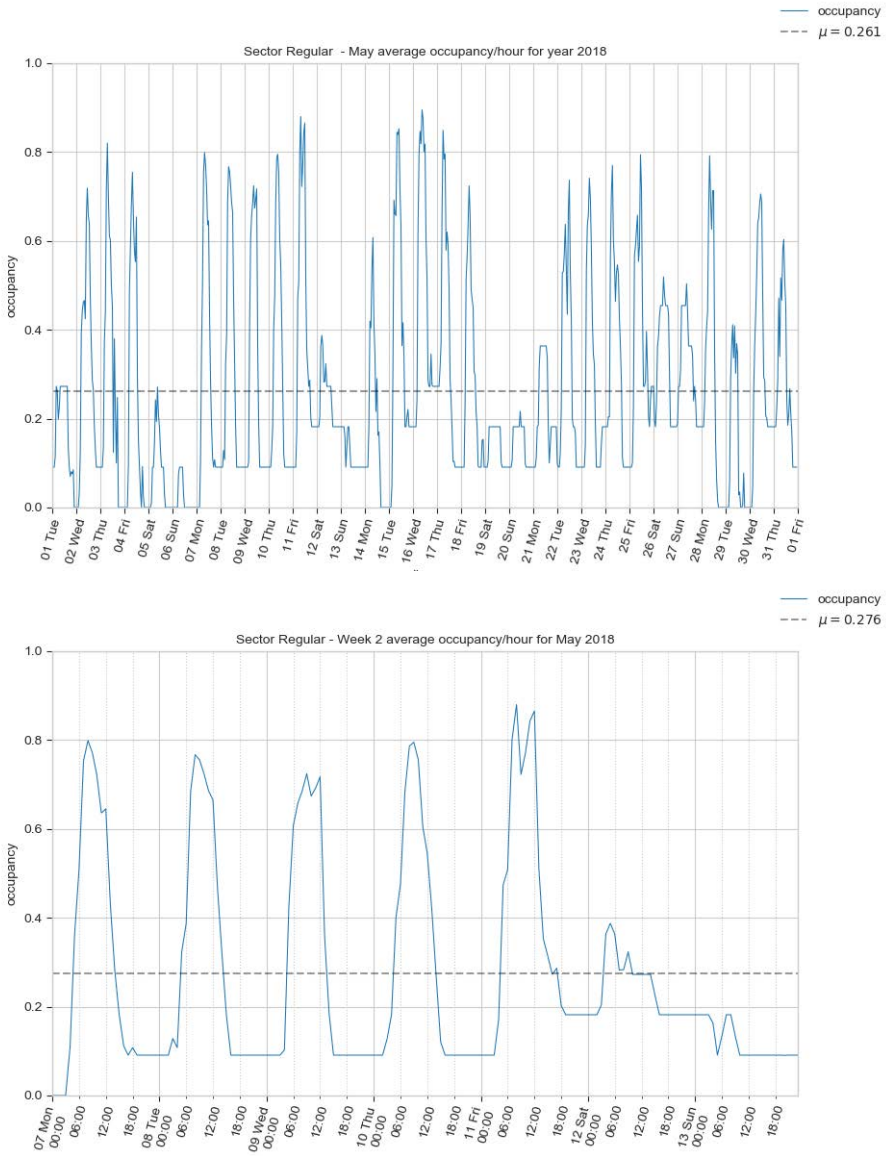


Figure 3.35: Barcelona time series for February 2018 and second week of May 2018.

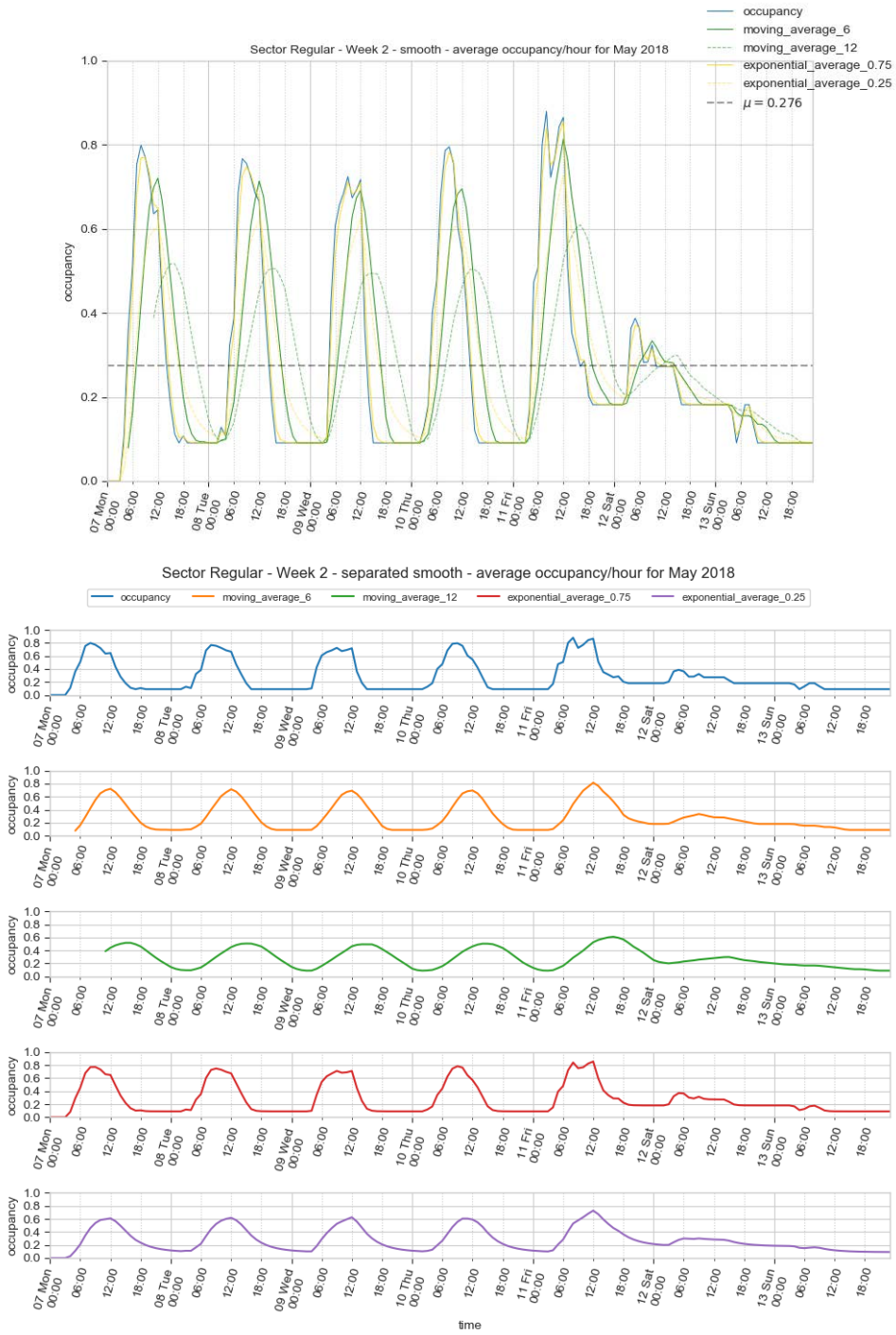


Figure 3.36: Smoothed time series for the Barcelona Regular sector for May 2018. The upper image shows the series layered one on top of the other, where green lines represent the window moving average series and the yellow ones the exponential weighted average results. The bottom figure decomposes the top plot and shows the smoothness of the series under the different methods.

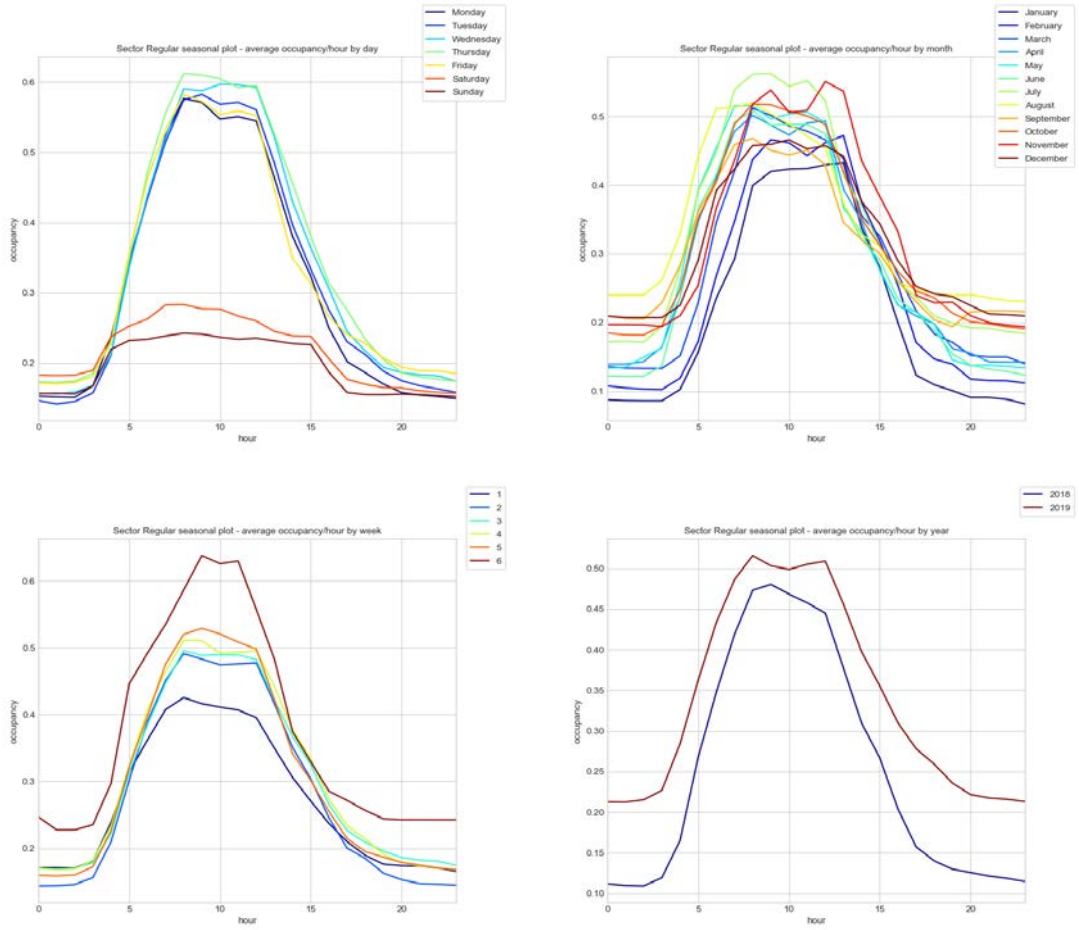


Figure 3.37: Seasonal plots for the Regular sector in Barcelona. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

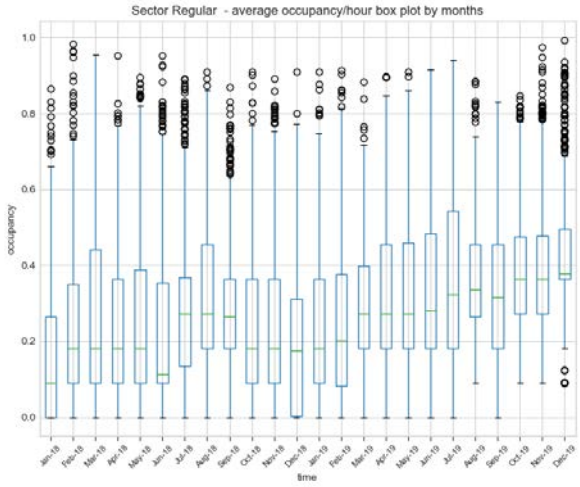


Figure 3.38: Box plot for the Regular sector in Barcelona.

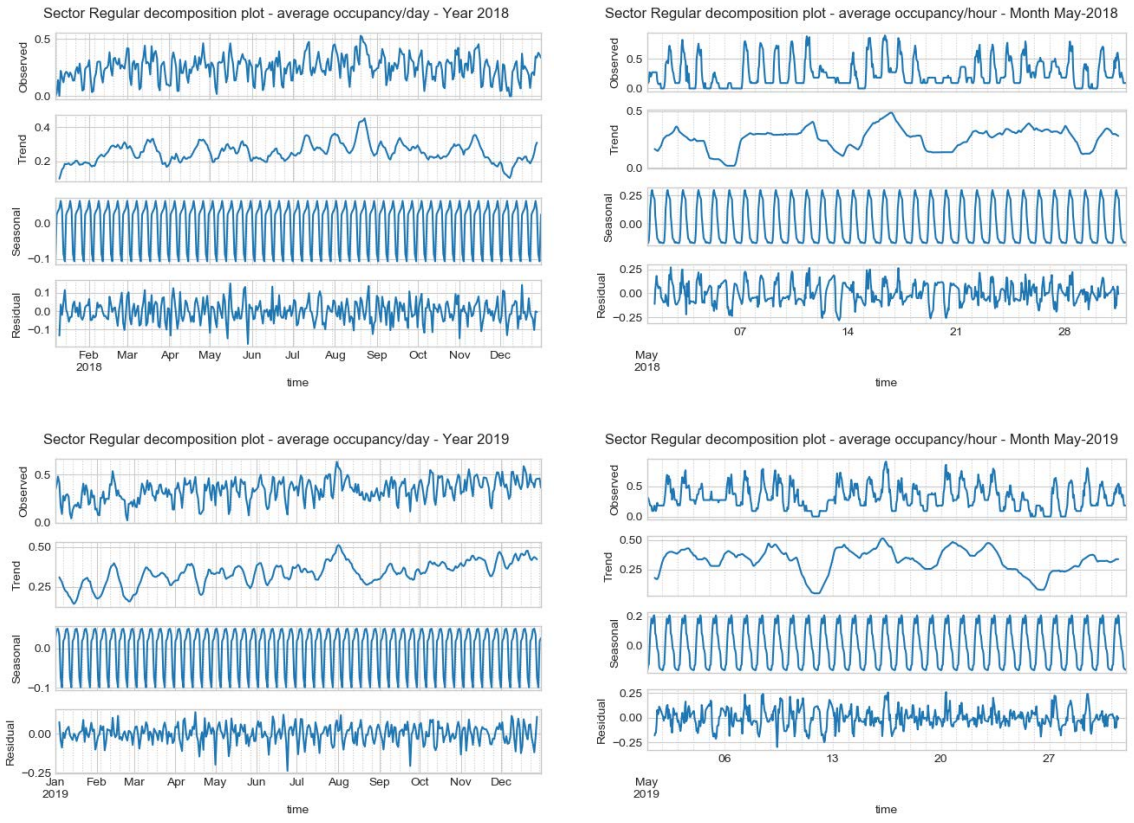


Figure 3.39: Decomposition of the time series for the Regular sector in Barcelona. Top images are for the year 2018 and May of that same year. Bottom images are for the year 2019 and May of that same year. The trend increases between years.

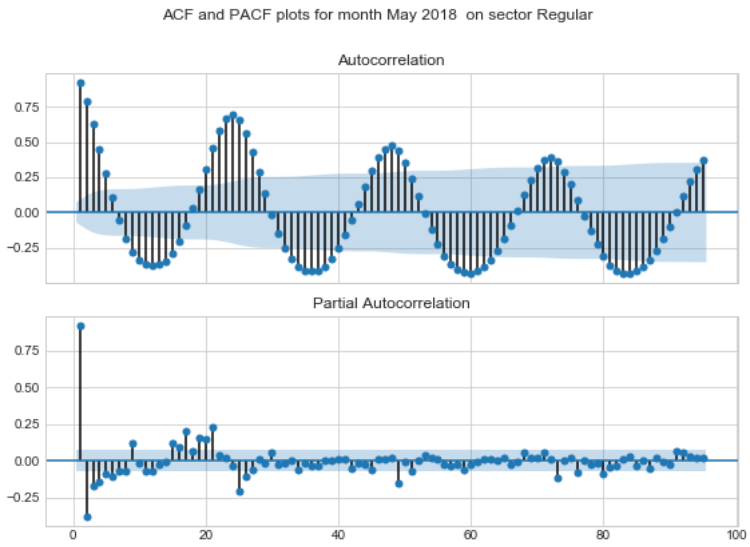


Figure 3.40: ACF and PACF plots for Barcelona sector, May 2018.

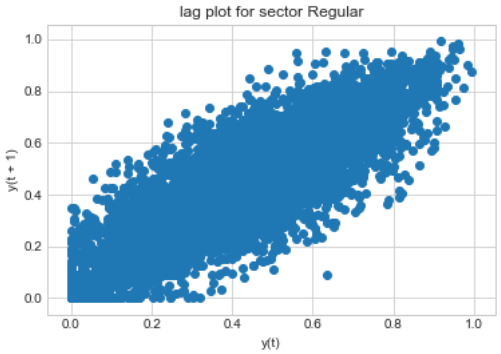


Figure 3.41: Lag plot of Barcelona Regular sector.

autoregressive process. The first two lags are the most significant for this sector, then every 25 hours there is a significant lag, thus indicating daily seasonality. The lag plot in Figure 3.41 shows a strong linear correlation in the series between lag t and lag $t-1$.

Checking the power spectrum of the Barcelona sector, the predominant periods are marked at the values 24 and 168, coinciding with daily and weekly seasonality.

Spatial relationships

Because there is only one sector in the Barcelona scenario, it does not make sense to check for cross-correlations in the series.

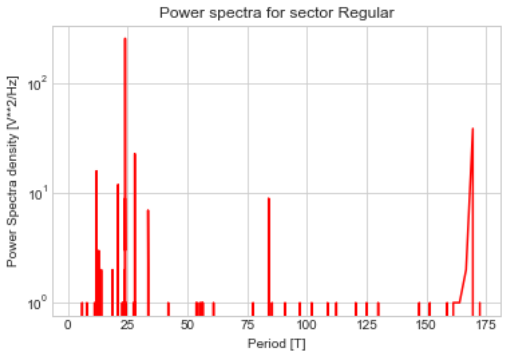


Figure 3.42: Barcelona occupancy/hour in the frequency domain.

Sector name	*Number of sensors*	*Deployment date*
Inside	22	02/03/2018
Outside	18	02/03/2018

Table 3.8: Wattens sectors at Swarco facilities.

3.5.3 Wattens (Austria)

Characteristics The third installation is in the city of Wattens. Table 3.8 shows the sectors in this scenario and the numbers of sensors in each of them.

Both sectors (see Figure 3.43) are part of the Swarco company's private parking. As the names denote, the Inside sector is inside the building area and the Outside sector is outside.

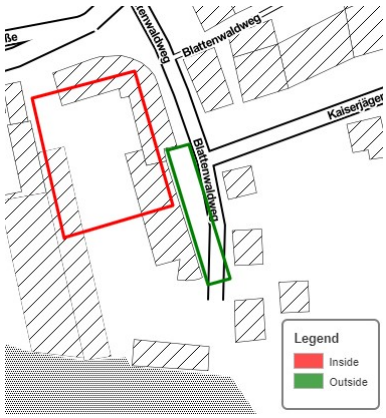


Figure 3.43: Wattens map. Sectors are located around the private company Swarco. The red square is the Inside sector and the green square is the Outside sector

Quantile statistics	
Minimum	0
5-th percentile	0.043
Q1	0.087
Median	0.174
Q3	0.359
95-th percentile	0.668
Maximum	0.858
Range	0.858
Interquartile range (IQR)	0.272

Descriptive statistics	
Standard deviation	0.206
Coefficient of variation	0.83
Kurtosis	-0.224
Mean	0.249
Median Absolute Deviation	0.170
Skewness	1.060
Variance	0.043

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0.0556
Median	0.111
Q3	0.341
95-th percentile	0.805
Maximum	1
Range	1
Interquartile range (IQR)	0.286

Descriptive statistics	
Standard deviation	0.260
Coefficient of variation	1.111
Kurtosis	0.356
Mean	0.234
Median Absolute Deviation	0.210
Skewness	1.282
Variance	0.068

Table 3.9: Statistics for the Wattens sectors Inside (top) and Outside (bottom).

Analysis of all available data for the entire period

The Wattens sectors were removed at the start of June 2019, as can be seen in Figure 3.44, which shows no more data from June 2019 onwards. Both sectors have low occupancy overall.

Statistics

Looking at Table 3.9, both sectors present low mean values and high variance (similar statistics to the Barcelona sector, check Table 3.7 for more information). The density plots (Figures 3.45) show the typical two peaks of the sectors with very different behaviour (day time vs. night time or holidays vs. non-holidays). The explanation is that these sectors are used for workers and are thus used mainly during office hours (from 6 a.m. to 6 p.m.).

Monthly and weekly data analysis

When looking at the occupancy/hour using a shorter time range, it is easier to visualize clear daily patterns. Both Figures (3.46 and 3.47) show how the occupancy/hour rises at

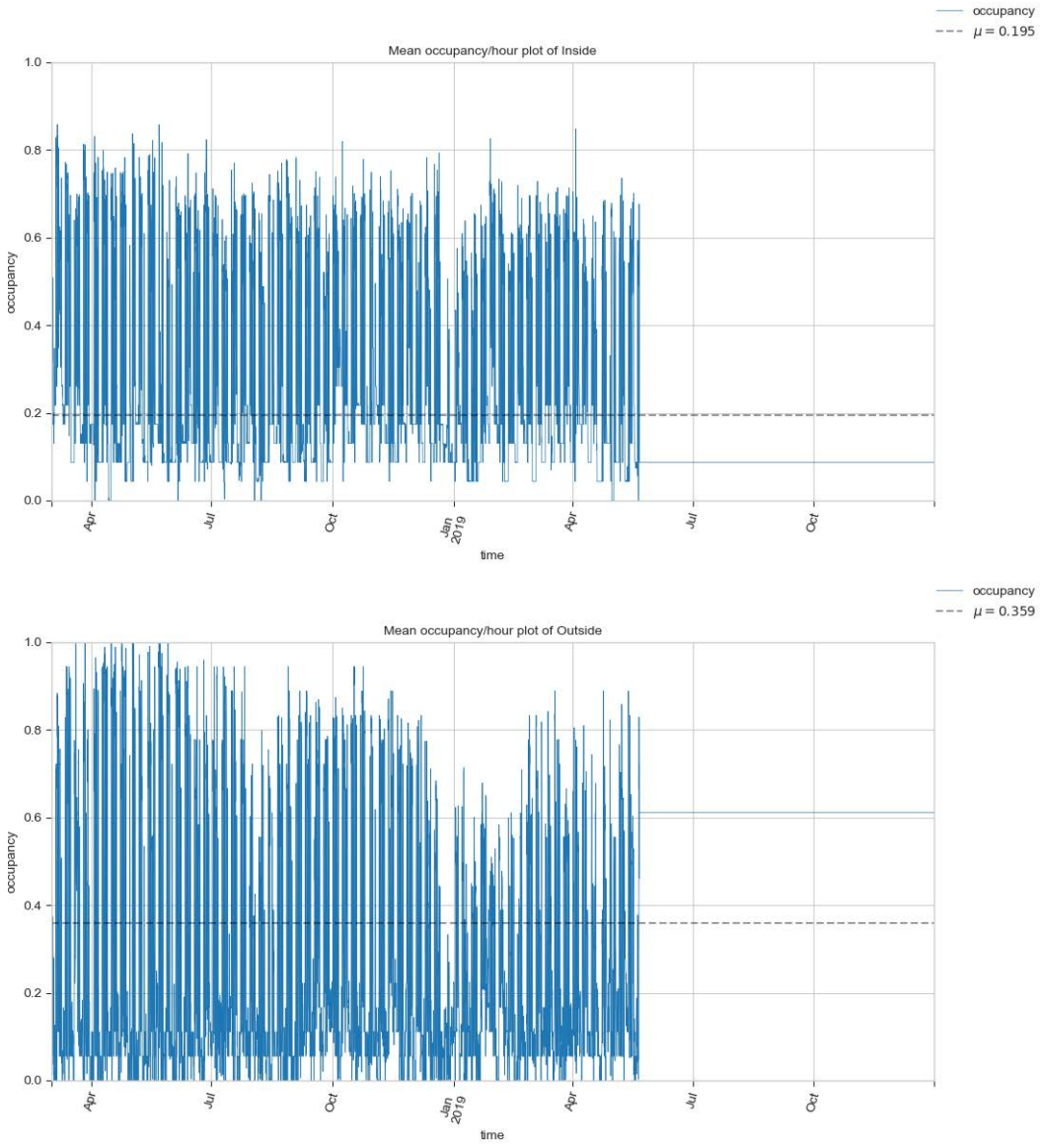


Figure 3.44: Wattens sectors. Data for the interval 2018 and 2019.

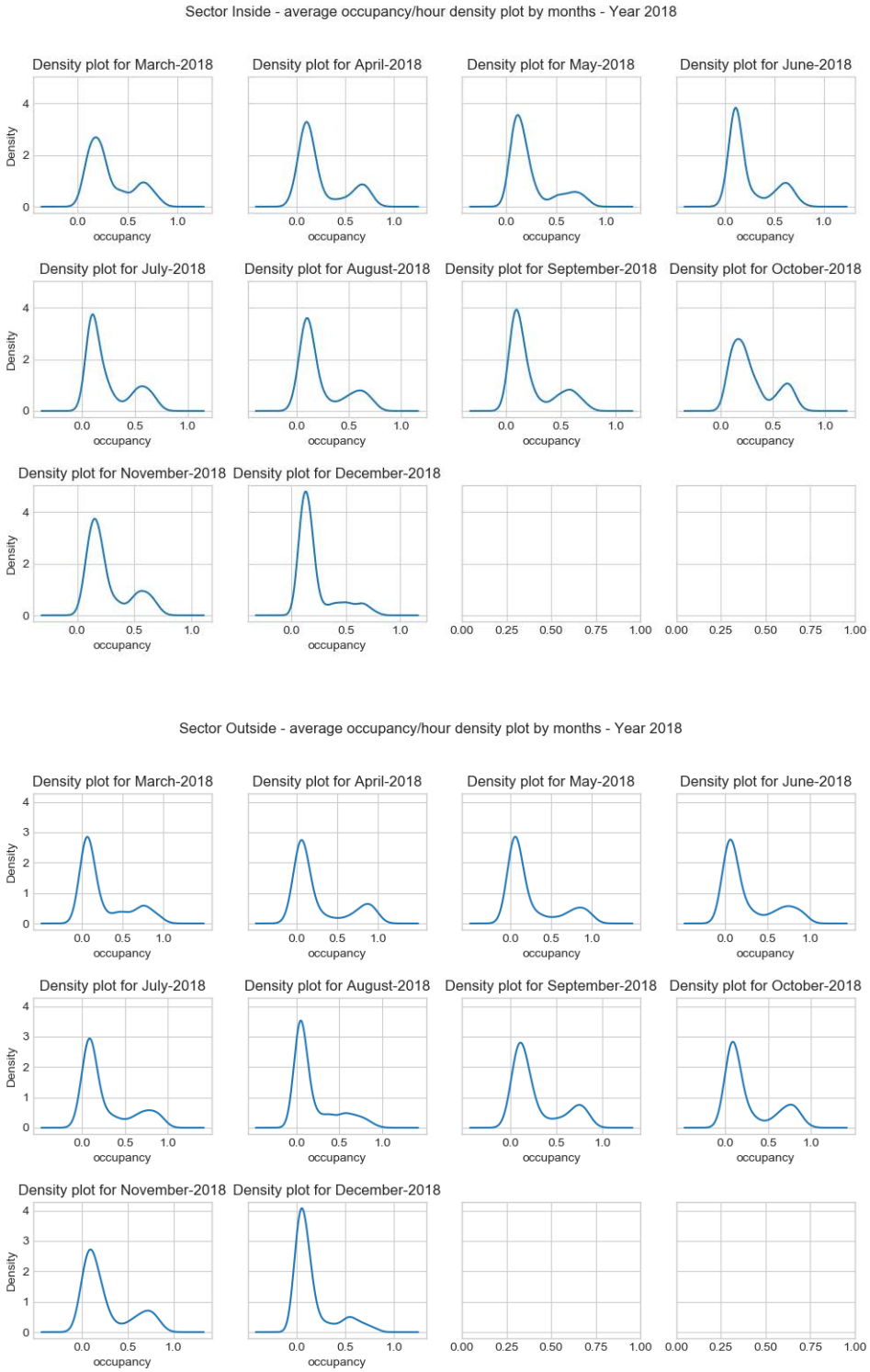


Figure 3.45: Density plots for the sectors Inside (top) and Outside (bottom) for the year 2018.

6 a.m. and lowers at 18 p.m. A quick search on Google finds that the Swarco company office hours are from 8 a.m. to 5 p.m., Monday to Thursday, and from 8 a.m. to 12 p.m. on Friday, then closed on weekends. This explains these parking occupancy patterns.

Plots of the smoothed series (Figure 3.48) for the Outside sector better reveal the daily occupancy patterns and the nearly empty occupancy for weekends.

The daily patterns are perfectly characterized in the seasonal plots (Figure 3.49), where the occupancy/hour perfectly distinguishes the day of the week. Thus, the Friday behaviour is due to the office hours, as the office closes at 12 p.m. on that day. For both Saturday and Sunday, the office is closed and thus the parking sectors are empty. Workday office hours can be seen clearly: 8 a.m. to 17 p.m.

From the plot of averaged occupancy/hour by months, the lower values for December can be explained by the holidays, as the office was closed during the last two weeks of the month.

Lower occupancy/hour values during the first weeks may be explained by the distribution of holidays throughout the year. Lastly, the average occupancy/hour by year shows very similar patterns, with values between the two years.

Seasonal plots for the Outside sector (Figure 3.50) behave exactly like the Inside sector, as both are governed by the office opening hours. The only difference is the slightly higher occupancy levels.

Specialized time series plots

Box plots for both sectors (Figure 3.51) indicate constant mean and variance values over time. Curiously, the Outside sector seems to be more affected by the holidays, as its levels are lower in August and December compared to the Inside sector – especially for the month of August, which maintains its levels. Another interesting characteristic is that, in September, the occupancy/hour for the Inside sector is lower while the occupancy levels in the Outside sector return to being similar to previous months (June or July, for example). This could indicate that the parking sectors are used by different professional profiles.

Figure 3.52 shows the decomposition of the time series from Wattens. Observed values, trend, seasonality and residuals remain after subtracting trend and seasonality from the observations. One conclusions that can be drawn are that seasonality clearly follows daily periods in both sectors. Another can be made from the residuals: large values could indicate holidays during the first week of April, some weeks in May, the third week of August, and the first week of November. When decomposing occupancy/hour series for a month (September), both indicate very similar periodic patterns.

The ACF and PACF shown in Figure 3.53 present similar conclusions: an autoregressive process is evident where significant lags are 1, 2 and 25 (daily seasonality). Both lag plots (Figure 3.54) provide similar visualizations showing a strong linear correlation between t and $t-1$ values, especially when the occupancy levels become higher.

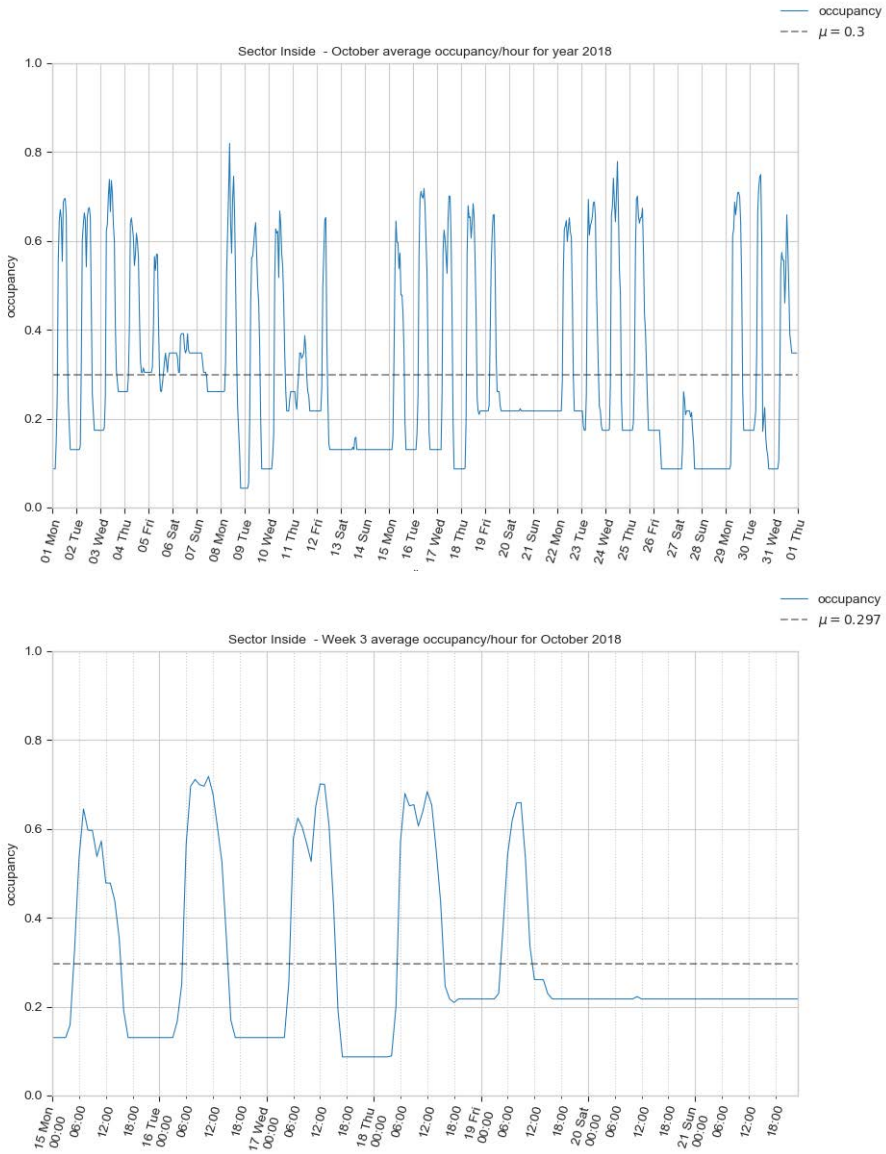


Figure 3.46: Inside sector time series for 2018 and third week of October 2018.

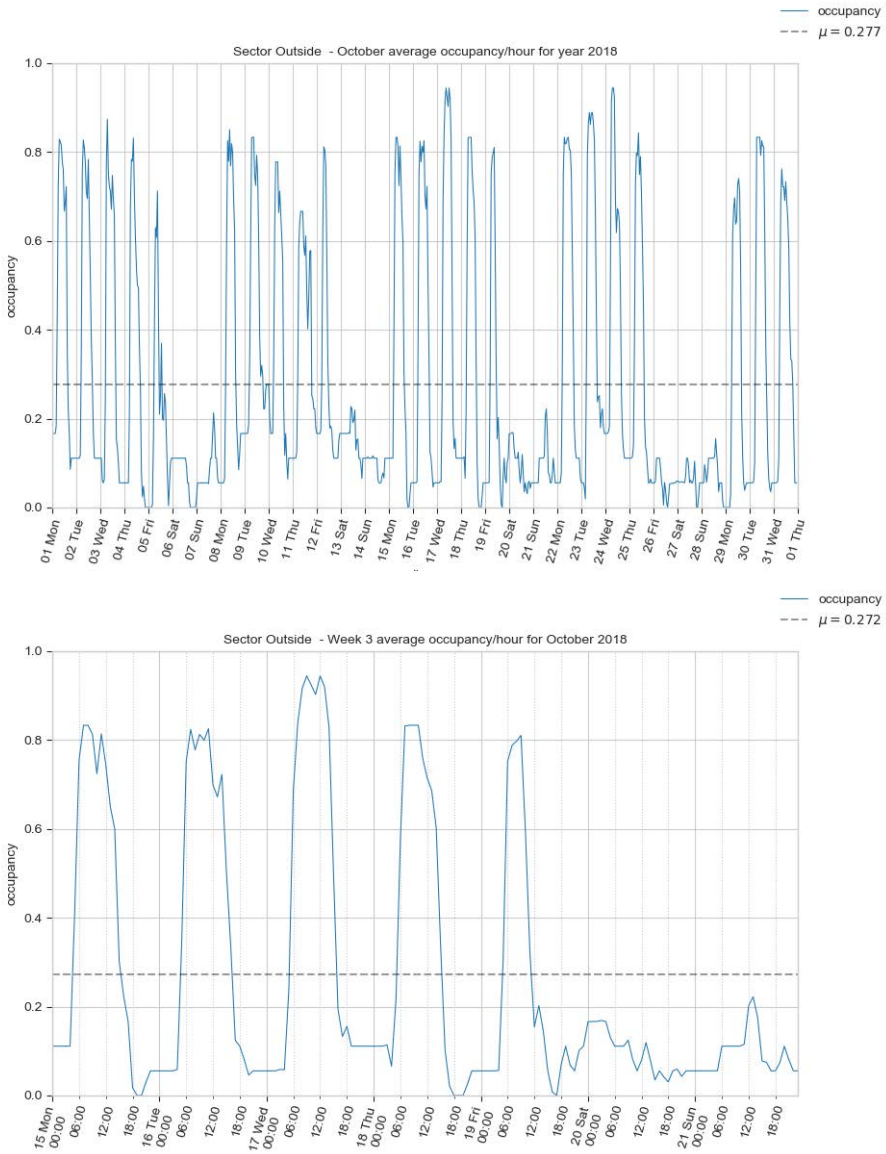


Figure 3.47: Outside sector time series for 2018 and third week of October 2018.

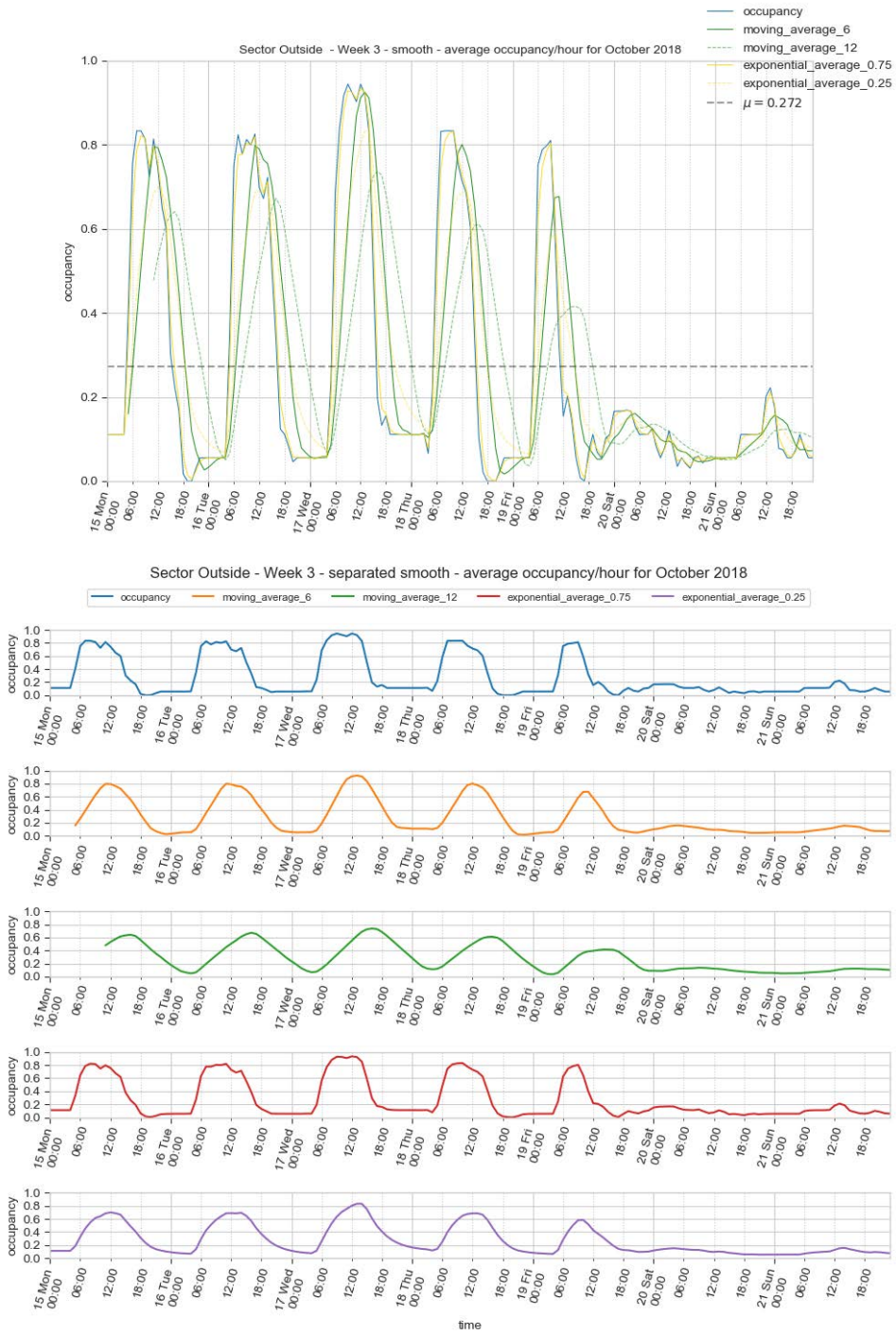


Figure 3.48: Smoothed time series for Outside sector in Wattens. The upper image shows the series layered one on top of the other, where green lines represent the window moving average series and the yellow ones the exponential weighted average results. The bottom figure decomposes the top plot and shows the smoothness of the series under different methods.

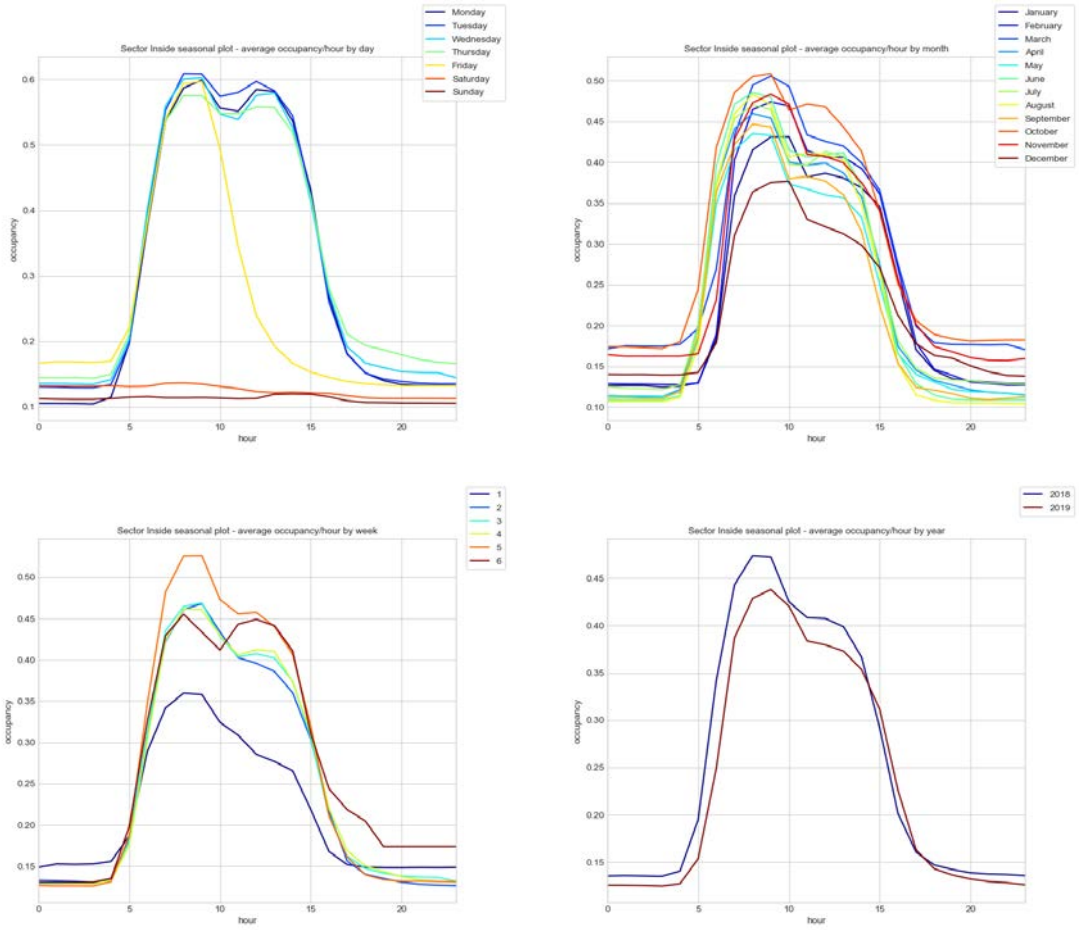


Figure 3.49: Seasonal plots for Inside sector in Wattens. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

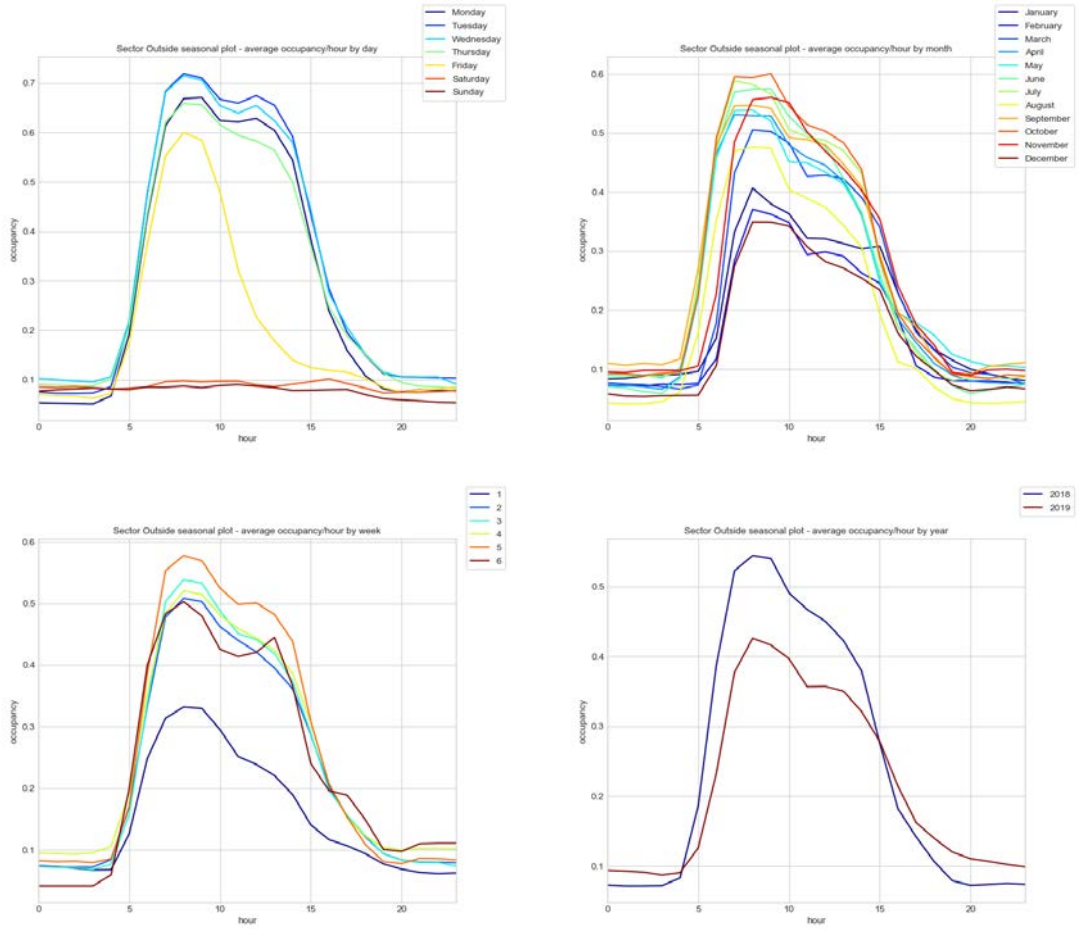


Figure 3.50: Seasonal plots for the Outside sector in Wattens. Hourly occupancy averaged at different periods. From top left to bottom right: day, month, week of the month and year.

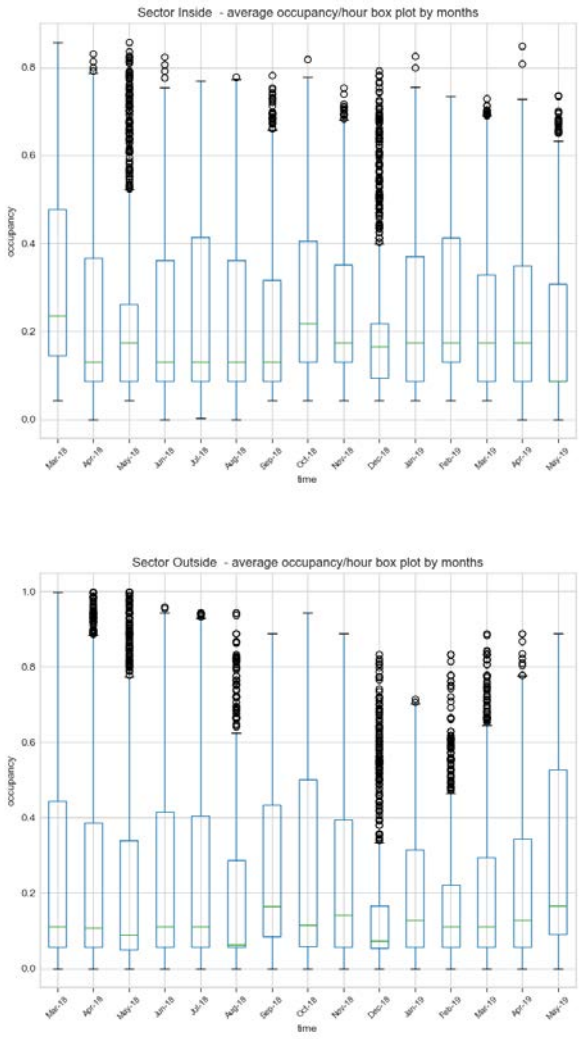


Figure 3.51: Box plots of Inside (top) and Outside (bottom) sectors in Wattens.

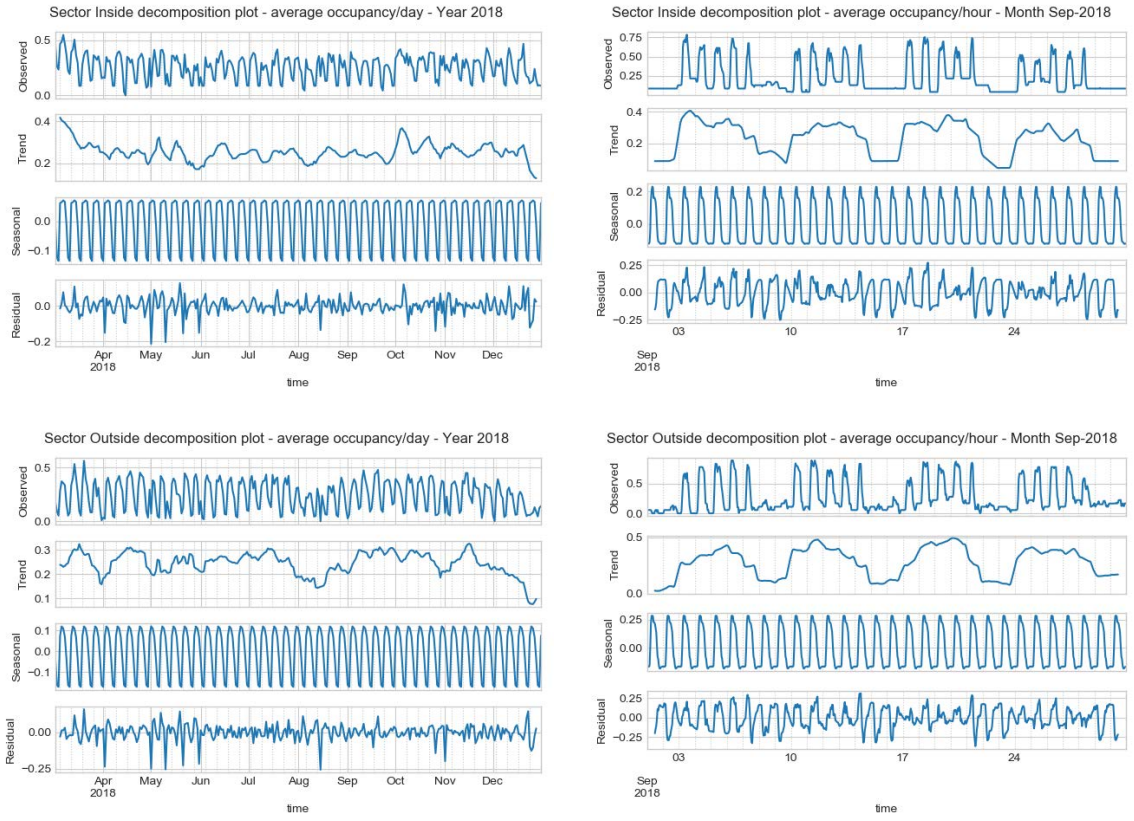


Figure 3.52: Decomposition of the time series for Inside (top) and Outside (bottom) sectors in Wattens. Left images correspond to occupancy/day for year 2018, and the right images are for occupancy/hour for September 2018.

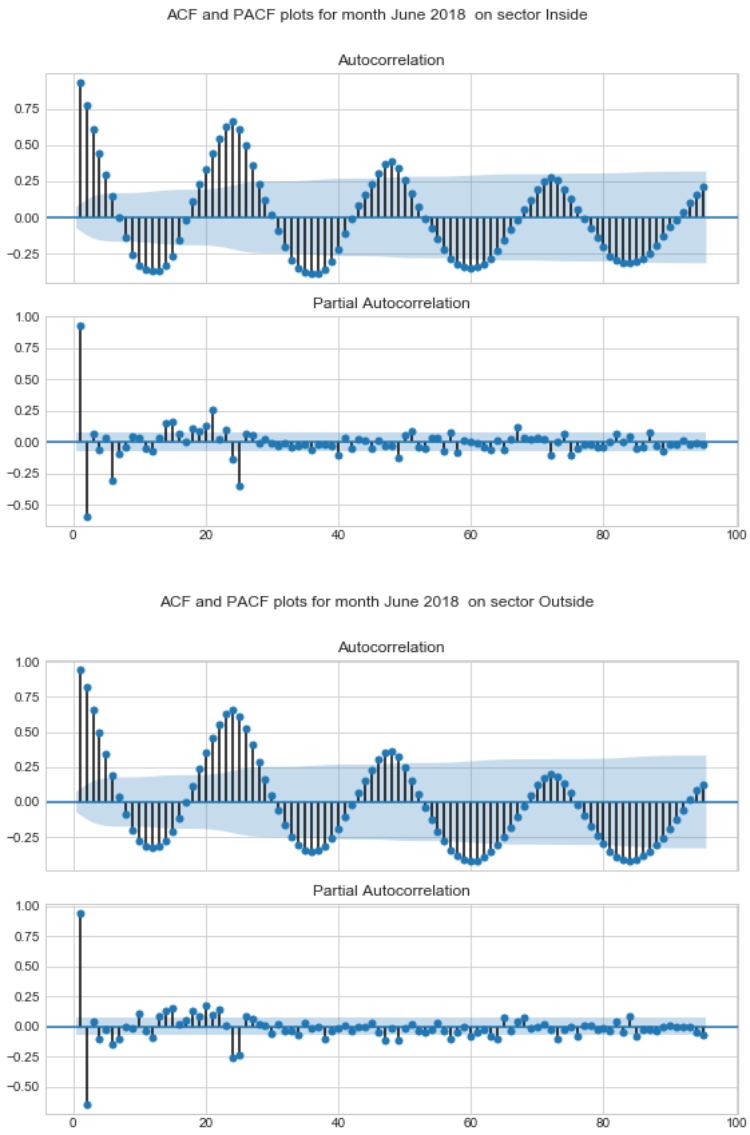


Figure 3.53: ACF and PACF plots for Inside (top) and Outside (bottom) sectors in Wattens, June 2018.

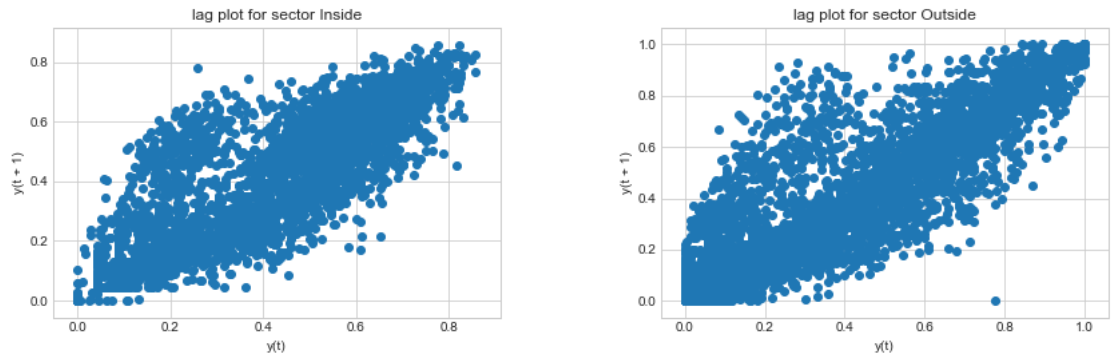


Figure 3.54: Lag plots of Wattens sectors. Left: Inside sector; right: Outside sector.

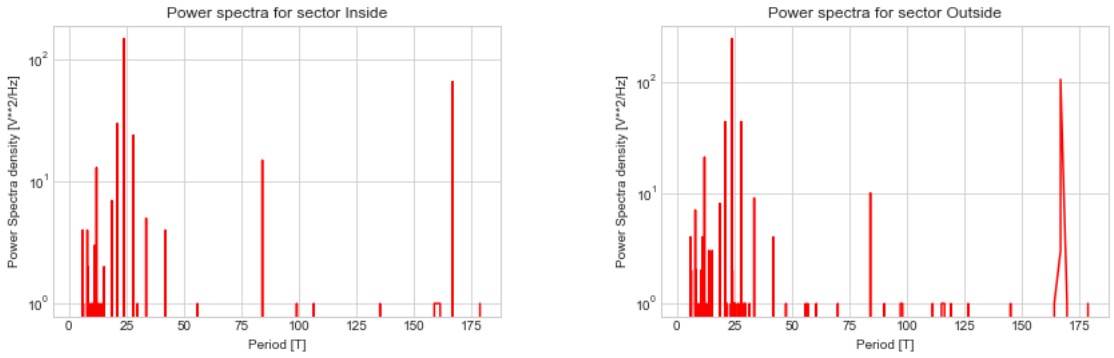


Figure 3.55: Wattens occupancy/hour in the frequency domain. Left: Inside sector; right: Outside sector.

The series in the frequency domain (Figure 3.55 shows the same information as the previous power spectra plots (Figures 3.27 and 3.42, for example), where 24 and 168 are the predominant periods for the data.

Time-space analysis

From Figure 3.56, we can determine that a correlation exists between occupancies and between sectors at closer time steps. Specifically, a linear relationship exists from time lag 0 to time lag 3, followed by the relationship fading until time lag 24, where the daily period begins again.

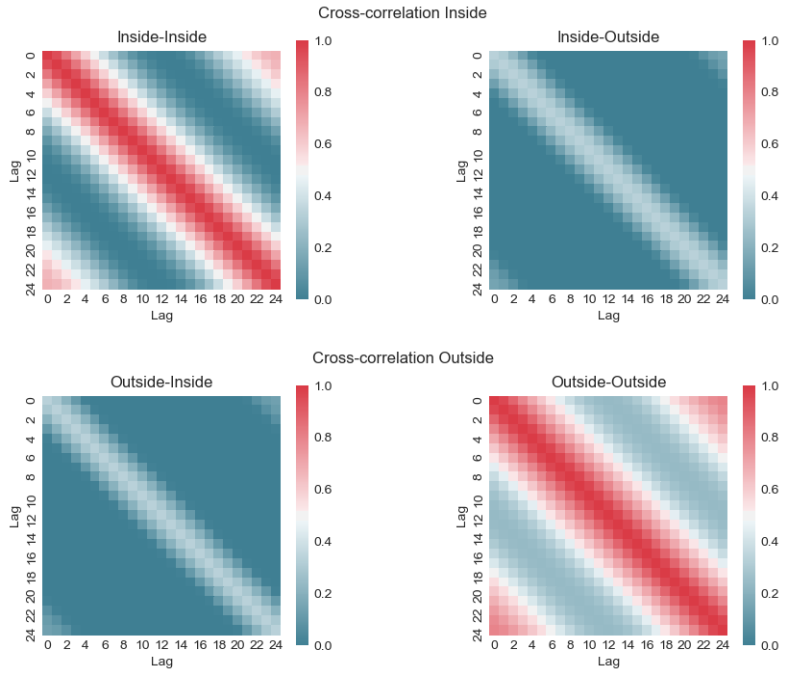


Figure 3.56: Cross-correlations for Wattens sectors. Inside sector on the top, Outside sector on the bottom.

Sector name	*Number of sensors*	*Deployment date*
Willow Standard	60	01/04/2018
Willow_Permit	31	01/04/2018
Willow_Disabled	5	01/04/2018
Willow Zipcar	4	01/04/2018

Table 3.10: Los Angeles Willow Street sectors.

3.5.4 Los Angeles (USA)

Characteristics

The Los Angeles scenario on Willow Street began in the spring of 2018 and consists of four sectors, for a total of 100 sensors in the same parking zone (see Table 3.10). Figure 3.57 shows that the sectors are next to each other, while the differences lie in their users: Standard for all types of users; Zipcar for vehicles rented from the Zipcar company; Permit are places reserved for people with permits to park in these specific parking lots; and Disabled are parking spots reserved for disabled people. Apart from the specific characteristics that define each parking sector, another common and special characteristic is that they are next to the metro station.

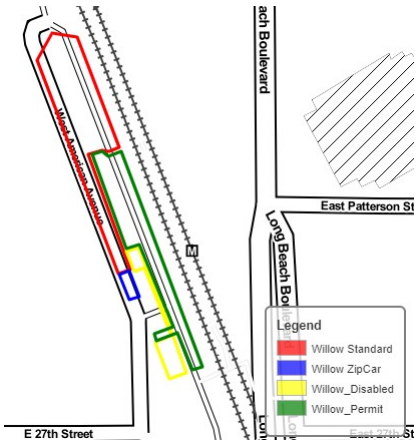


Figure 3.57: Los Angeles Willow Street map. Sectors are located near the metro station..

Analysis of all available data for the entire period

Collecting data from Los Angeles Willow Street started on the first of April, 2018. At first sight, we see clear patterns of daily and weekly periods. On 26 January 2019, infrastructure works on the metro line affected the parking places, with access to some of the parking spots being closed. Because of this, the occupancy in some sectors dropped to very low levels. At the end of 2019, some of the sectors started to recover their normal activity. In the area of outlier detection and intervention analysis for time series, this is identified as a level shift (LS) outlier, where the mean value of the series abruptly changes. In the analysis of these sectors, the date range considered is from 01/04/2018 to 24/10/2019, and care has been taken to mention and provide visualizations for those parts of the analysis that are affected by the level shift outlier. The last date of data is 24/10/2019, because from that date forward all sectors were equally affected by the appearance of missing values with unknown sources.

Statistics

In this section, two tables are presented for each sector. The corresponding first one presents the statistics that consider the level shift outlier event, and the second provides only the data up to when the level shift occurred.

Tables 3.11 present statistics for the Standard Willow sector. The level shift reduced the values of the quantile and the descriptive statistics. It is interesting to note that the kurtosis sign shifted from negative to positive (platykurtic to leptokurtic shape) and the skewness value increased. Observing the density plots in Figure 3.59, we can see that even though the distribution shape is maintained (that is, two peaks occur, with one taller than the other – a characteristic shared by sectors that are used differently on weekends/weekdays or at daytime/nighttime), the density values have become lower because fewer parking spots are available. Thus, full occupancy has dropped to lower

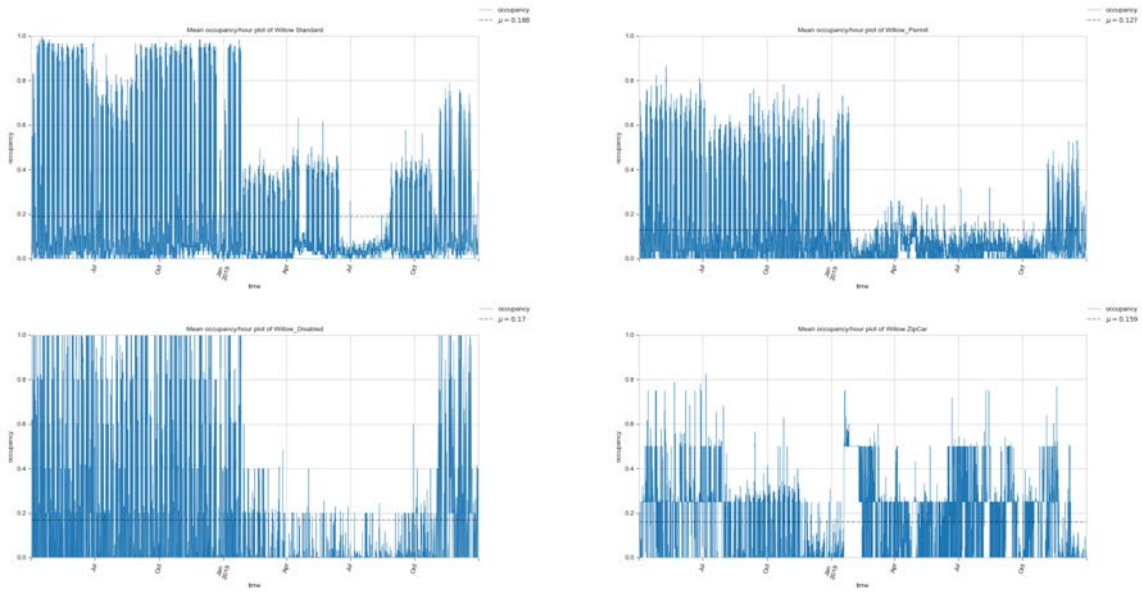


Figure 3.58: Los Angeles Willow Street sectors. Data for the interval 2018 and 2019. From top left to bottom right: Standard, Permit, Disabled and Zipcar.

than 1. Independently of the lower occupancy levels due to fewer available spots, we can assume by the distribution having similar shapes that the underlying patterns are maintained over time.

The tables in 3.12 present statistics for the Willow Permit sector, the sector with the second most parking spots in the zone. The level shift reduced the values of the quantile and the descriptive statistics. Here also, the kurtosis sign shifted from negative to positive (platykurtic to leptokurtic shape), though the skewness value was duplicated. These observations are shared with the Standard sector(Table 3.11). Observing the density plots in Figure 3.60, we can see that before the level shift, the sector behaved similarly to the Standard sector (Figure 3.59); but after the level shift, this sector seems to have been very much affected by the works as many parking spots become inaccessible. This explains the great shift to a leptokurtic shape.

Standard and Permit sectors seem to behave very similarly. Their correlation coefficient is very high, and they can be explained by the same underlying process.

The quantile statistics in Table 3.13 show that the Disabled sector is generally empty, with the most use before the level shift. Similarly to the Standard and Permit sectors, the main change in the descriptive statistics due to the level shift results from a shift in the kurtosis sign and the skewness value being duplicated.

Observing the density plots in Figure 3.61, the shape of the distribution is similar to the Standard and Permit sectors before the level shift. Once the level shift occurs, the shape is maintained by the occupancy levels never reaching 1.

Finally, the quantile statistics in Table 3.14 show that the values for the Zipcar sector

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0.016
Median	0.05
Q3	0.251
95-th percentile	0.907
Maximum	1
Range	1
Interquartile range (IQR)	0.235

Descriptive statistics	
Standard deviation	0.278
Coefficient of variation	1.463
Kurtosis	1.445
Mean	0.190
Median Absolute Deviation	0.215
Skewness	1.661
Variance	0.077

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0.03
Median	0.086
Q3	0.586
95-th percentile	0.944
Maximum	1
Range	1
Interquartile range (IQR)	0.556

Descriptive statistics	
Standard deviation	0.338
Coefficient of variation	1.194
Kurtosis	-0.742
Mean	0.283
Median Absolute Deviation	0.296
Skewness	0.953
Variance	0.115

Table 3.11: Statistics for the Standard sector. The top table considers all available data while the bottom table considers only data until 26-01-2019

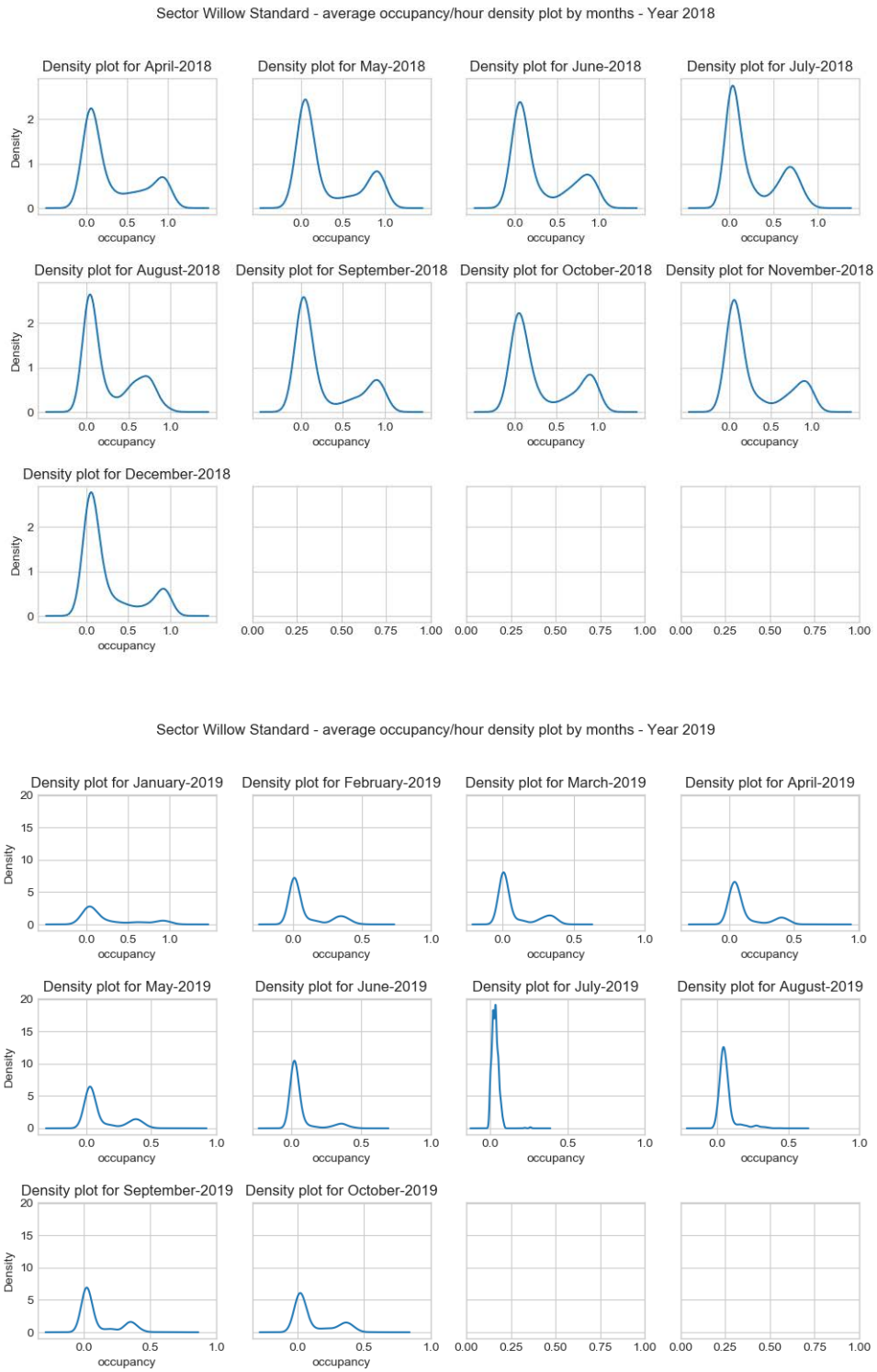


Figure 3.59: Density plots for the Standard sector on Willow Street.

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0.004
Median	0.042
Q3	0.123
95-th percentile	0.601
Maximum	0.865
Range	0.865
Interquartile range (IQR)	0.119

Descriptive statistics	
Standard deviation	0.190
Coefficient of variation	1.5
Kurtosis	2.048
Mean	0.127
Median Absolute Deviation	0.139
Skewness	1.818
Variance	0.036

Quantile statistics	
Minimum	0
5-th percentile	0.043
Q1	0.019
Median	0.077
Q3	0.410
95-th percentile	0.652
Maximum	0.865
Range	0.865
Interquartile range (IQR)	0.390

Descriptive statistics	
Standard deviation	0.234
Coefficient of variation	1.153
Kurtosis	-0.661
Mean	0.203
Median Absolute Deviation	0.204
Skewness	0.930
Variance	0.054

Table 3.12: Statistics for the Permit sector. The top table considers all available data; the bottom table considers only data until 26-01-2019

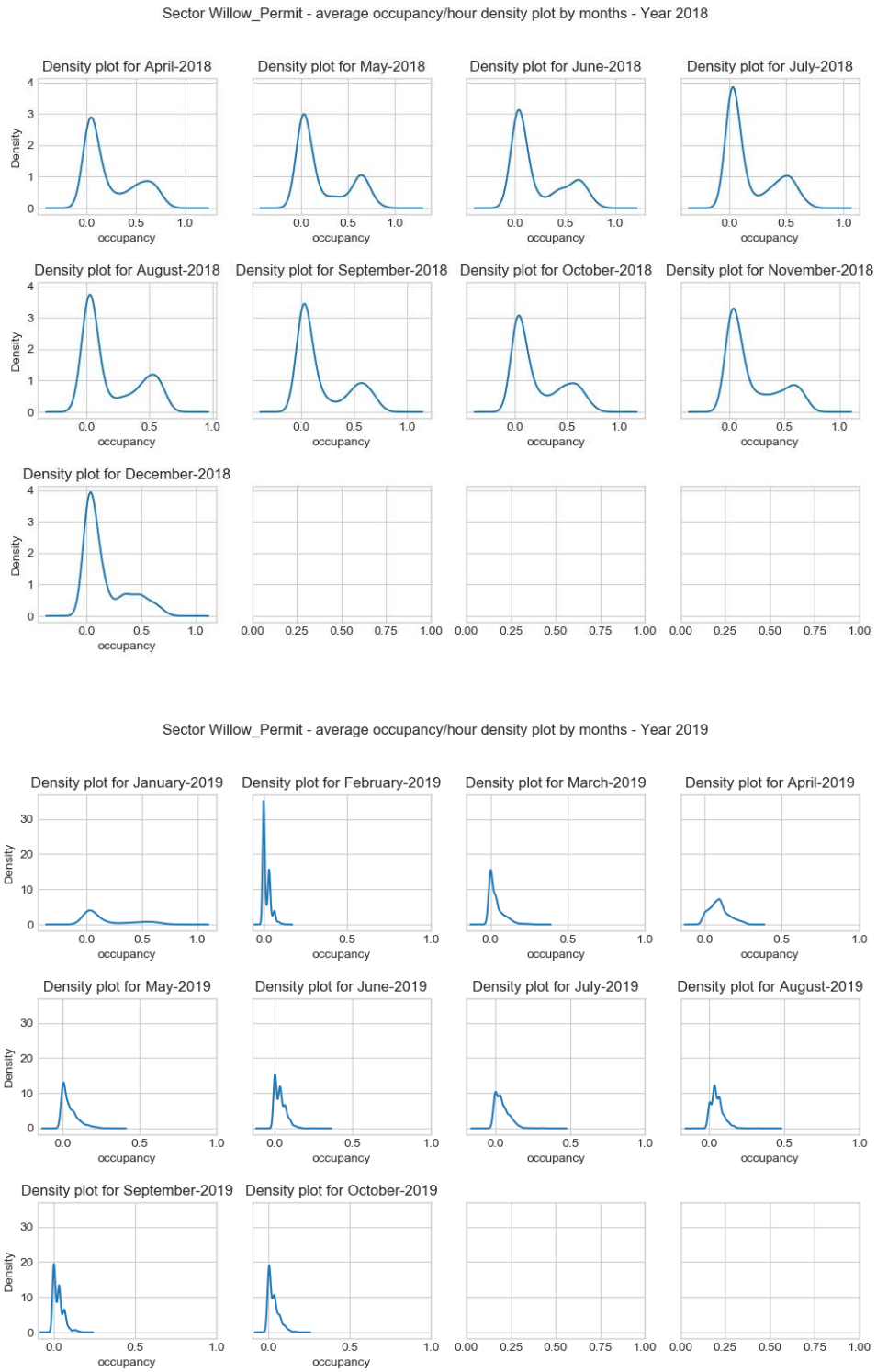


Figure 3.60: Density plots for the Permit sector on Willow Street.

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0
Median	0
Q3	0.2
95-th percentile	0.9
Maximum	1
Range	1
Interquartile range (IQR)	0.2

Descriptive statistics	
Standard deviation	0.288
Coefficient of variation	1.805
Kurtosis	2.046
Mean	0.16
Median Absolute Deviation	0.215
Skewness	1.823
Variance	0.083

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0
Median	0.081
Q3	0.573
95-th percentile	1
Maximum	1
Range	1
Interquartile range (IQR)	0.573

Descriptive statistics	
Standard deviation	0.35
Coefficient of variation	1.256
Kurtosis	-0.626
Mean	0.278
Median Absolute Deviation	0.301
Skewness	0.936
Variance	0.122

Table 3.13: Statistics for the Disabled sector. The top table considers all available data; the bottom table considers only data until 26-01-2019

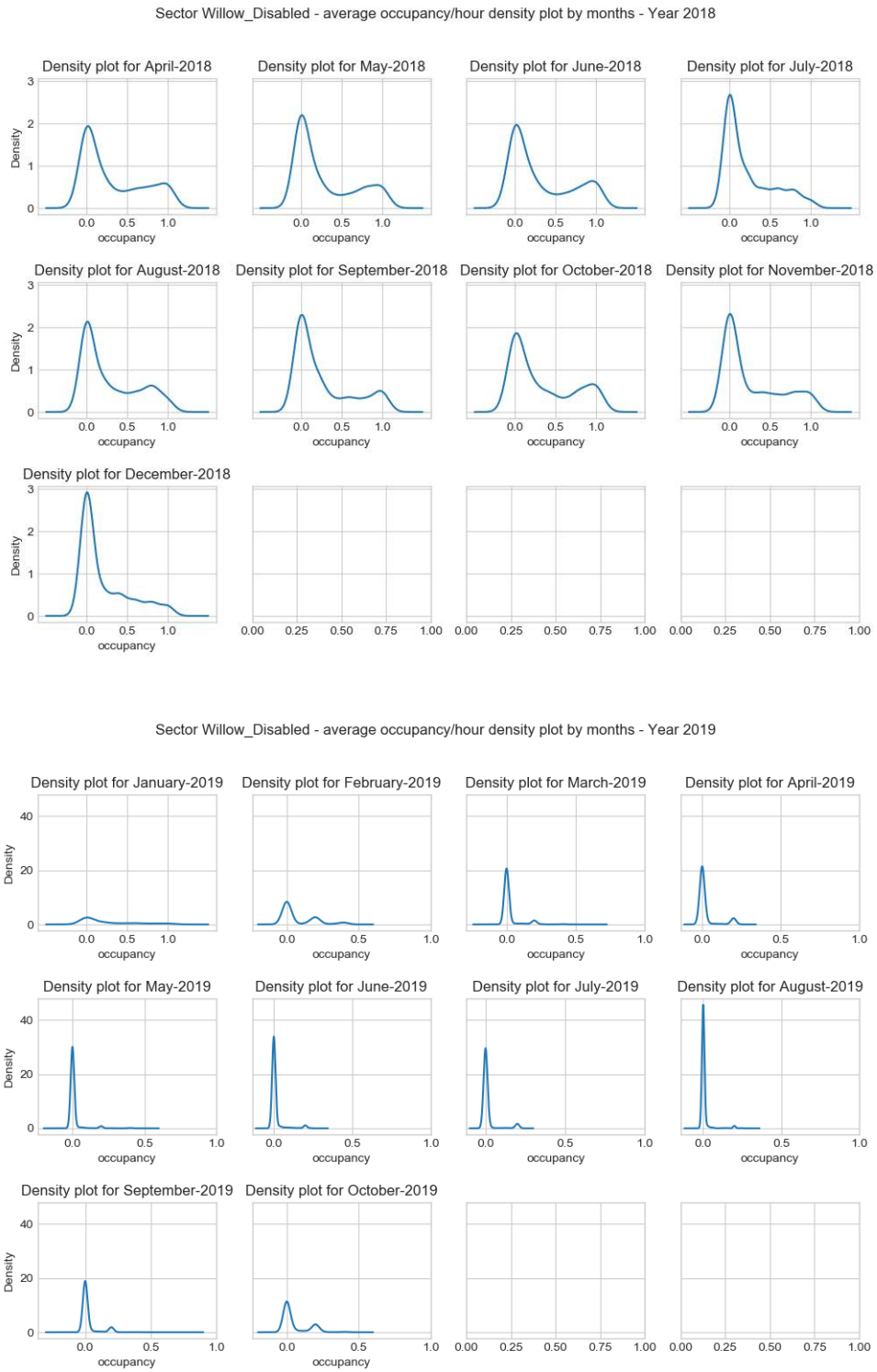


Figure 3.61: Density plots for the Disabled sector on Willow Street.

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0
Median	0.17
Q3	0.25
95-th percentile	0.5
Maximum	0.825
Range	0.825
Interquartile range (IQR)	0.25

Descriptive statistics	
Standard deviation	0.172
Coefficient of variation	1.056
Kurtosis	-0.374
Mean	0.16
Median Absolute Deviation	0.151
Skewness	0.706
Variance	0.03

Quantile statistics	
Minimum	0
5-th percentile	0
Q1	0
Median	0.017
Q3	0.25
95-th percentile	0.5
Maximum	0.825
Range	0.825
Interquartile range (IQR)	0.25

Descriptive statistics	
Standard deviation	0.17
Coefficient of variation	1.183
Kurtosis	0.134
Mean	0.144
Median Absolute Deviation	0.15
Skewness	0.92
Variance	0.029

Table 3.14: Statistics for the Zipcar sector. Top table considers all available data while bottom table only considers data until 26-01-2019

have remained the same independently of the level shift. Unlike the Standard and Permit sectors, the kurtosis statistic changed from positive to negative and the skewness decreased a little. The variance has remained more or less the same value. One reason for the low changes in the statistical values could be that these parking places were unaffected by the works in this zone.

The density plots in Figure 3.62 show that the parking places were empty for all of December 2018, but the distributions were unaffected overall by the level shift.

Monthly and weekly data analysis

Figures 3.63 show the behaviour of the Standard sector for October 2018 and 2019 (before and after the level shift). We can see that, while the occupancy/hour levels have decreased due to the parking spots being closed, the patterns persist. Thus, we can easily identify a daily pattern that distinguishes weekdays from weekends. This distinction could explain the two peaks in the distribution. At this point, we can assume that the level shift did not change the patterns in the sectors, only their levels of occupancy/hour.

Notice that the occupancy/hour levels of the daily pattern started to rise at 10 a.m. and then decrease at 23 p.m.

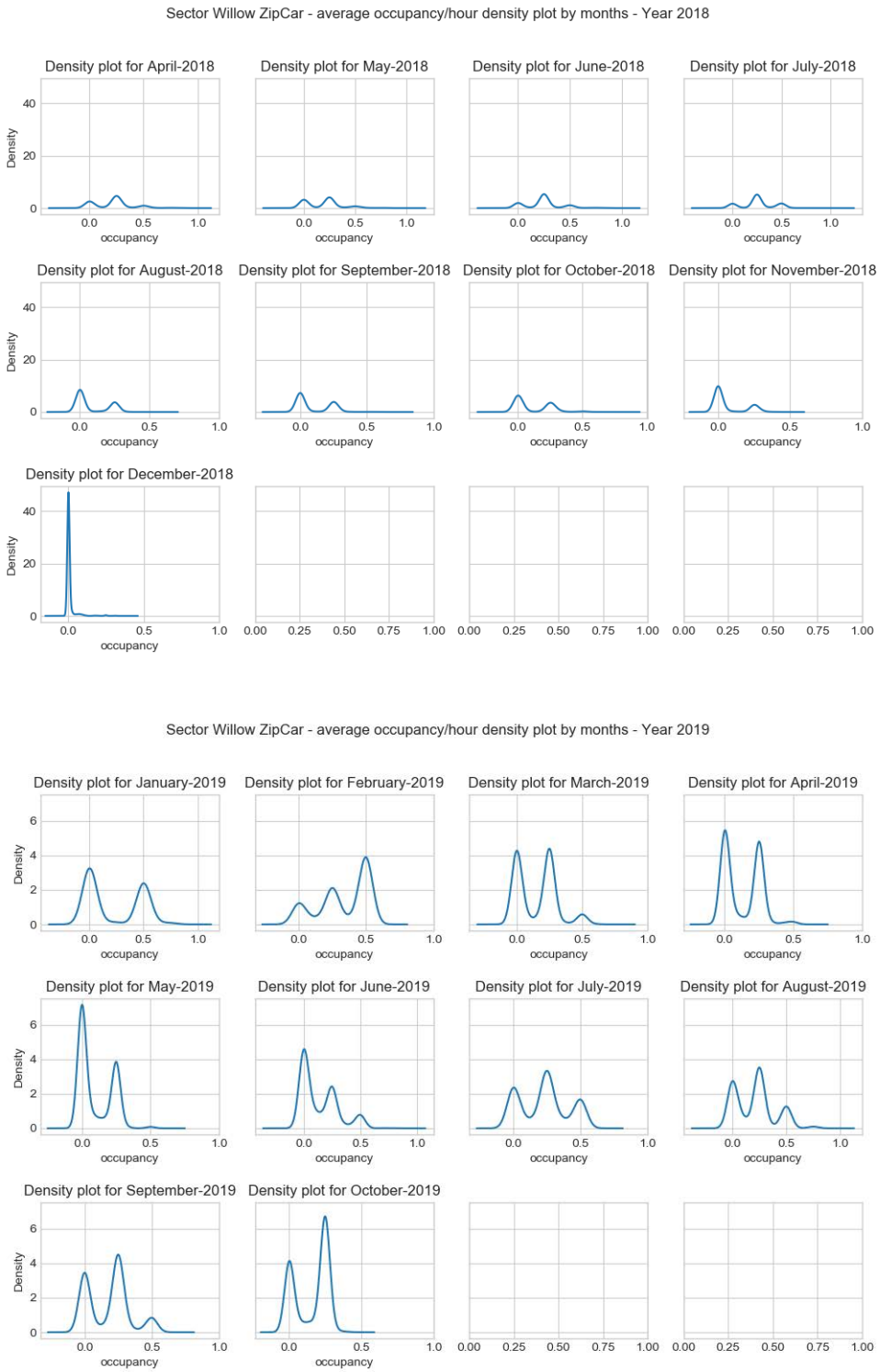


Figure 3.62: Density plots for the ZipCar sector on Willow Street.

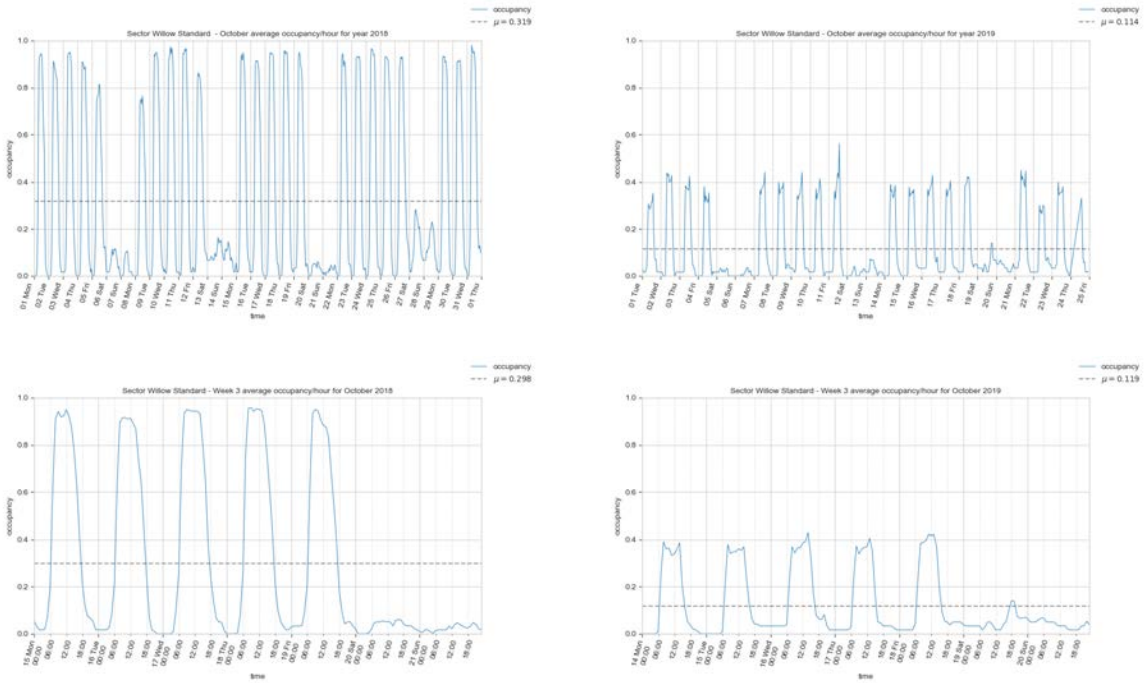


Figure 3.63: Top: October 2018 and October 2019 time series for the Standard sector. Bottom: the third week of October for the years 2018 and 2019, Standard sector.

The Permit sector was very affected by the infrastructure works in the zone, as evidenced by the plots in Figure 3.64. This could be because this sector is right at the entrance of the metro station. Daily patterns still exist after the works began, but it is difficult to distinguish weekdays from weekends.

The occupancy/hour time series for the Disabled sector on Willow Street in Figure 3.65 shows that the sector reaches full occupancy and emptiness at the same times every day throughout the work week. In a similar fashion to the other sectors, weekends and weekdays have different occupancy levels. Some of the parking spots were affected by the level shift in January 2019, but recovered their normal state in November.

The plots in Figure 3.66 show that the sector has not been affected by the infrastructure works in the zone. Moreover, the mean and the occupancy levels have remained more or less at the same values. Daily patterns are not guaranteed, as can be seen in the October 2019 plot, where the occupancy levels remained the same from 17 October until the end of the month.

The visualizations of the smoothed time series for the Standard sector reveals (Figure 3.67) that the occupancy clearly follows a daily pattern and that the occupancy is very marginal in the zone on weekends. Occupancy/hour levels increase at 6 a.m. and decrease at 18 p.m. every day from Monday to Friday, and the pattern is repeated throughout the year 2019, with lower occupancy levels due to the works in the zone.

From the seasonal plots for the Standard sector in Figure 3.68, and independently of

Figure 3.64: Top: October 2018 and October 2019 time series for the Permit sector. Bottom: the third week of October for the years 2018 and 2019, Permit sector.

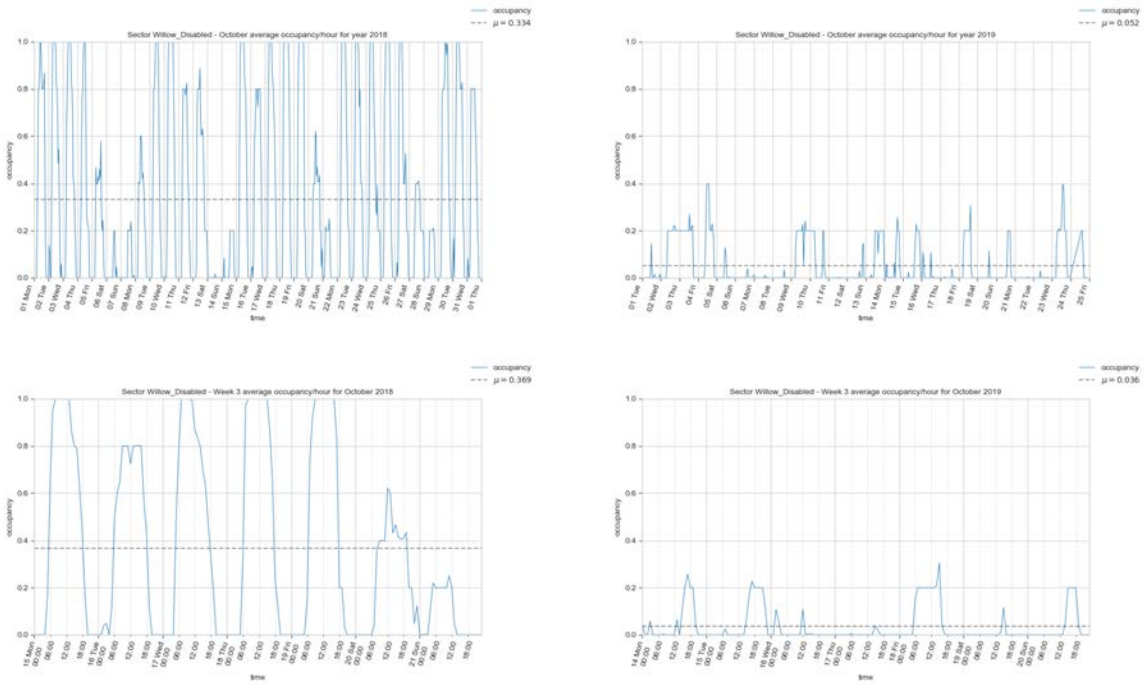


Figure 3.65: Top: October 2018 and October 2019 time series for the Disabled sector. Bottom: the third week of October for the years 2018 and 2019, Disabled sector.

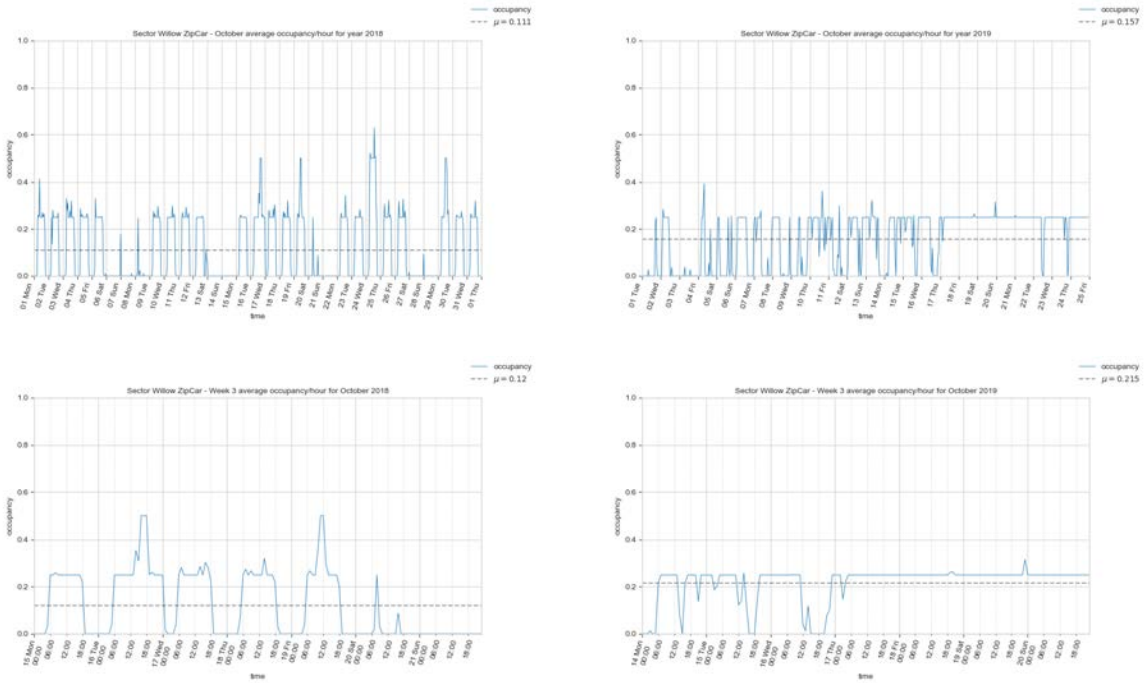


Figure 3.66: Top: October 2018 and October 2019 time series for Zipcar sector. Bottom: the third week of October for years 2018 and 2019 for Zipcar sector.

the level of aggregation (day, month, week or year), we can see that the occupancy pattern is always the same: it increases at 6 a.m. and starts to decrease at 16 p.m. The months with less occupancy are July, February and March (the last two because their data correspond only to 2019 and are affected by the infrastructure works in the zone). Similarly to other sectors (e.g., Figure 3.49), the first week of the month is the week with less occupancy, maybe due to holidays coinciding with that week. The decomposition by year makes the level shift evident, but the patterns still remain.

Similarly to the seasonal plots of the Standard sector (Figure 3.68), the seasonal plots from the Permit sector (see Figure 3.69) indicate daily patterns with a very discriminative visualization between weekdays and weekends. From the monthly aggregation, February and March indicate very low occupancy levels because their data is from 2019. Weekly occupancy/hour behaves similarly to the Standard sector, with less occupancy in first week of the month. Lastly, the yearly aggregation presents the effects on the occupancy/hour from the works in the zone.

The Disabled sector patterns (see Figure 3.70) behave similarly to the previous sectors (Standard, Figure 3.68 and Permit, Figure 3.69). The daily patterns distinguish weekdays from weekends and, while the levels of occupancy/hour follow the same pattern, we can see how February, March or 2019 data have lower values due to the works in the zone affecting access to the parking spots.

From monthly and weekly visualizations as well as the statistics tables (Figure 3.66 and

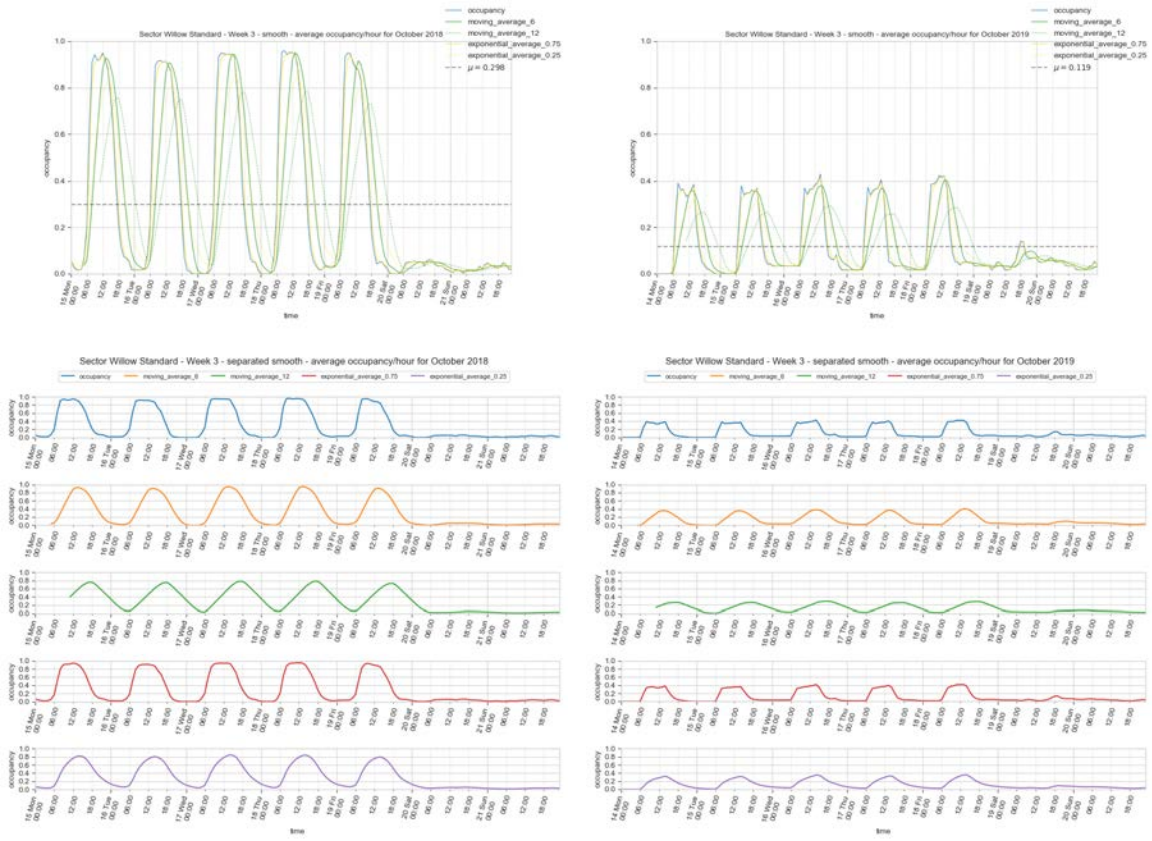


Figure 3.67: Smoothed time series for the Standard sector in Los Angeles. The upper image shows the series layered one on top of the other, where green lines represent the window moving average series and the yellow ones the exponential weighted average results. The bottom figure decomposes the top plot and shows the smoothness of the series under different methods.

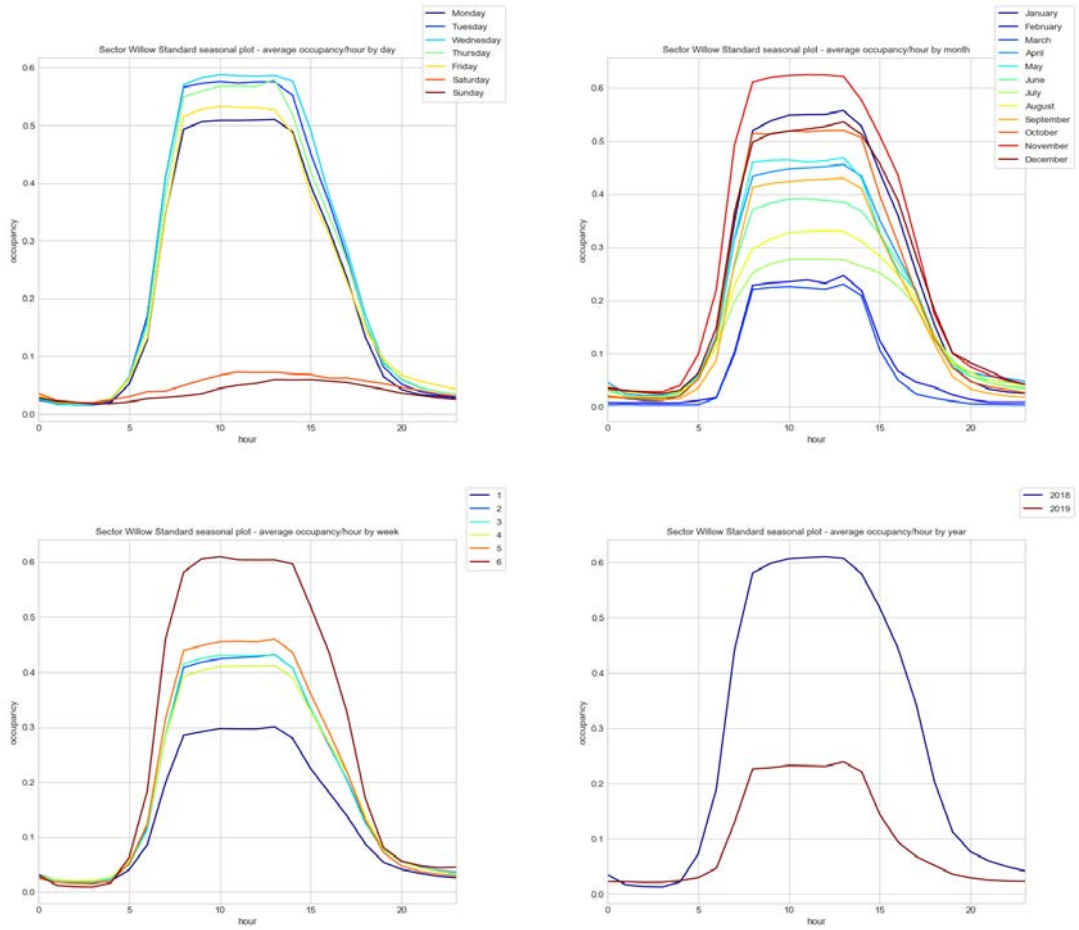


Figure 3.68: Occupancy/hour by days seasonal plots for Standard sector on Willow Street.

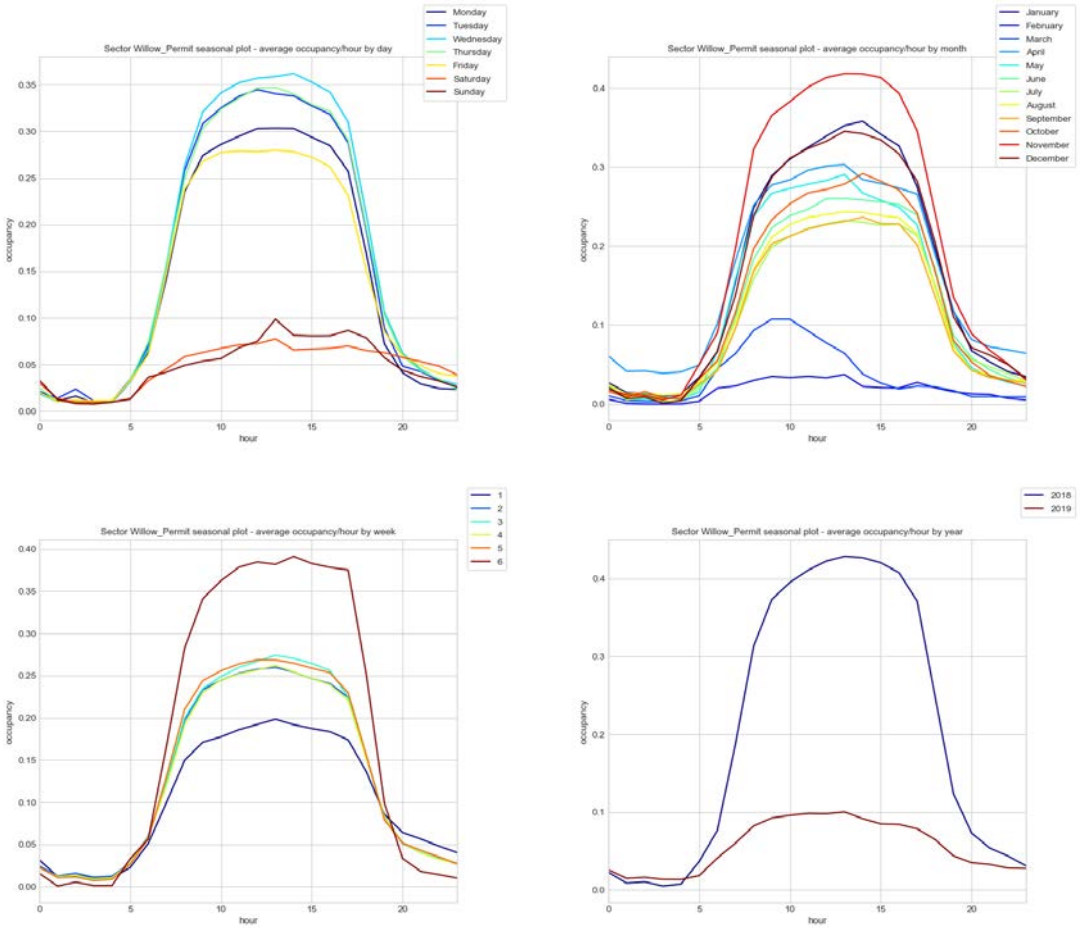


Figure 3.69: Occupancy/hour by days seasonal plots for Permit sector on Willow Street.

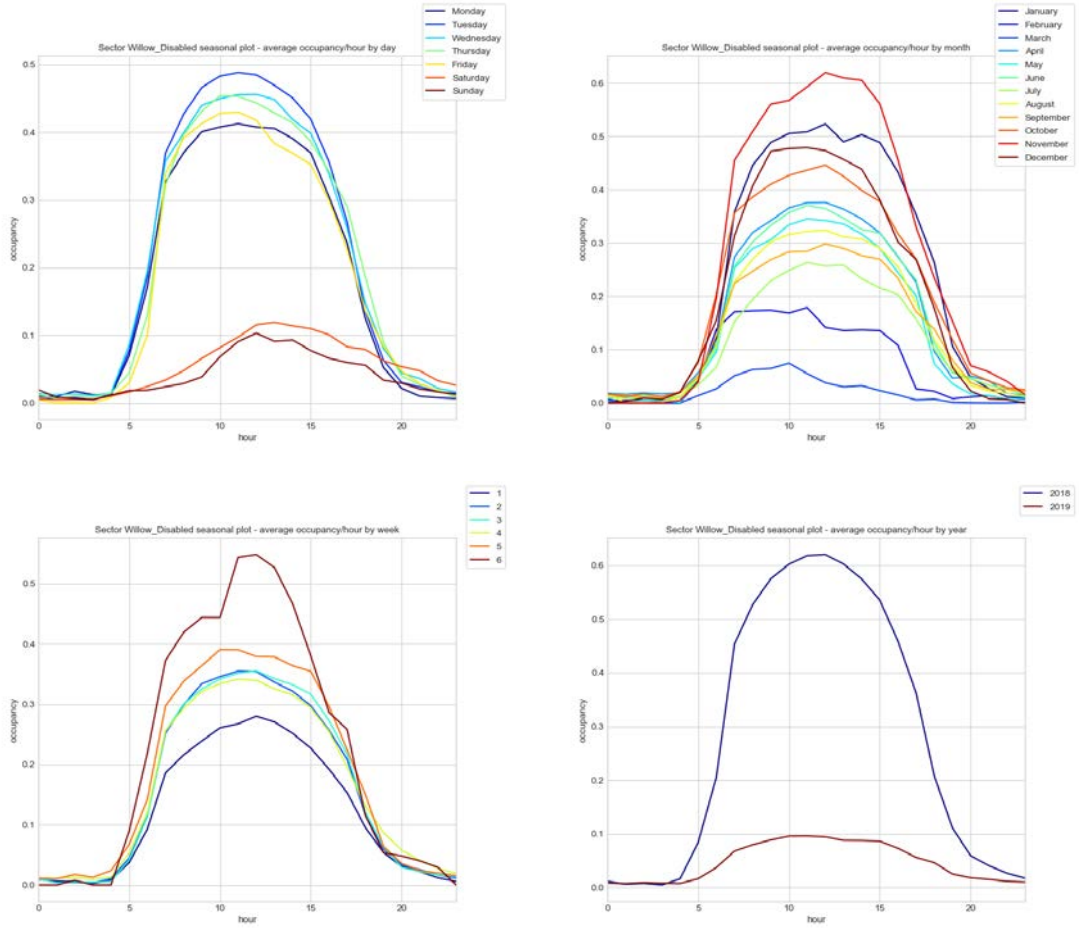


Figure 3.70: Occupancy/hour by days, seasonal plots for the Disabled sector on Willow Street.

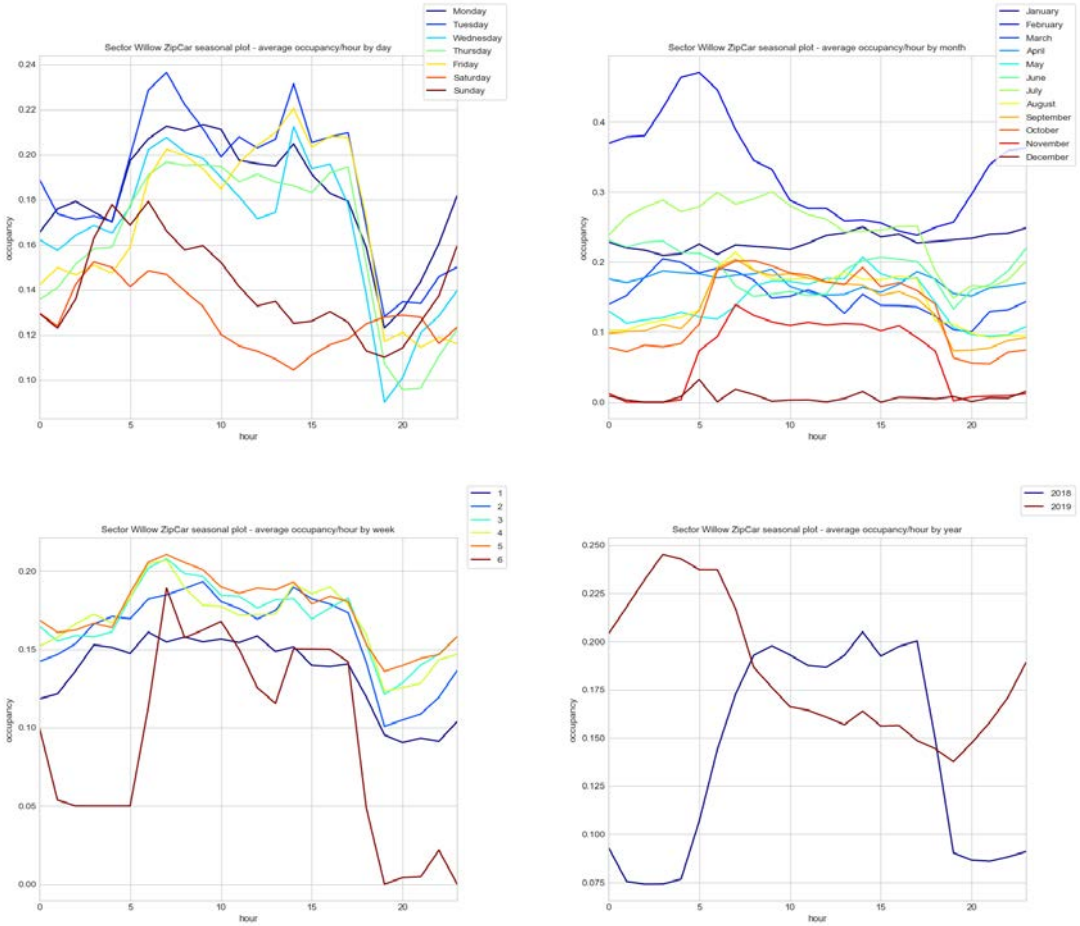


Figure 3.71: Occupancy/hour by days, seasonal plots for the Zipcar sector on Willow Street.

Table 3.14), it is known that the infrastructure works in the zone did not affect the behaviour of this sector. Because of the small number of parking places and, possibly, these spaces being purposed for rental car vehicles, the pattern seems to be chaotic: it increases at 5 a.m. and decreases at 18 p.m. It is not possible to distinguish weekdays from weekends like in the other sectors. Monthly, weekly and yearly aggregations do not present discriminative differences.

Specialized time series plots

The box plot visualizations (see Figure 3.72) reveal that the Standard, Permit and Disabled sectors have been governed by the same process, as they share similar mean and variance levels for the different months until 2019. Zipcar behaviour is more chaotic, with the mean changing from month to month while more or less maintaining the variance.

From the decomposition of the time series for the Standard sector in Figure 3.73, we can

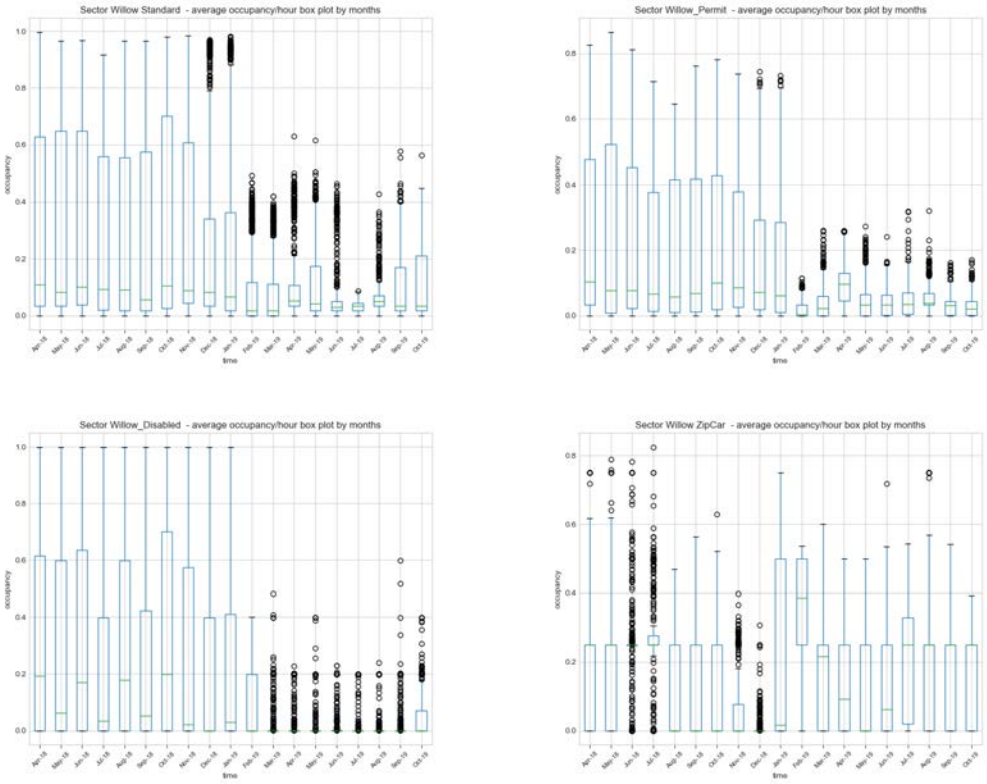


Figure 3.72: Box plot visualizations for the Willow Street sectors. From top left to bottom right: Standard, Permit, Disabled and Zipcar.

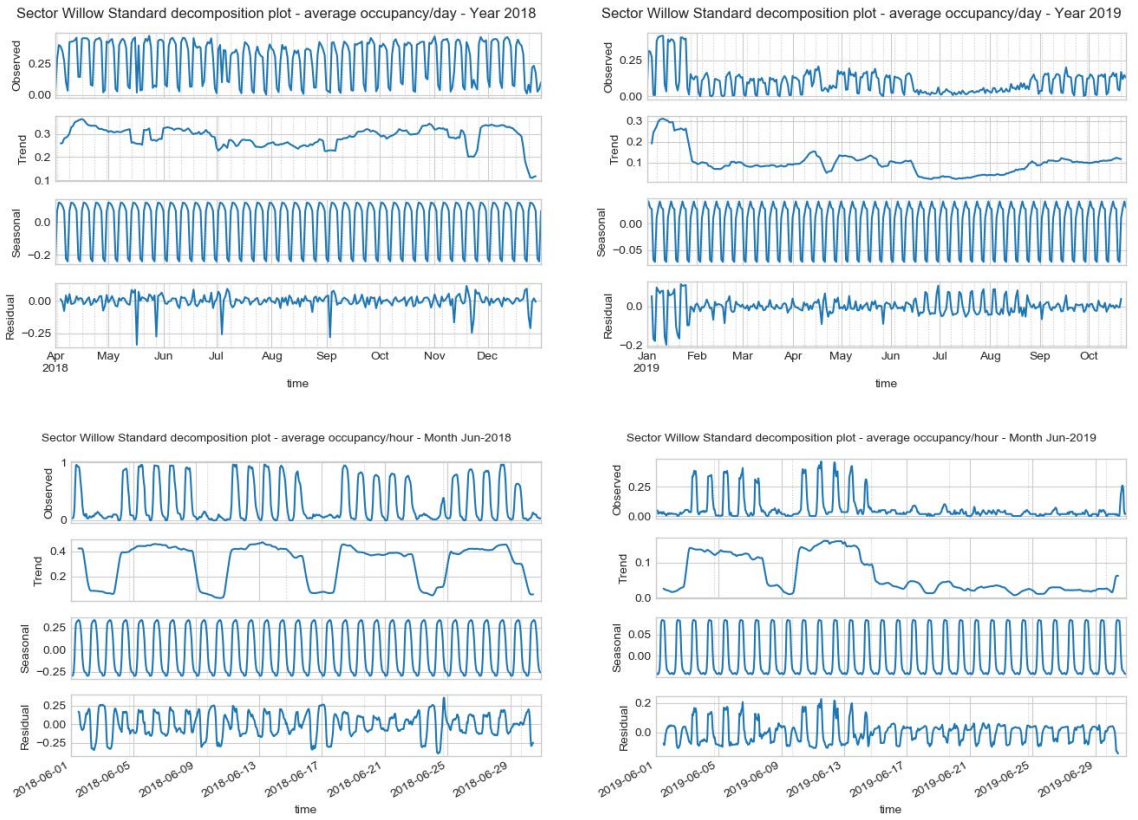


Figure 3.73: Decomposition of the time series for the Standard sector on Willow Street. Top corresponds to occupancy/day for years 2018 and 2019. Bottom corresponds to occupancy/hour for June 2018 and June 2019

see that the trend is maintained over time and, while the levels change after January 2019, the trend level continues with the same lower value over time. July, August and September of 2019 have a lower trend than expected. From the decompositions for June 2018, the trend clearly distinguishes occupancy levels between weekdays and weekends, and a daily seasonal pattern can be observed. A similar trend can be seen in June 2019, but in the last half of the month the occupancy dropped to nearly 0. This could be an effect from the works in the zone. Trend + seasonality cannot capture patterns in the data at all, as the residuals reach values of ± 0.25. In other words, there is a 1/4 difference between prediction and observation. This means that, working from 60 parking places, one prediction could be wrong for 15 parking spots.

The decomposition of the time series for the Permit sector (plots in Figure 3.74) show similar patterns to those revealed by the Standard sector, with the same outlier values in May, July, September and November of 2018. The trend value is maintained throughout the years, even in 2019 after the level shift. Daily seasonality is always present. When looking at the occupancy levels for June, the trend has a seasonal pattern (weekly) that is similar to the Standard sector. Seasonality plus trend has large residuals that are

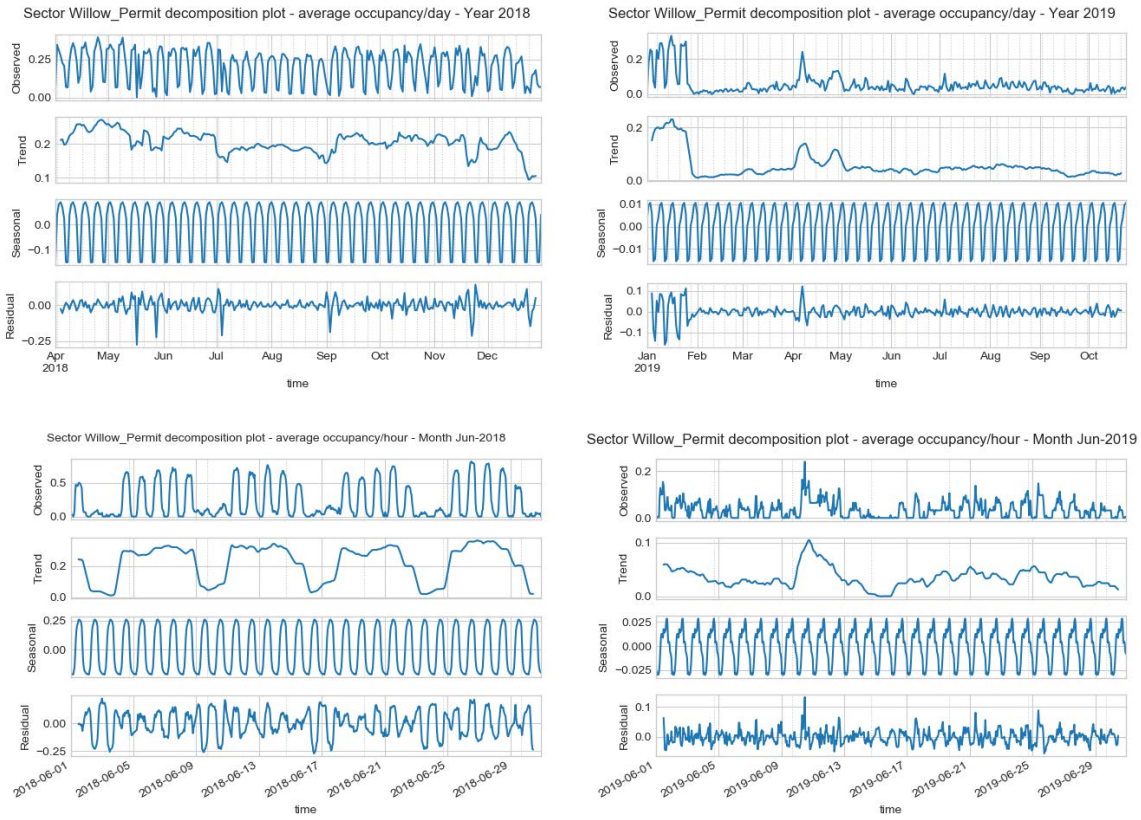


Figure 3.74: Decomposition of the time series for the Permit sector on Willow Street. Top corresponds to occupancy/day for years 2018 and 2019. Bottom corresponds to occupancy/hour for June 2018 and June 2019

similar to those of the Standard sector for weekends. For 2019, the patterns become hard to detect, as the occupancy levels are nearly 0.

Similarly to the previous sectors, the decomposition of the time series for the Disabled sector (see Figure 3.75) shows daily seasonality and a weak weekly pattern in the trend for occupancy/hour in June. Outliers in 2018 are the same as in previous sectors. Trend plus seasonality is not enough to capture the occupancy/hour patterns, and the residuals are large on weekends in the June 2018 plot. In the 2019 plots, we can see that the trend levels are maintained after the level shift while the daily seasonality is also maintained. The 2019 plot shows that the expected value for the time series lies in the 0 value; thus, when some occupancy occurs, its residual value is large.

Lastly, the decomposed time series for the Zipcar sector (Figure 3.76) shows erratic behaviour in the trend, independently of the year. What is more, the daily seasonality is more clear in occupancy/day than in occupancy/hour.

The ACF and PACF plots in Figure 3.77 show the autocorrelation and partial autocorrelation functions for the Willow Street sectors for May 2018. All plots show a strong

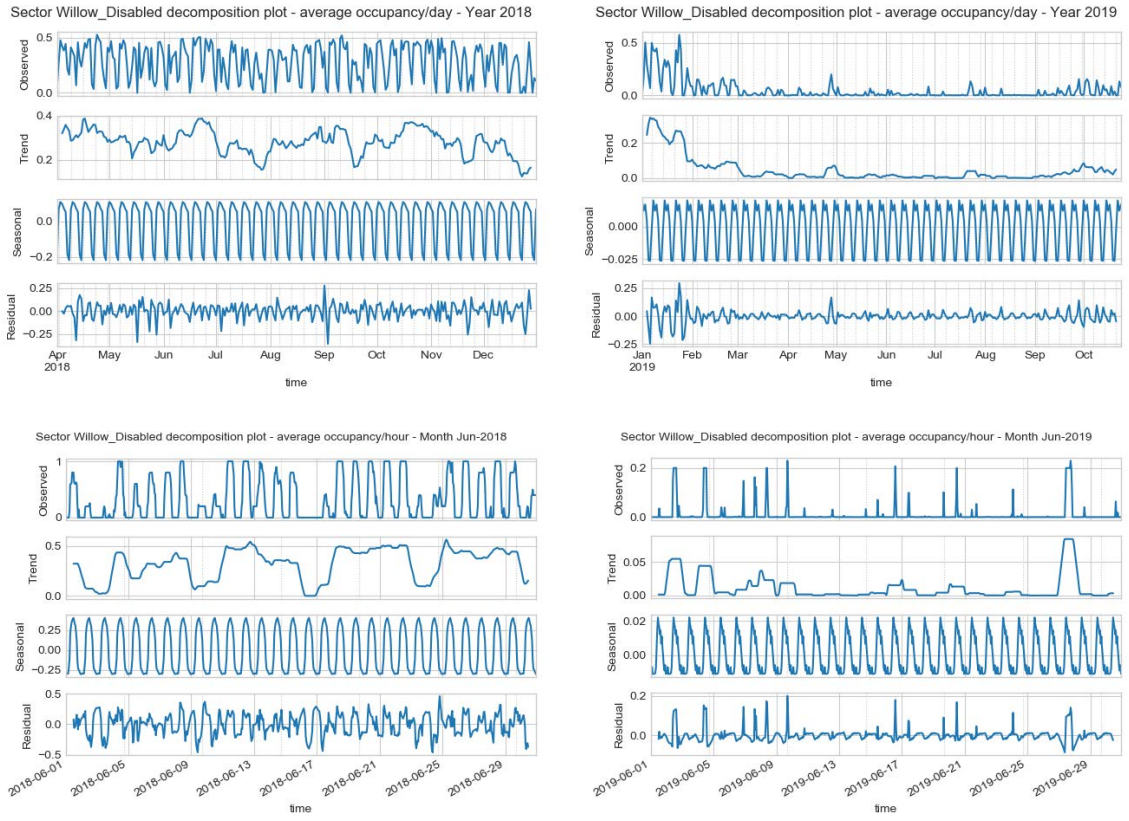


Figure 3.75: Decomposition of the time series for the Disabled sector on Willow Street. Top corresponds to occupancy/day for years 2018 and 2019. Bottom corresponds to occupancy/hour for June 2018 and June 2019

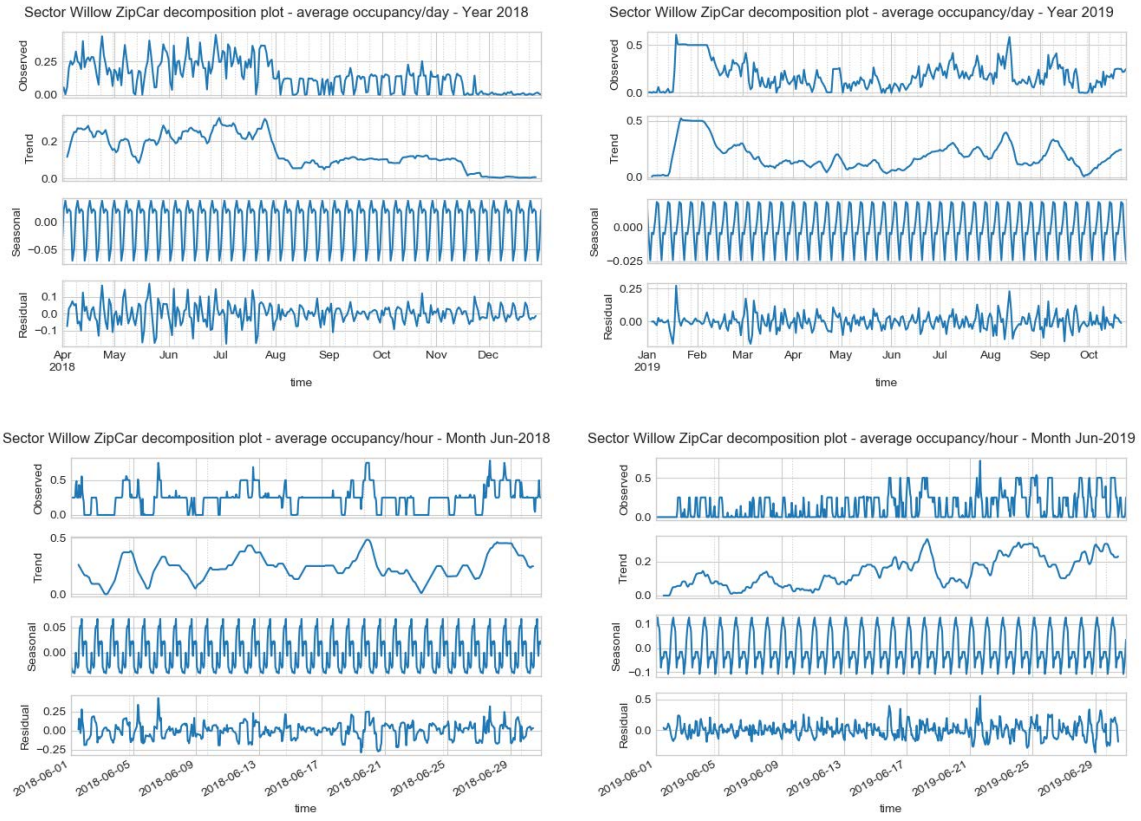


Figure 3.76: Decomposition of the time series for the Disabled sector on Willow Street. Top corresponds to occupancy/day for years 2018 and 2019. Bottom corresponds to occupancy/hour for June 2018 and June 2019

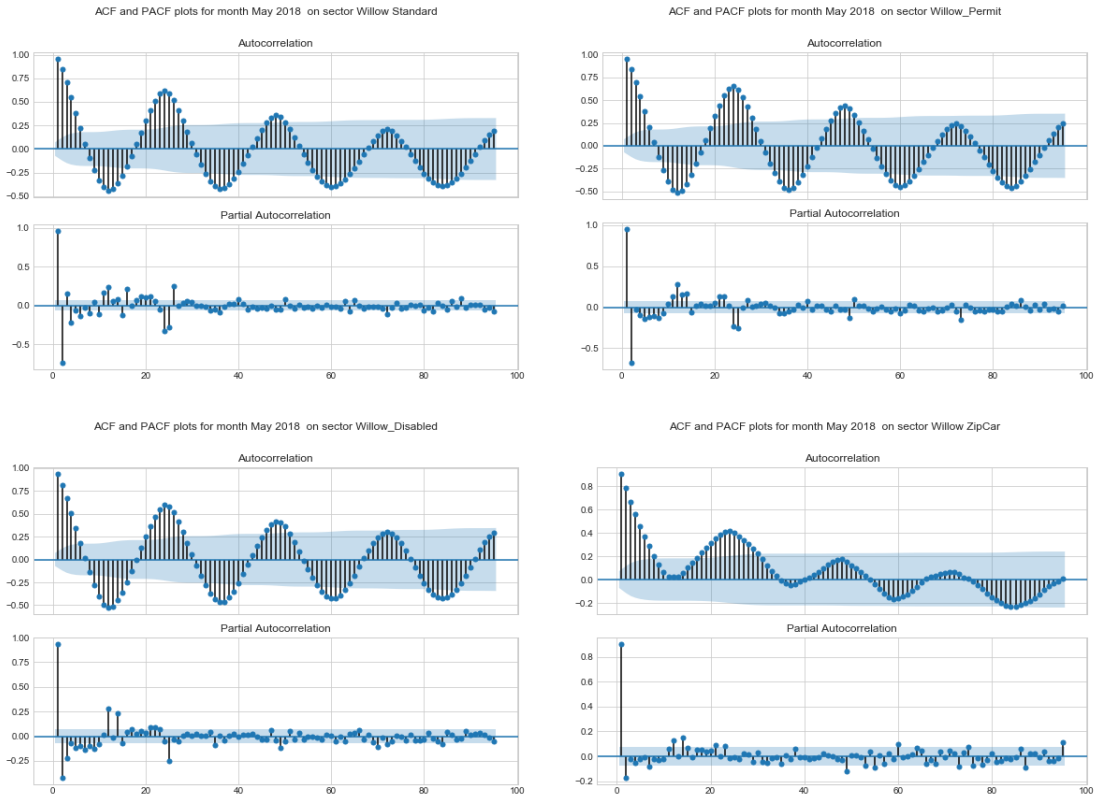


Figure 3.77: ACF and PACF plots for Standard (top left), Permit (top right), Disabled (bottom left) and Zipcar (bottom right) sectors on Willow Street for the year 2018.

autocorrelation process, where the first two lags are significant. The Standard, Permit and Disabled sectors additionally have an important lag, namely lag 25, which indicates daily seasonality.

Regarding the ACF and PACF plots for the year 2019 (Figure 3.78), the Standard and Permit sectors show a strong autoregressive pattern, with lags 1, 2 and 25 (daily seasonality) being important, while the behaviour of Disabled and Zipcar is a mixture of autoregressive and moving average. The PACF of the Disabled sector shows that lags 1, 2 and 25 have autoregressive patterns that can be combined with a moving average, as the ACF plot shows lags 1, 2 and 3 to be significant. The case is similar in the Zipcar sector, with lags 1 and 2 being significant in the PACF plot while lags 1, 2, 3, 4 and 5 are significant in the ACF plot.

Figure 3.79 shows the lag plots for the Willow Street sectors, indicating a strong linear correlation in the Standard and Permit sectors.

In the frequency domain, the periodic patterns emerge in the time series. Figure 3.80 shows the predominant periods for the Willow Street sectors. Similarly to the other sectors (see Figures 3.27 and 3.55 as examples), the predominant period is at value 24, and the second one is at value 168.

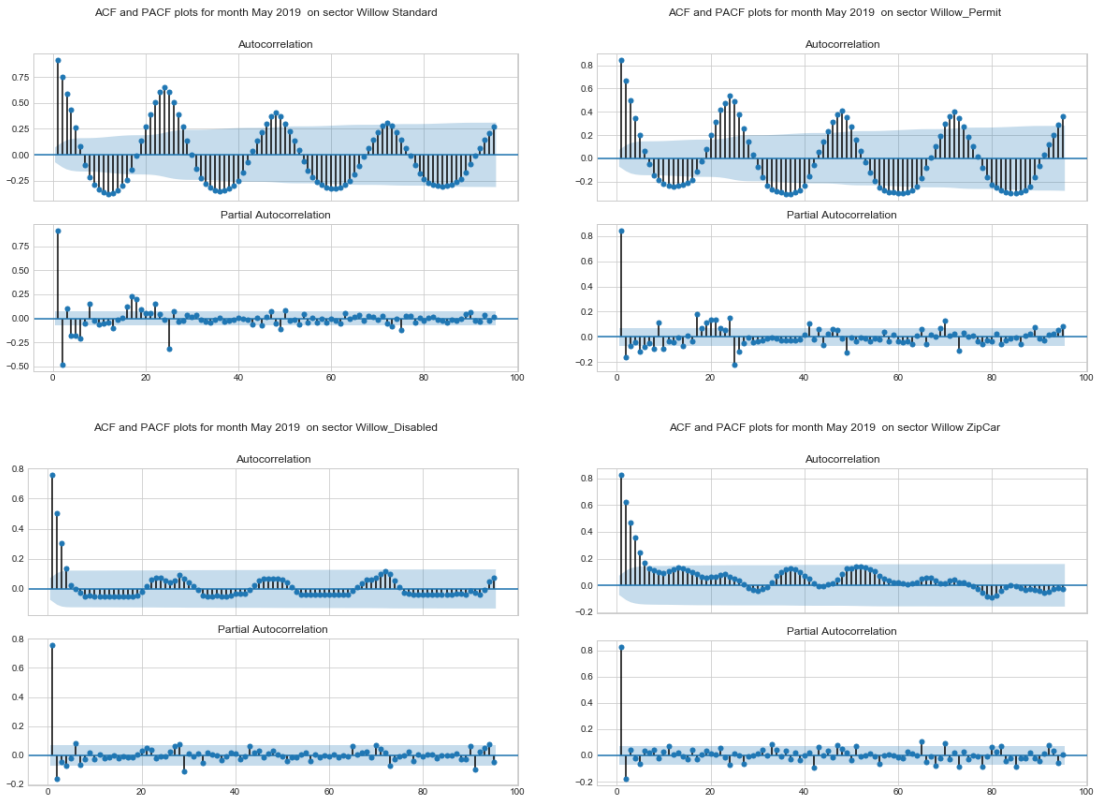


Figure 3.78: ACF and PACF plots for the Standard (top left), Permit (top right), Disabled (bottom left) and Zipcar (bottom right) sectors on Willow Street for the year 2019.

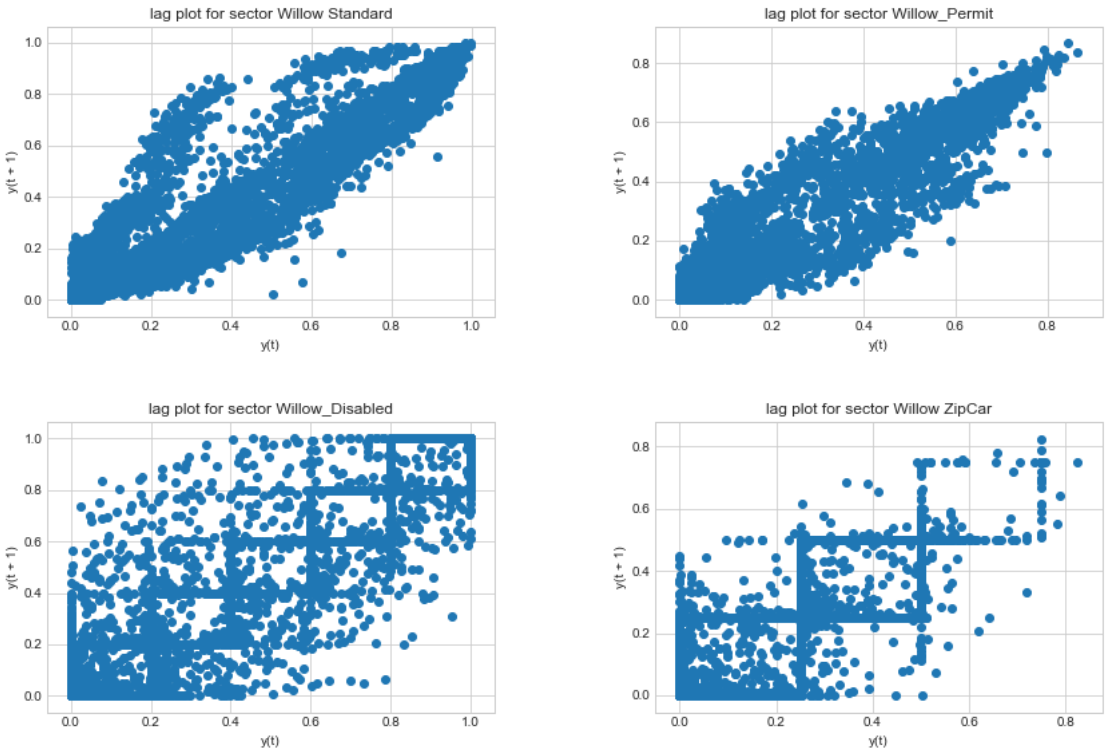


Figure 3.79: Lag plots for the Willow Street sectors. From top left to bottom right: Standard, Permit, Disabled and Zipcar

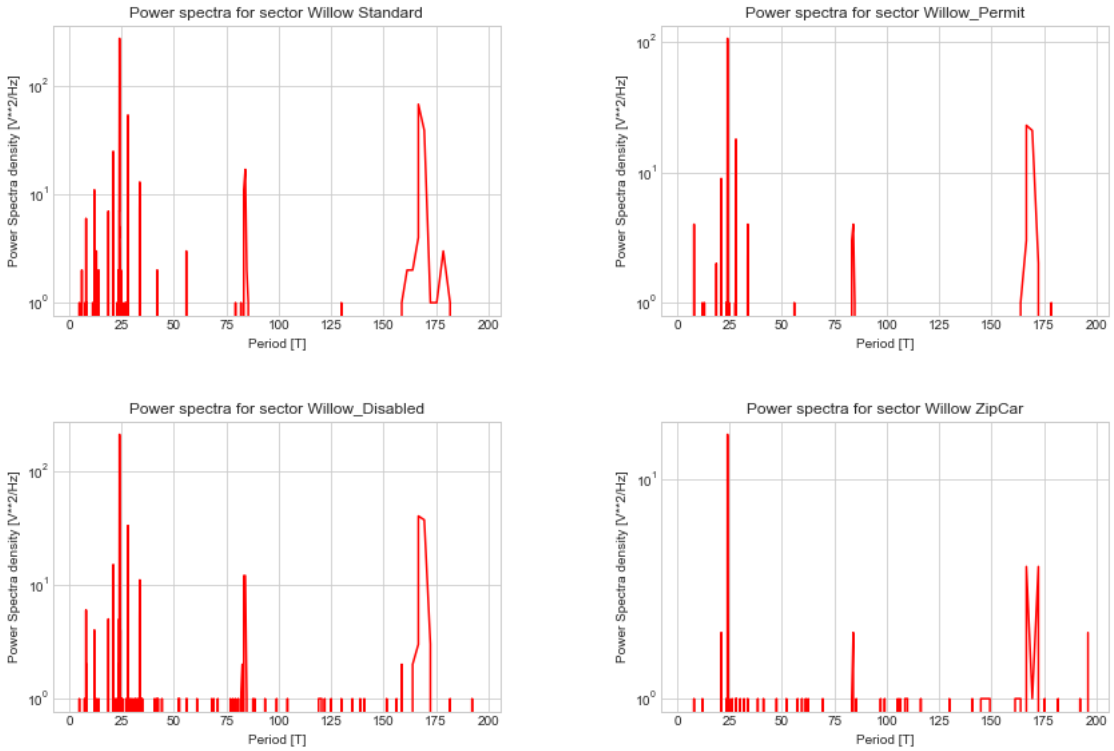


Figure 3.80: Willow Street occupancy/hour in the frequency domain.

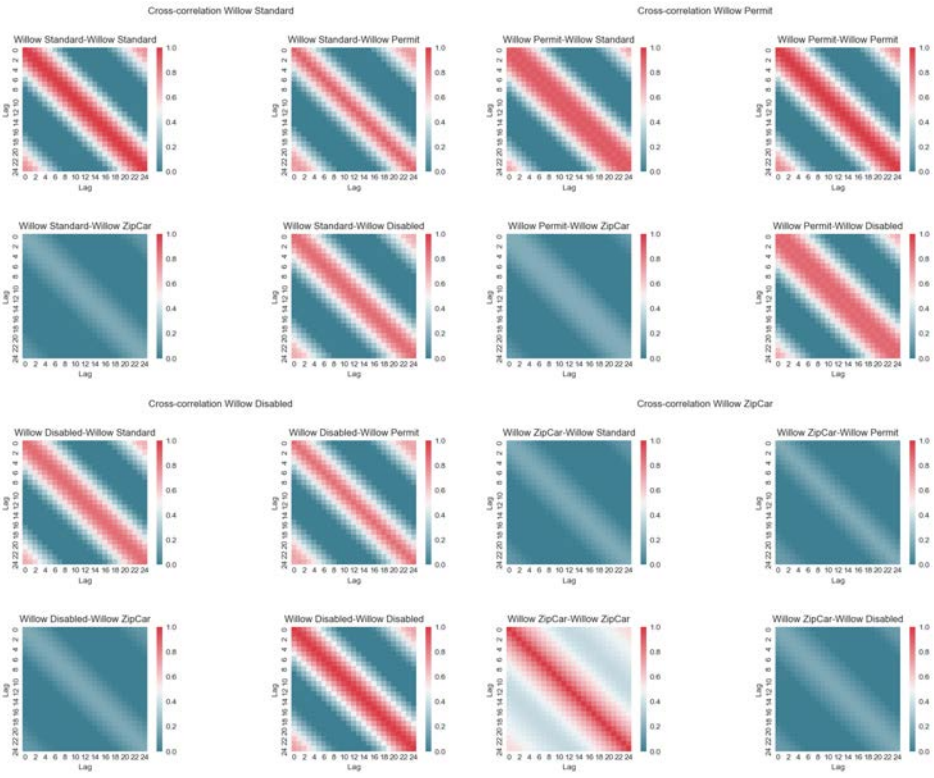


Figure 3.81: Cross-correlations for the Willow Street sectors. Top left for Standard sector, top right for Permit sector, bottom left for Disabled sector and bottom right for Zipcar sector.

Time-space analysis

From Figure 3.81, we can determine that a high correlation exists between occupancies among sectors at closer time steps. Specifically, a strong linear relationship exists from time lag 0 to time lag 6 between the Standard, Permit and Disabled sectors. The Zipcar sector is less correlated with the other sectors through lags 0 to 3.

3.5.5 Riyadh (UAE)

Characteristics

The last scenario considered in this data analysis is for the sectors in the city of Riyadh. This scenario has over 1000 sensors distributed across 10 sectors in the downtown area of the city, next to the Kingdom Tower skyscraper. For the specific number of sensors, check Table 3.15.

The main characteristic of this area is the huge amount of sensors that cover the entire district and provide information about parking behaviours at a large scale. Figure 3.82 shows the distribution of the sectors and sensors.

Sector name	*Number of sensors*	*Deployment date*
Sector 1	132	01/09/2018
Sector 2	115	01/09/2018
Sector 3	143	01/09/2018
Sector 4	81	01/09/2018
Sector 5	94	01/09/2018
Sector 6	94	01/09/2018
Sector 7	111	01/09/2018
Sector 8	74	01/09/2018
Sector 9	84	01/09/2018
Sector 10	79	01/09/2018

Table 3.15: Riyadh sectors..

Figure 3.82: Map of the Riyadh sectors.

Individually, each sector has its own characteristics that can affect its behaviour. Sectors 1, 2, 3 and 4 are situated in front of the Kingdom Tower skyscraper, an area that consists of offices and restaurants for workers in the zone. Sector 5 is surrounded by luxurious hotels. Sectors 6 to 10 are part of the zone surrounding a medical complex: the maternity hospital, emergencies, etc. Also, sectors 6, 7 and 8 are next to one of the biggest malls in the area.

In this scenario, sectors are not classified by parking spot uses, like in the Los Angeles scenario (see Section 3.5.4 for detailed information). Thus, in this case, all spots pertain to the general definition of a parking place on a city street: free and without restrictions based on the type of user.

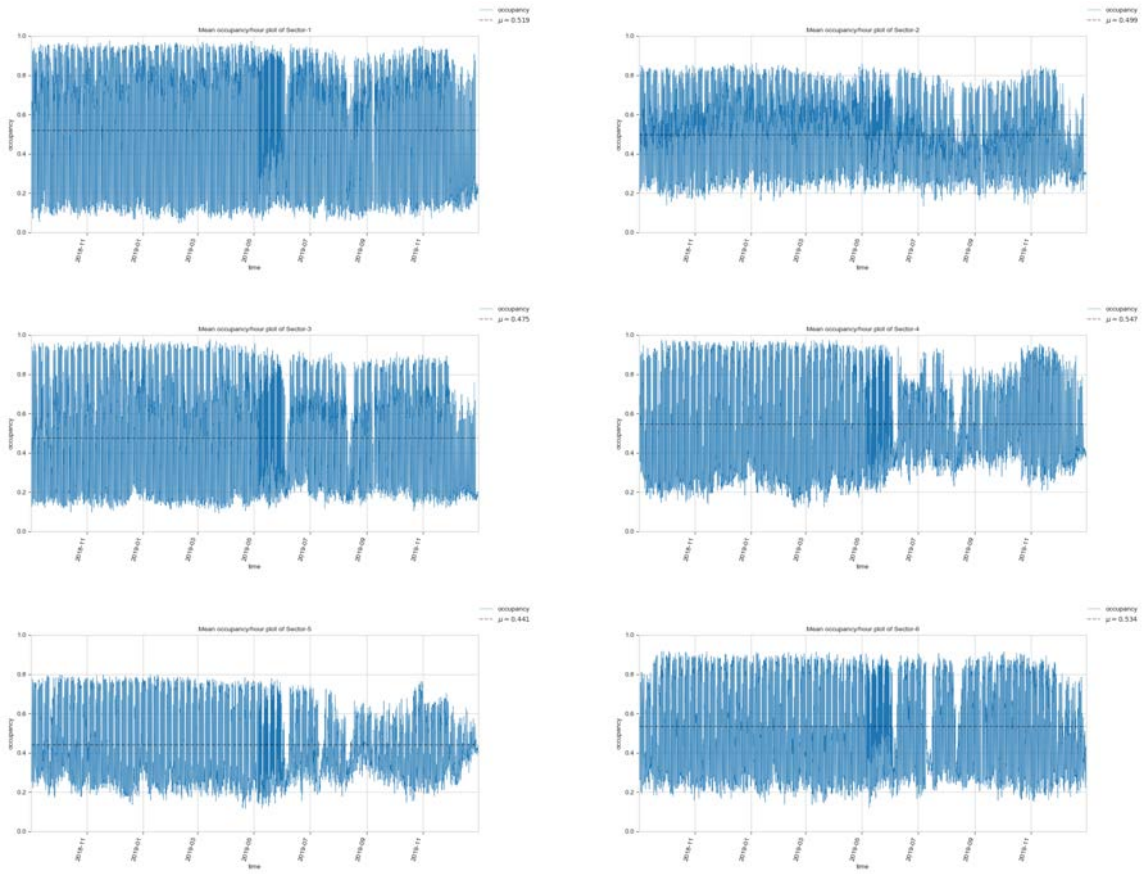


Figure 3.83: Riyadh data from the time interval 2018 to 2019. From top left to bottom right: sector 1 to sector 6.

Analysis of all available data for the entire period

Observing all the data available from the first six sectors in Riyadh (see Figure 3.83), they behave very similar, with means at the 0.5 value. It seems that anomalies (June 2019 or August 2019) equally affected all the sectors. Sector 5 is the only one that seems a bit different than the rest, with a lower mean value.

The time series plots for the rest of the sectors (from 7 to 10) show similar behaviour to the other sectors in the zone, with mean values near 0.5 and affected by the same anomalies. Sector 7 has higher occupancy compared to the others, possibly because it is nearest to the hospitals.

Statistics

The sectors in Riyadh present different characteristics than those of the previous scenarios, although they all behave more or less equally. Quantile and descriptive statistics for the first six sectors are presented in Table 3.16. None of the sectors reaches 0 or 1,

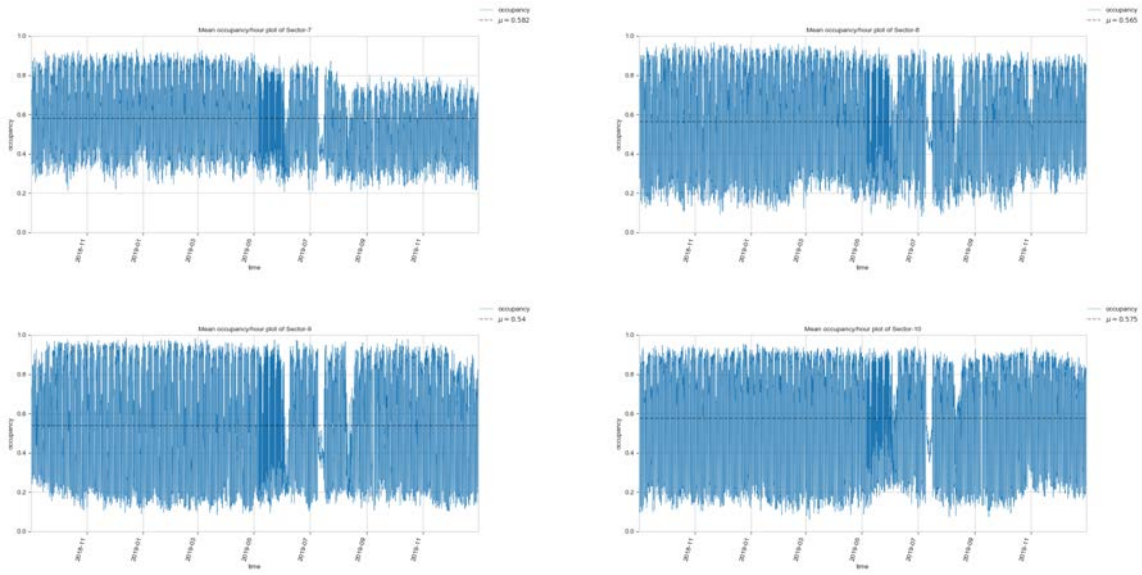


Figure 3.84: Riyadh data from the time interval 2018 and 2019. From top left to bottom right: sector 7 to sector 10.

indicating that enough vehicles find a parking spot and that there are enough places to satisfy this demand. The means and medians are more or less situated at the 0.5 value, and all sectors present negative kurtosis and positive skewness. The sectors with the most sensors (sector 1 and sector 3) are those with higher variance values.

Density plots (Figure 3.85) present the common shape of two peaks, similarly to other scenarios. This could be indicative of a distinction between weekdays and weekends or between daytime and nighttime. All sectors indicate lower occupancy levels in December.

Sectors 7, 8, 9 and 10 resemble the previous sectors: negative kurtosis (flat shape), means and medians slightly over 0.5, shapes with two peaks and low variability in general. For July, sectors 8 and 10 have three peaks instead of the common two, while for December, all of the sectors behave differently from sectors 1 to 6. For detailed information, check Table 3.17 and Figure 3.86.

Monthly and weekly data analysis

Figure 3.87 shows all sector time series for March 2019. Daily patterns can be seen easily with slight differences between weekdays and weekends. Sector 2 presents a lower range of values compared to the others. Sector 4 has less occupancy on Saturdays compared to other days (note that Friday and Saturday are the official days of rest in Saudi Arabia). Occupancy levels start to increase at 6 a.m. every day and start to decrease at 18 p.m.

All sectors in Riyadh seem to follow the same pattern on weekdays: occupancy/hour rises at 6 a.m. and drops at 18 p.m., but on weekends the sectors behave differently (see Figure 3.88). The day with major differences from the others is Friday: occupancy

Quantile statistics	Sector 1	Sector 2
Minimum	0.046	0.133
5-th percentile	0.105	0.226
Q1	0.119	0.304
Median	0.549	0.489
Q3	0.832	0.678
95-th percentile	0.935	0.812
Maximum	0.976	0.865
Range	0.930	0.732
Interquartile range (IQR)	0.653	0.374

Descriptive statistics	Sector 1	Sector 2
Standard deviation	0.316	0.2
Coefficient of variation	0.608	0.401
Kurtosis	-1.650	-1.338
Mean	0.519	0.499
Median Absolute Deviation	0.293	0.177
Skewness	-0.036	0.16
Variance	0.1	0.04

Quantile statistics	Sector 3	Sector 4
Minimum	0.091	0.123
5-th percentile	0.149	0.212
Q1	0.202	0.323
Median	0.414	0.482
Q3	0.738	0.790
95-th percentile	0.922	0.937
Maximum	0.984	0.975
Range	0.893	0.852
Interquartile range (IQR)	0.535	0.468

Descriptive statistics	Sector 3	Sector 4
Standard deviation	0.28	0.253
Coefficient of variation	0.589	0.463
Kurtosis	-1.447	-1.417
Mean	0.475	0.547
Median Absolute Deviation	0.255	0.229
Skewness	0.309	0.225
Variance	0.078	0.064

Quantile statistics	Sector 5	Sector 6
Minimum	0.117	0.120
5-th percentile	0.203	0.213
Q1	0.278	0.288
Median	0.395	0.517
Q3	0.599	0.792
95-th percentile	0.757	0.872
Maximum	0.798	0.918
Range	0.680	0.798
Interquartile range (IQR)	0.321	0.504

Descriptive statistics	Sector 5	Sector 6
Standard deviation	0.186	0.244
Coefficient of variation	0.421	0.457
Kurtosis	-1.161	-1.578
Mean	0.441	0.534
Median Absolute Deviation	0.162	0.223
Skewness	0.404	0.072
Variance	0.034	0.06

Table 3.16: Statistics for Riyadh sectors 1 to 6.

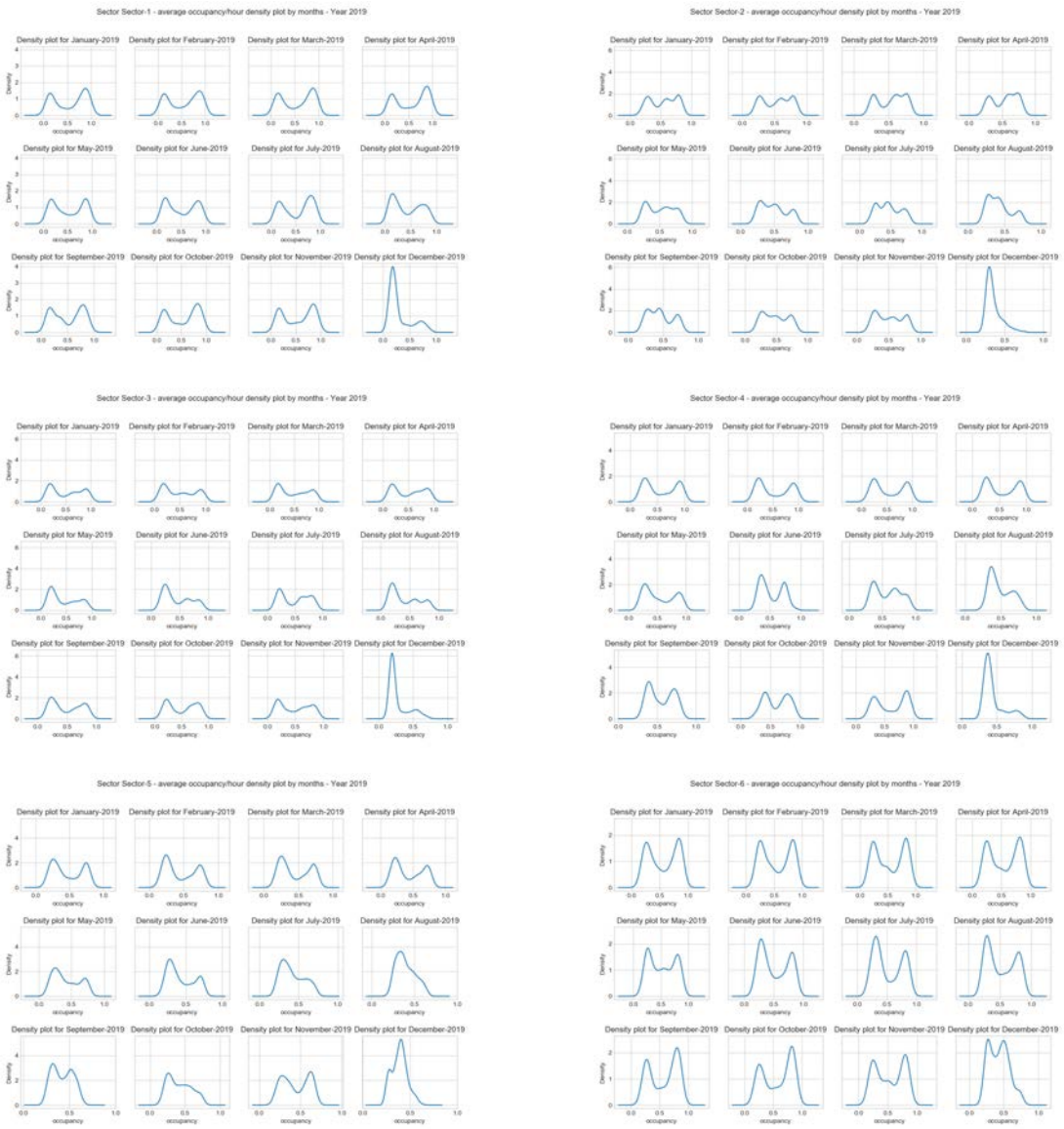


Figure 3.85: Density plots for the first 6 sectors in Riyadh.

Quantile statistics	Sector 7	Sector 8
Minimum	0.206	0.082
5-th percentile	0.296	0..178
Q1	0.39	0.290
Median	0.6	0.604
Q3	0.755	0.829
95-th percentile	0.869	0.905
Maximum	0.935	0.971
Range	0.73	0.889
Interquartile range (IQR)	0.364	0.539

Descriptive statistics	Sector 7	Sector 8
Standard deviation	0.196	0.267
Coefficient of variation	0.337	0.473
Kurtosis	-1.37	-1.563
Mean	0.581	0.565
Median Absolute Deviation	0.174	0.245
Skewness	-0.03	-0.146
Variance	0.038	0.071

Quantile statistics	Sector 9	Sector 10
Minimum	0.083	0.060
5-th percentile	0.145	0.152
Q1	0.225	0.251
Median	0.545	0.668
Q3	0.856	0.85
95-th percentile	0.937	0.907
Maximum	0.985	0.956
Range	0.902	0.895
Interquartile range (IQR)	0.631	0.599

Descriptive statistics	Sector 9	Sector 10
Standard deviation	0.3	0.286
Coefficient of variation	0.556	0.497
Kurtosis	-1.644	-1.562
Mean	0.540	0.574
Median Absolute Deviation	0.278	0.263
Skewness	0.011	-0.302
Variance	0.090	0.082

Table 3.17: Statistics for Riyadh sectors 7 to 10.

Figure 3.86: Density plots for sectors 7, 8, 9 and 10 in Riyadh..

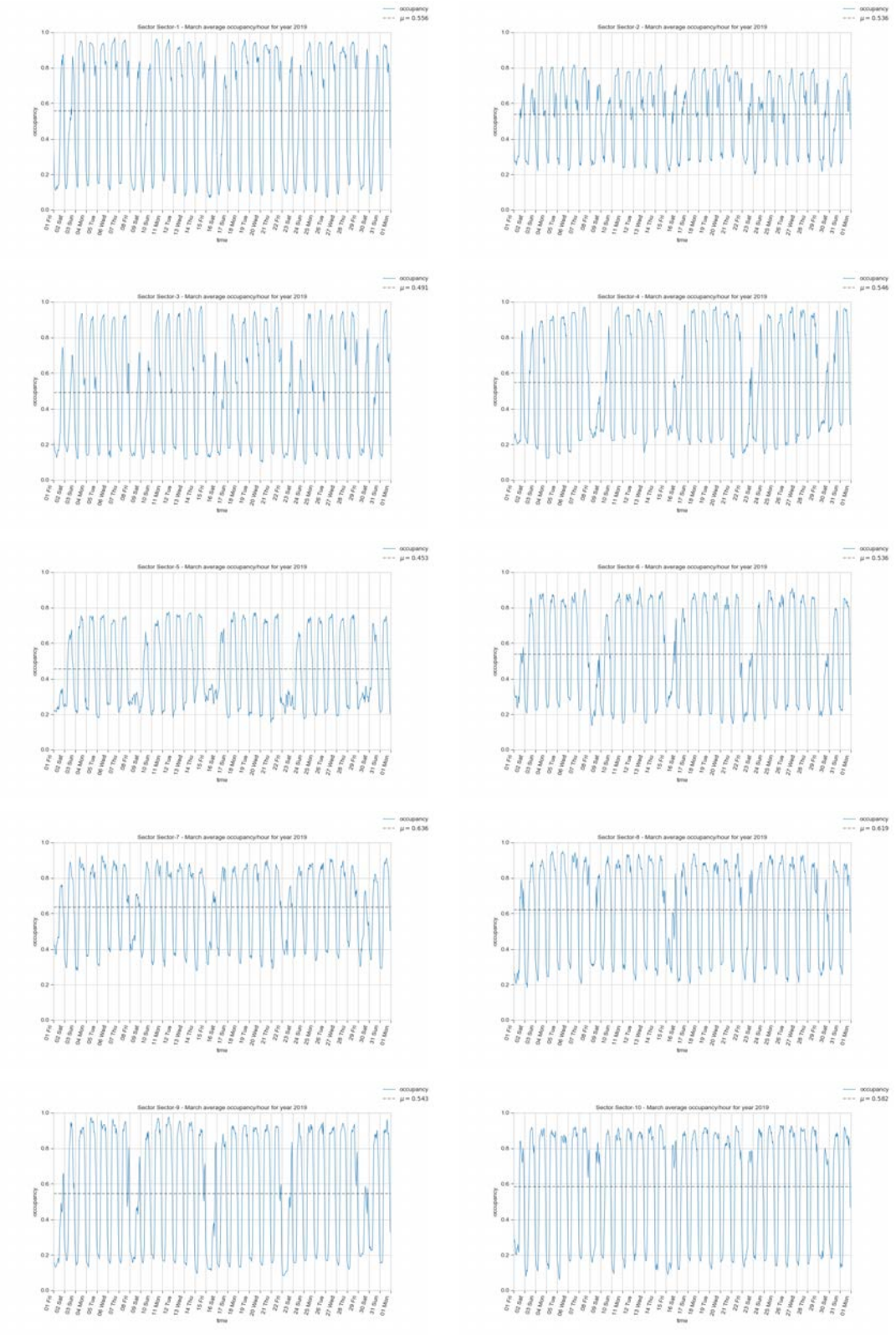


Figure 3.87: Time series for Riyadh sectors in March 2019. From top left to top right: sector 1 to sector 10.

levels are lower and usually rise between 10 a.m. and 12 a.m. (depending on the sector), then decrease at night (12 a.m.).

The smoothed time series for sector 1 can be seen in Figure 3.89. Similarly to other scenarios (e.g., Figure 3.48), the moving average with a window size of 12 perfectly shows the daily pattern and the differences between weekdays and weekends.

Figure 3.90 shows the behaviour of parking occupancy/hour at different levels of aggregation (day, month, week and year) for sector 1. The daily aggregation shows how different the behaviour is on weekdays compared to weekends: in the former, occupancy levels rise at 6 a.m.; while in the latter they rise slowly from 10 a.m. onward.

The monthly aggregation shows that all months behave similarly, except that May has a strange decrease in occupancy at 18 p.m. The occupancy for June, July and August are lower than the other months. December occupancy behaves like Saturday in the daily seasonal plot, possibly indicating holidays and that the area is used for leisure at night.

Weeks behave similarly to each other, which also happens with the yearly plot.

Figure 3.91 shows similar patterns to the sector 2 seasonal plots. The daily patterns show that weekends in the area are used for leisure (perhaps because of restaurants in the area).

Monthly patterns show that December and August are the months with the lowest occupancy.

Figure 3.92 shows the seasonal plot for sector 3. Daily aggregation shows how closely related Thursday and Friday are at night, possibly indicating that people use this area for leisure at night on the weekends. Monthly aggregation shows that December has lower levels of occupancy/hour. The occupancy levels for year 2019 are lower than those for 2018.

Figure 3.93 shows the sector 4 seasonal plots, which are not very different from the other sectors already mentioned.

The plots in Figure 3.94 show the seasonal patterns for sector 5. The daily aggregation differs from the previously mentioned sectors in the sense that Saturday behaves more like the other days of the week, while Friday levels do not rise very much during the day. The monthly seasonal plot shows less occupancy in August, and the decrease in May at 18 p.m. is the same as that of sector 2.

Seasonal plots for sector 6 (Figure 3.95) show exactly how sector occupancy behaves Monday to Wednesday. On Thursday, occupancy starts to decrease earlier than the previous days, and Friday has its own behaviour. Saturday behaves between Friday and the first days of the week.

The monthly aggregation (second plot in Figure 3.95) shows patterns that are similar to the previous sectors: a decrease in May starting at 15 p.m., and less occupancy in December and August. The rest of the months behave more or less equally.

No distinctions can be made in the weekly and yearly seasonal plots, as they offer no new information compared to the other sectors.

Figure 3.88: Time series for the Riyadh sectors in March 2019, week 2. From top left to top right: sector 1 to sector 10.

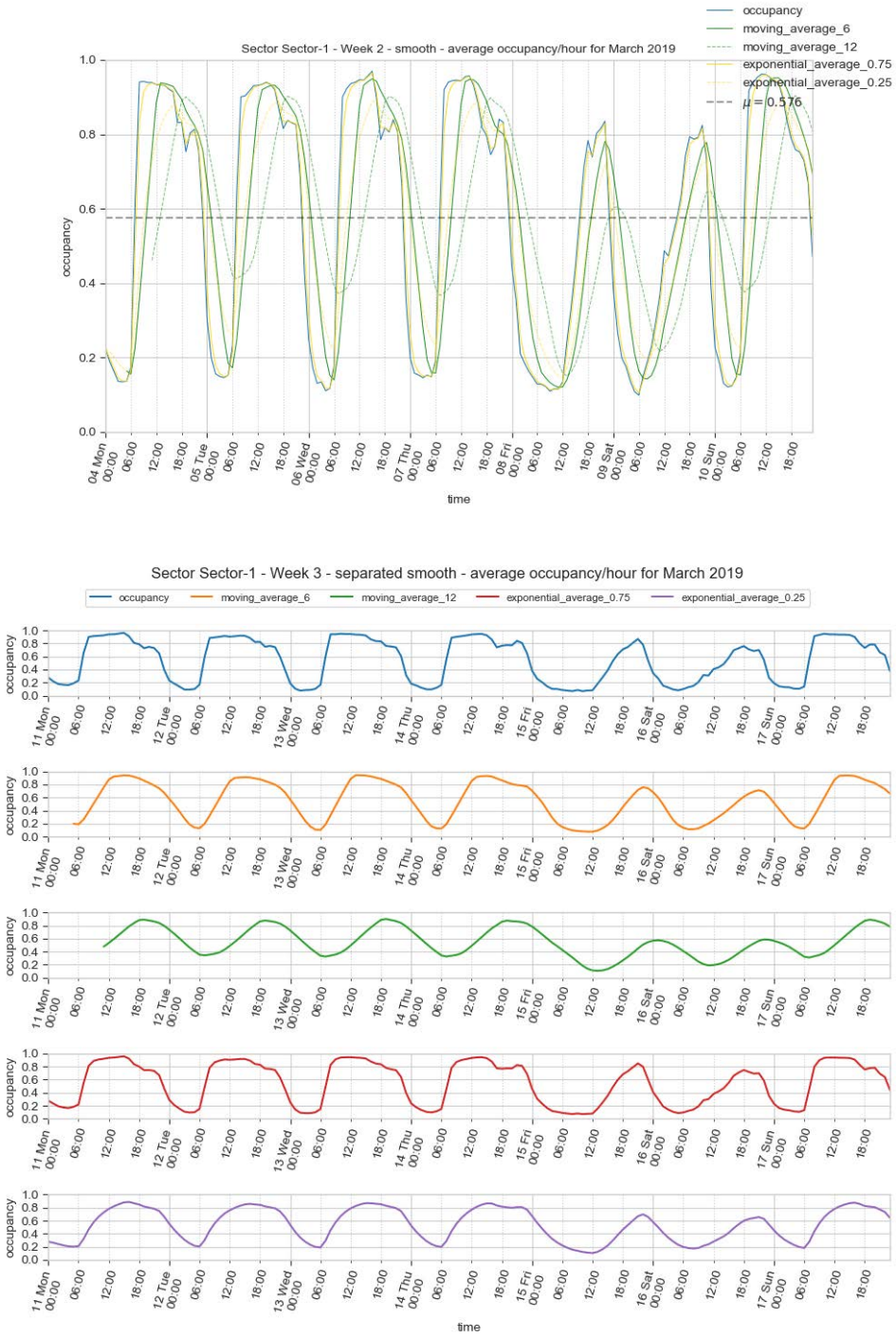


Figure 3.89: Smoothed time series for sector 1 in Riyadh. Methods used for smoothing the series are moving average and exponential smoothing. The top visualization shows all time series in the same plot, where green lines correspond to moving averages with window sizes of 6 and 12 hours; yellow lines pertain to the exponential smoothing average, with alpha values of 0.75 and 0.25. The visualization at the bottom shows the time series separately.

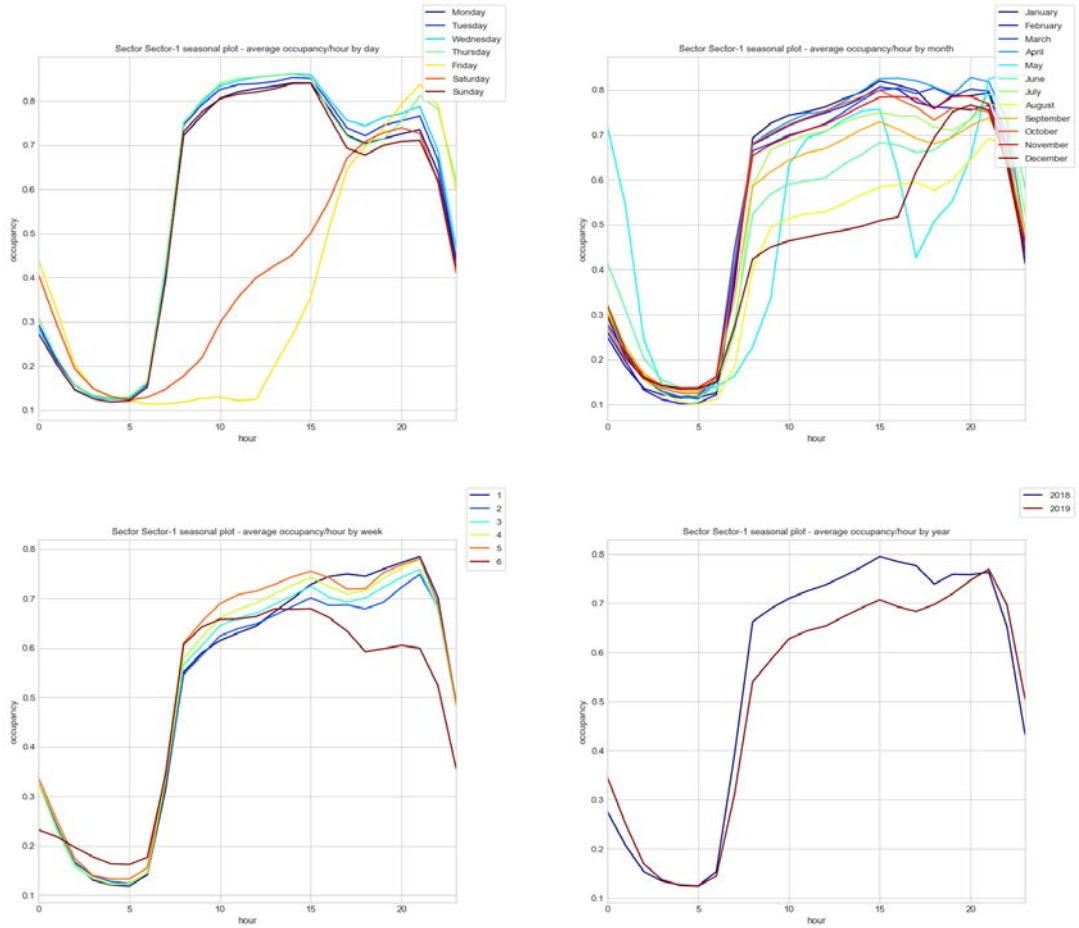


Figure 3.90: Occupancy/hour seasonal plots for sector 1 in Riyadh.

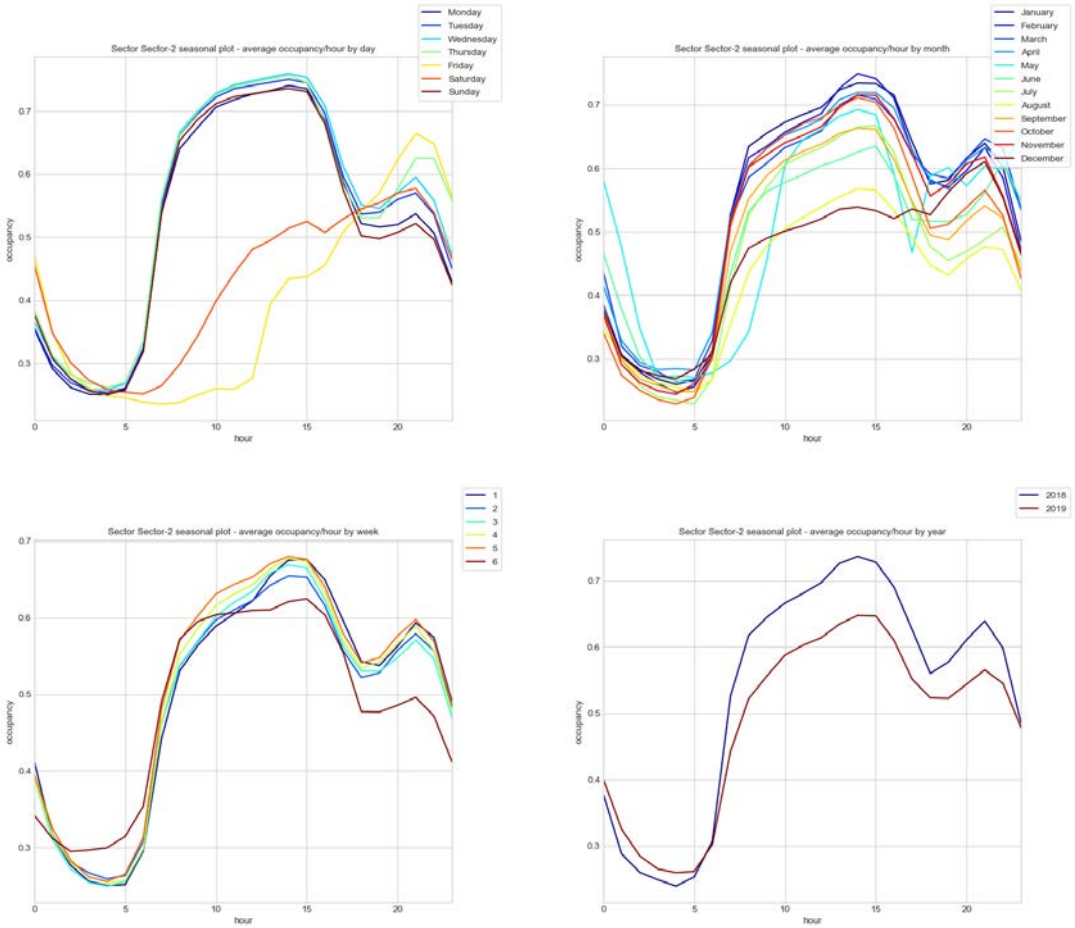


Figure 3.91: Occupancy/hour seasonal plots for sector 2 in Riyadh.

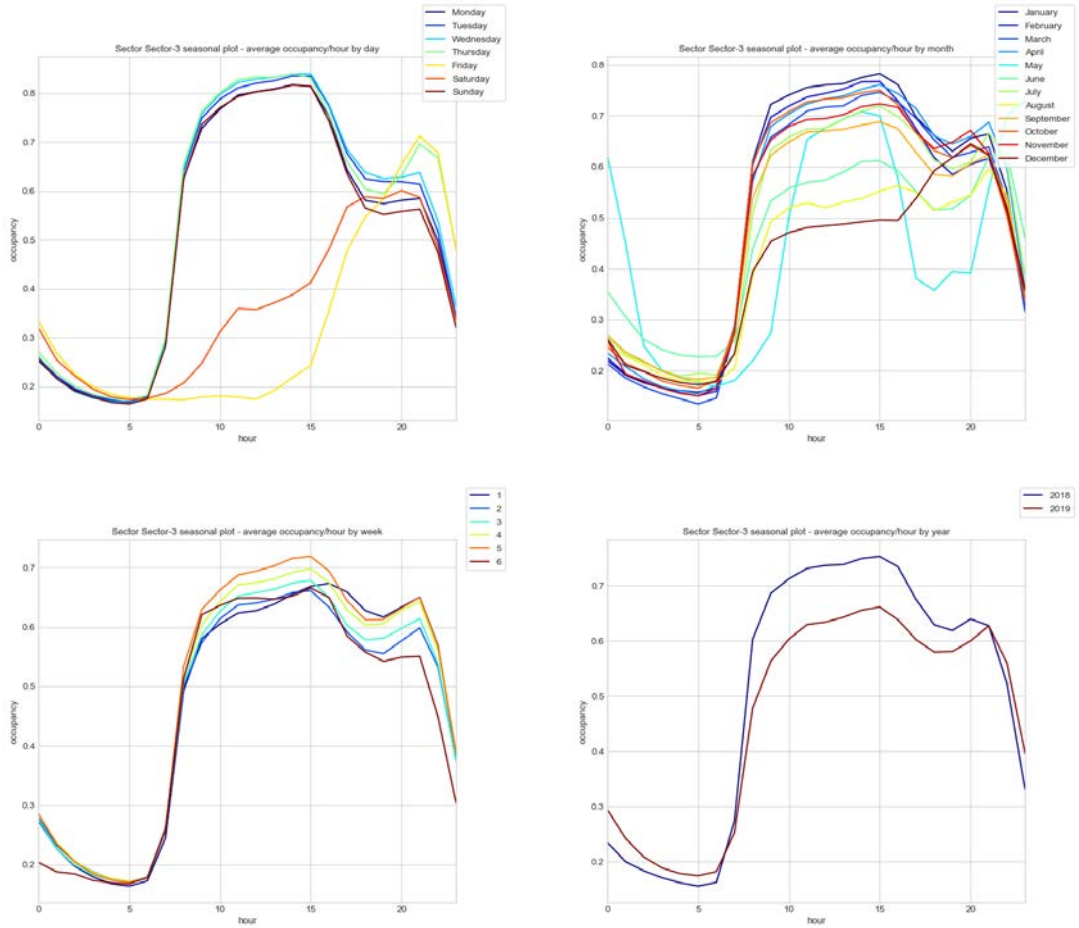


Figure 3.92: Occupancy/hour seasonal plots for sector 3 in Riyadh.

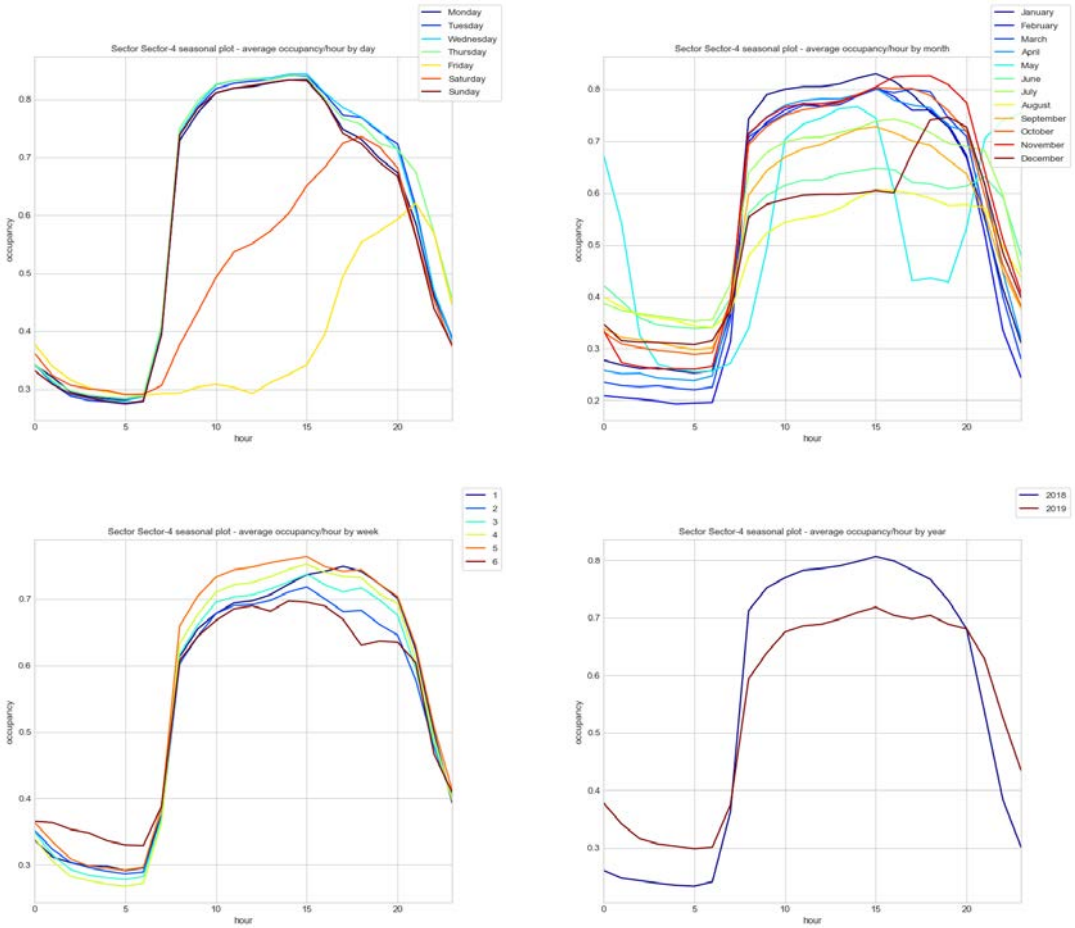


Figure 3.93: Occupancy/hour seasonal plots for sector 3 in Riyadh.

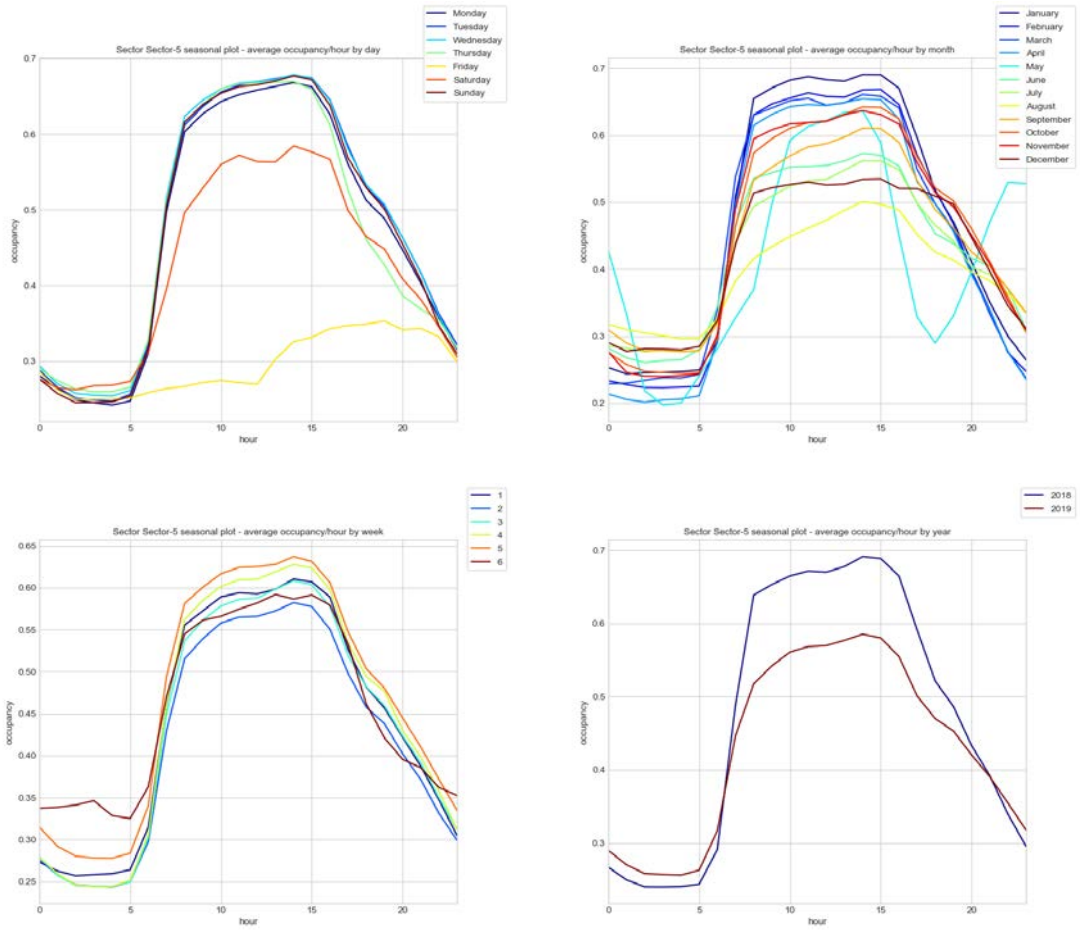


Figure 3.94: Seasonal plots for sector 5 in Riyadh at different levels of aggregation (daily, monthly, weekly and yearly)

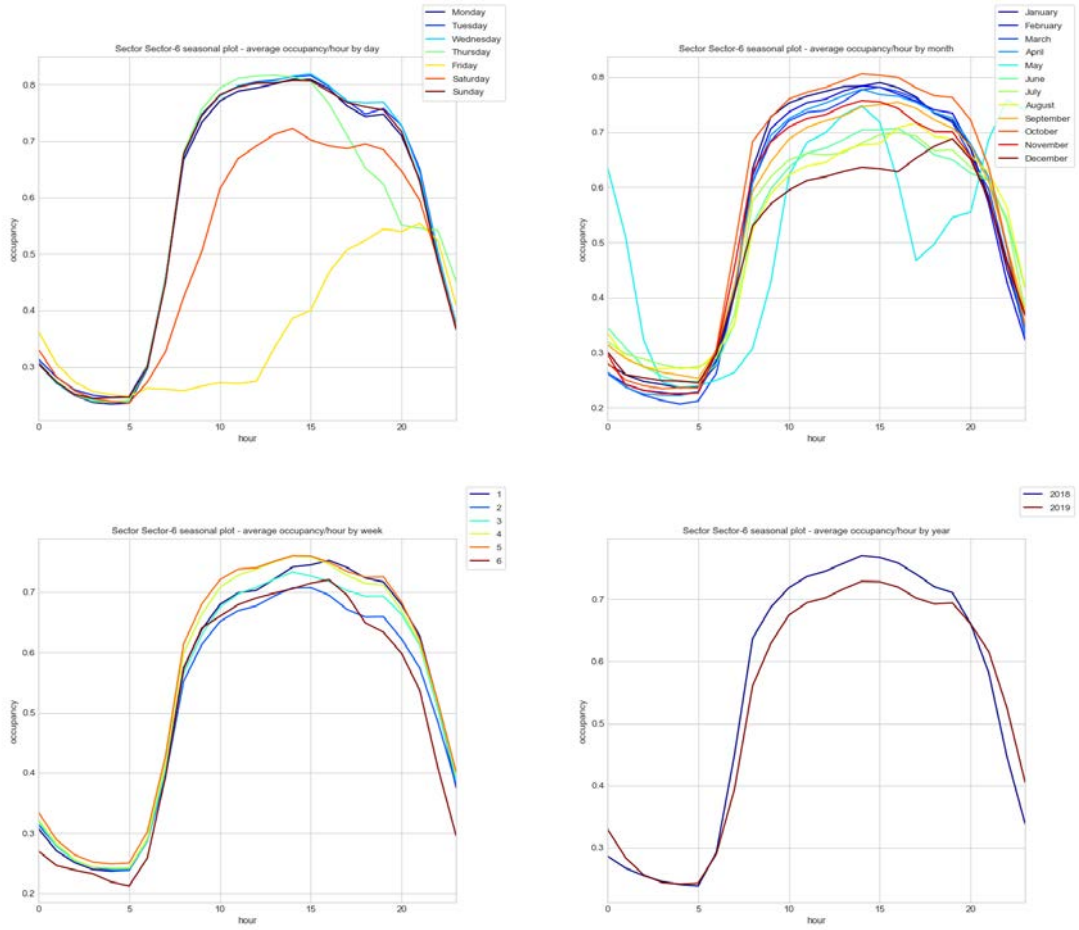


Figure 3.95: Visualizations of seasonal plots at different periodic aggregations for sector 6 in Riyadh. From top left to bottom right: daily, monthly, weekly and yearly aggregations.

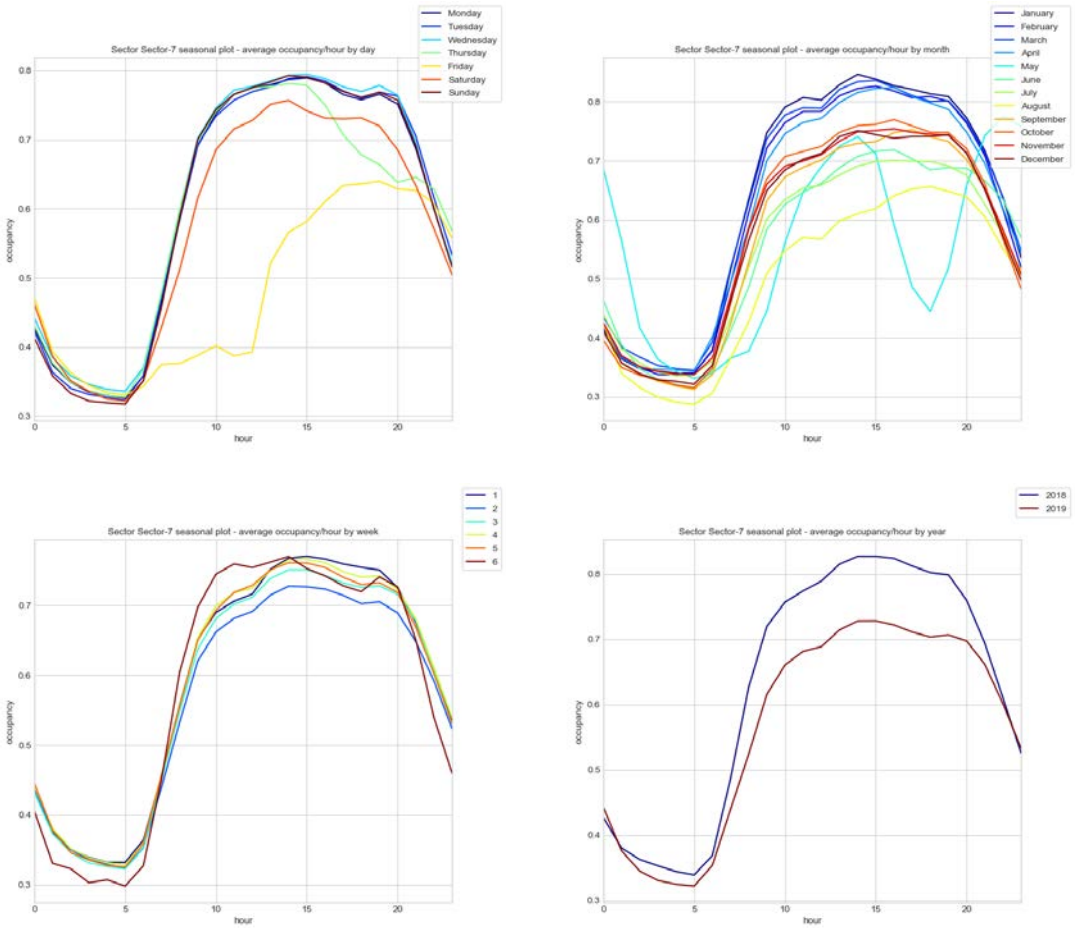


Figure 3.96: Riyadh sector 7 seasonal plots at different periodic aggregations: daily, monthly, weekly and yearly.

For sector 7, the daily patterns in the first seasonal plot of Figure 3.96 show that Friday is the only day that the occupancy/hour pattern is different from the other days. This time, Saturday behaves more like the earlier days of the week.

In the monthly seasonal plot, August is the one that presents less occupancy/hour levels. It may be that a strange pattern persists.

The weekly and yearly plots do not offer any relevant information that has not been previously mentioned.

For sector 8, the daily seasonal plots (first plot in Figure 3.97) show how Friday peaks at night. This could be explained by the presence of some restaurant in the sector.

In the monthly data, May's behaviour is also indicated here (as with the other sectors), and the month with the least occupancy is December (as in sector 7).

The weekly and yearly seasonal plots do not offer any new information.

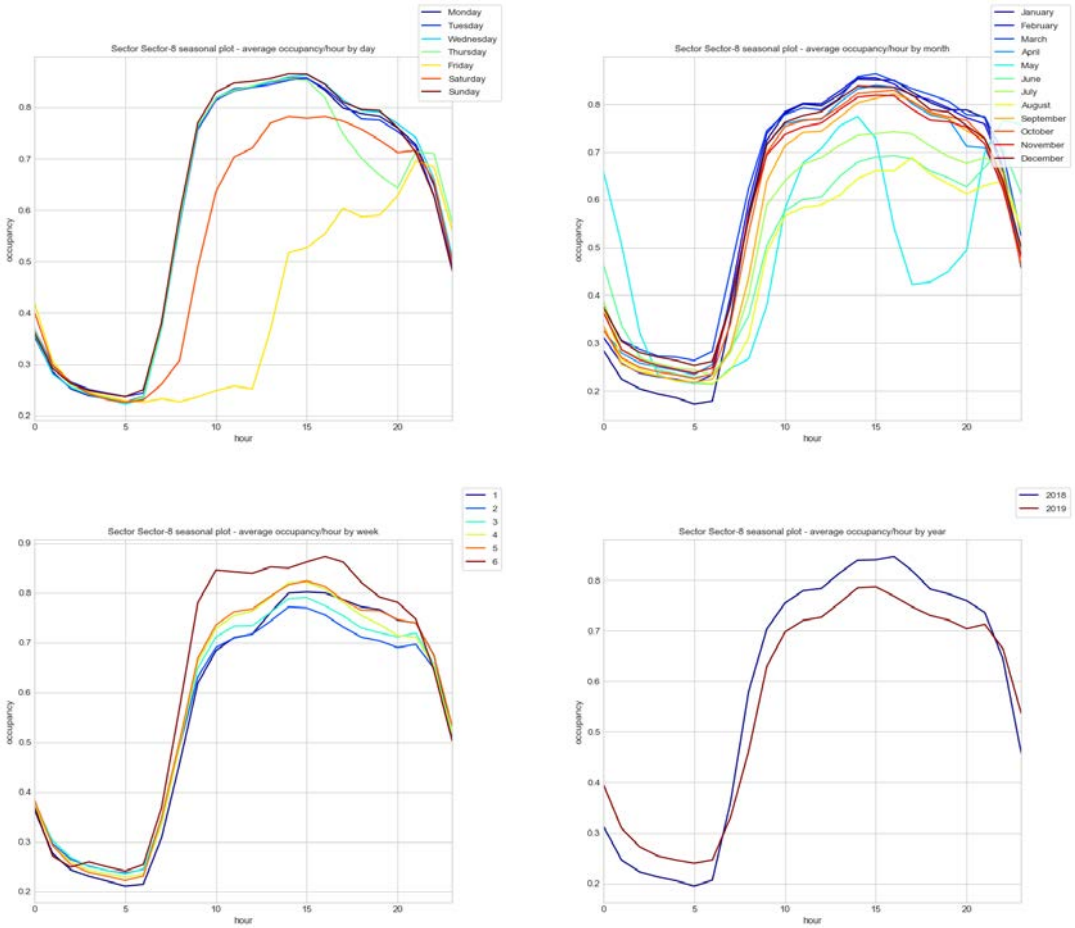


Figure 3.97: Seasonal plots for sector 8 in Riyadh.

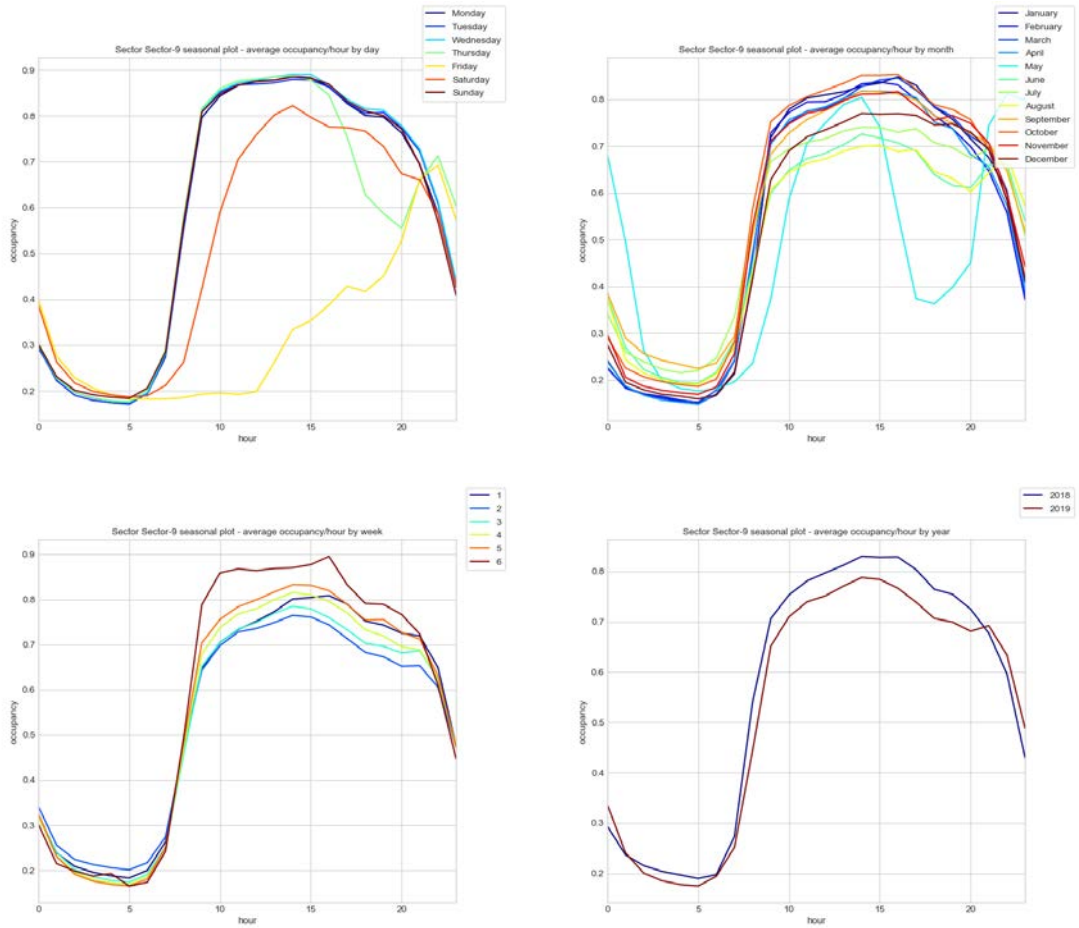


Figure 3.98: From top left to bottom right: seasonal plots of daily, monthly, weekly and yearly aggregations for sector 9 in Riyadh.

Sector 9 behaves similarly to sector 8 in the seasonal plots (see Figure 3.98). This could be explained by the same fact as above: both sectors are surrounded by restaurants, while sector 9 is additionally surrounded by shops.

The last sector, sector 10, presents slightly different behaviour than sector 8 and sector 9. The daily plot (first plot in Figure 3.99) shows different behaviour on Friday, with occupancy/levels starting to rise at 12 a.m. One explanation for this different behaviour could be that sector 10 is positioned on a major street (Musa Ibn Nusair street), whereas the other sectors are not.

Monthly data indicates less occupancy for June, July and August while maintaining the strange pattern of May.

Weekly and yearly data adds no new information.

Some last words on seasonal plots: it may be that the patterns for May in all sectors indicate that people in the zone are working part-time during that month and returning

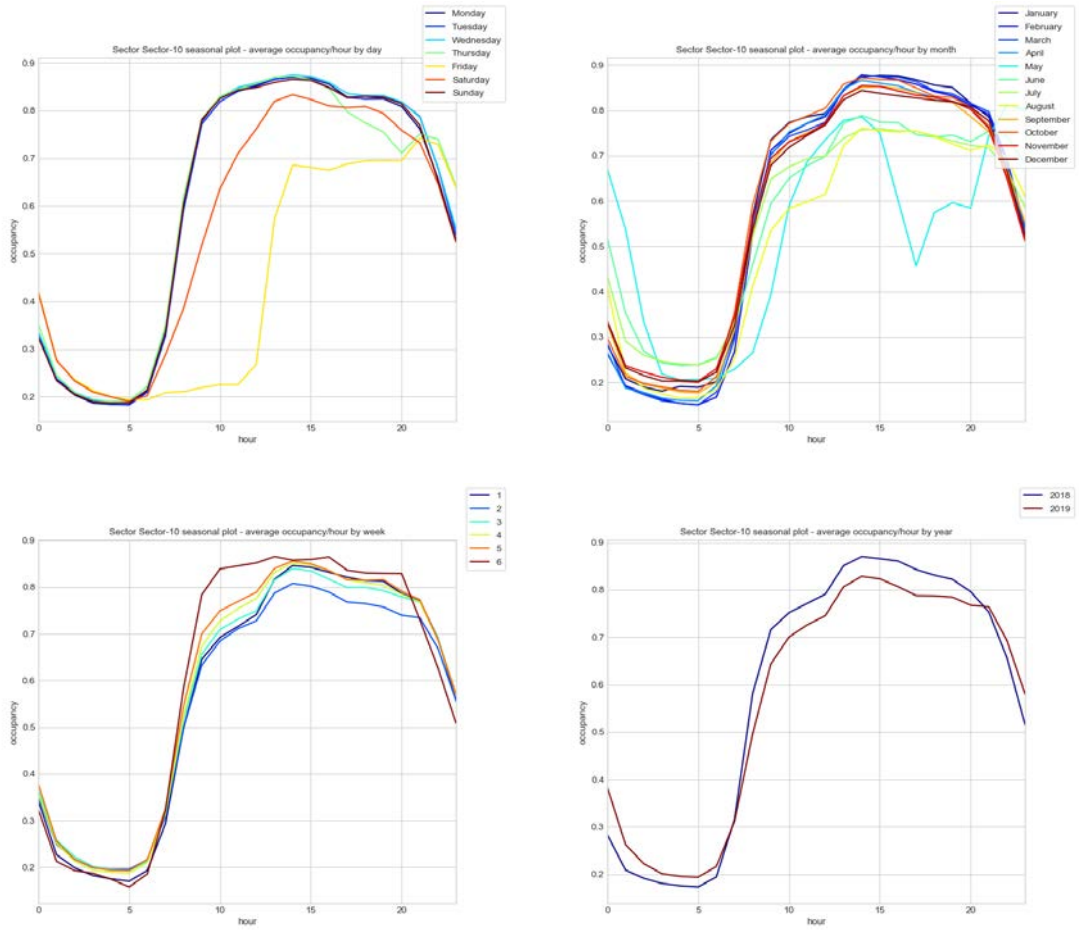


Figure 3.99: Occupancy/hour seasonal plots at different period aggregations for sector 10 in Riyadh. The periods considered are day, month, week and year.

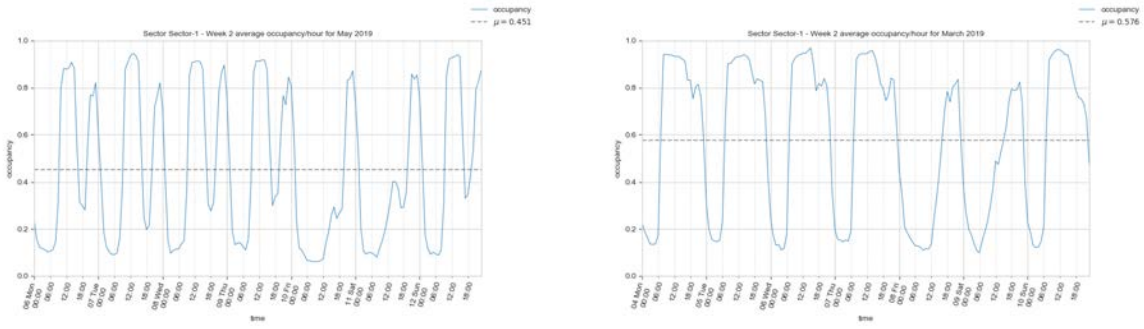


Figure 3.100: Sector 1: May (left) vs. March (right) occupancy/hour behaviour.

home earlier, while at night the area is visited by other people for leisure. Figure 3.100 shows a comparison of the occupancy/hour levels between May and March, where we can clearly see how different the working day is. This change in behaviour could be due to the fact that 5 May 2019 to June 3 2019 was the Ramadan period.

Specialized time series plots

Box plot visualizations for sectors 1 to 4 in Figure 3.101 show how similarly sectors behave until May. In August, the occupancy levels decrease (clearly an effect of the holidays). The change in behaviour between December 2018 and December 2019 is surprising. It may be that parking is affected by some infrastructure works in the zone. Clearly, December 2019 is an anomaly.

Box plots for sectors 5, 6, 7 and 8 (Figure 3.102) show slightly different patterns when compared to previous sectors (Figure 3.101). From July 2019 onwards, there is less occupancy, but December 2019 is not affected in the same way as previous sectors. In fact, sector 8 behaves more or less the same as December 2018.

Box plots for sectors 9 and 10 (Figure 3.103) show that these sectors behave similarly to sector 8 (see the fourth plot in Figure 3.102). This could be an indication that sectors 1 to 6 (north zone) are affected by some work in the zone that is not affecting the southern sectors. Overall, the only months with less occupancy are from May (Ramadan) to August, indicative of summer time.

After decomposing the series on trend and seasonality, some new patterns arise and we can also confirm some assumptions from previous plots. Figure 3.104 presents the decomposition of sectors 1, 2 and 3. The top row indicates the data for the year 2019, in which the trend is similar for all three sectors: they remain constant except May (Ramadan) and August (summer holidays). All three sectors have very low occupancy at the start of December 2019 and throughout the whole month. Checking the residuals, they are low, except for the first week of June (end of Ramadan holidays), August (holidays), September (some national days) and December. So, we can assume that holidays have great influence on the behaviour of parking occupancy. This could possibly

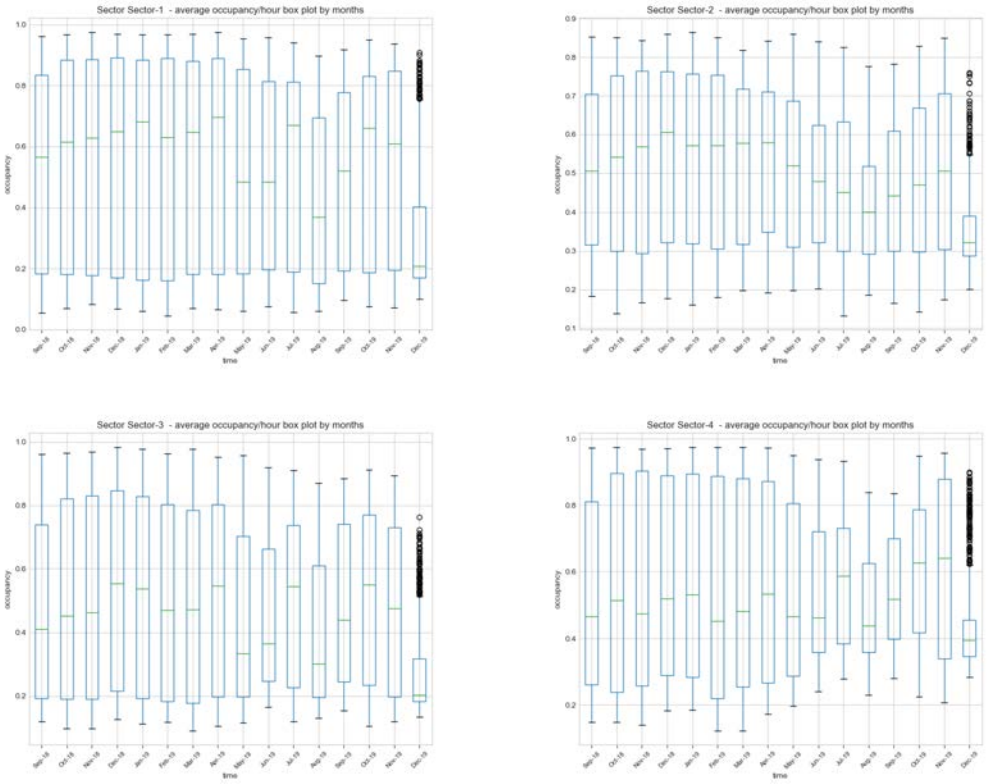


Figure 3.101: Box plots for sectors 1 (top left), 2 (top right), 3 (bottom left) and 4 (bottom right) in Riyadh.

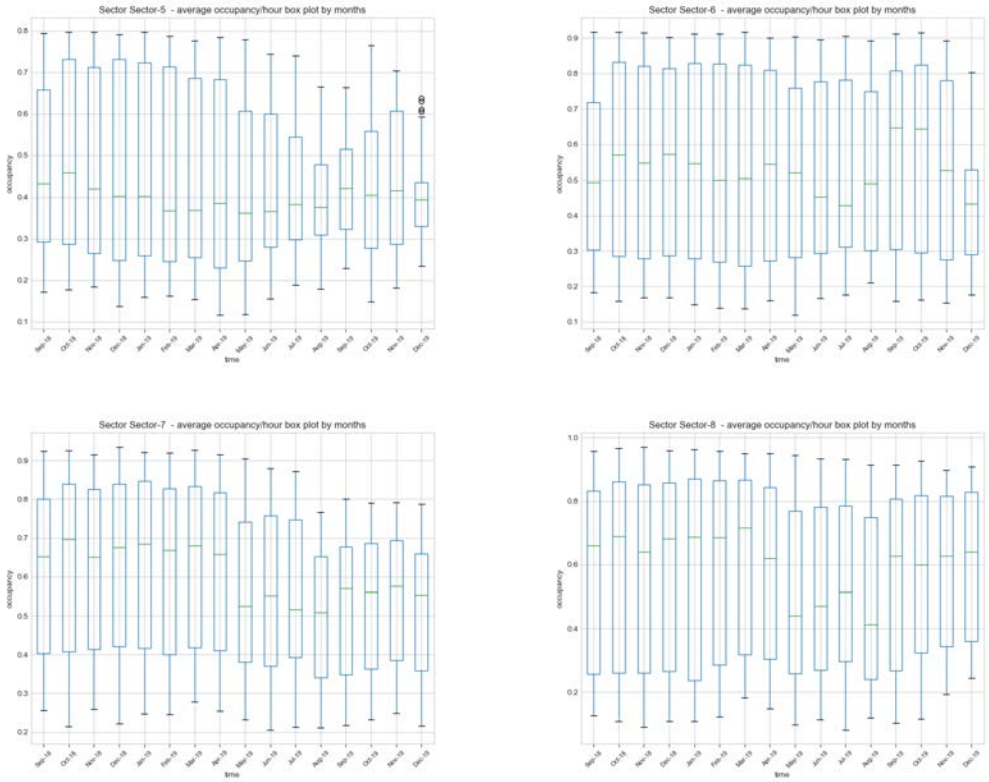


Figure 3.102: Box plots for sectors 5 (top left), 6 (top right), 7 (bottom left) and 8 (bottom right).

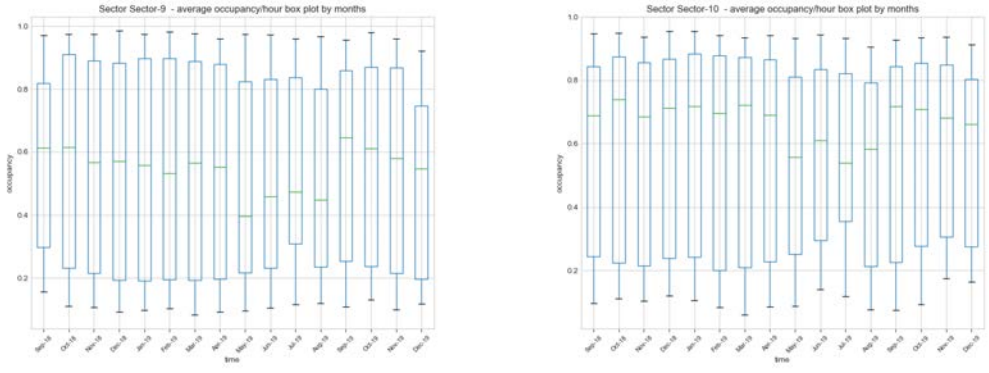


Figure 3.103: Box plots for sectors 9 (left) and 10 (right) in Riyadh.

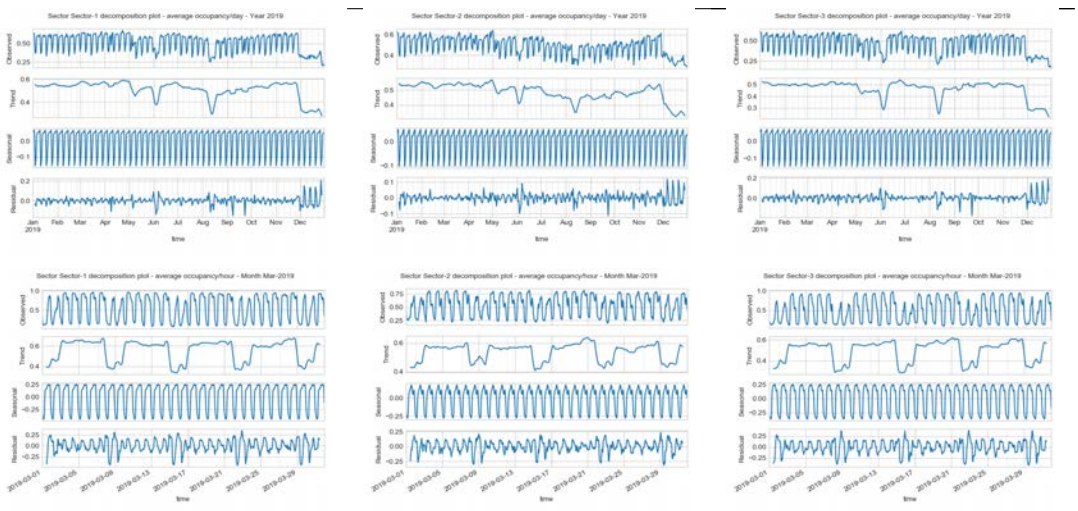


Figure 3.104: Decomposition of the Riyadh time series for sectors 1 (left), 2 (middle), and 3 (right). At top are the series from 2019, at bottom the series for March 2019.

explain the two peaks in the distributions, as the area is an office/leisure part of the city with very different behaviours, depending on the day (holiday vs. working day).

When focusing on the daily patterns and looking at the second row of Figure 3.104, where the time range is limited to March 2019, we can observe that the trend has a constant pattern like a square wave, with the top values corresponding to weekdays and the lowest values to the weekends. Residuals are notable on weekends (2 days), as the daily seasonality is lost.

Decomposition for sectors 4, 5, and 6 show patterns (Figure 3.105) that are similar to those above. We can easily spot changes in the patterns for June, mid-August and the end of September. These sectors also suffer from the December anomaly, but to a lesser extent compared to the aforementioned sectors. Also, sector 5 and sector 6 present an anomaly in July, which does not coincide with any known holiday.

The March decomposition for sector 4 (at the left of the bottom row in Figure 3.105) indicates an increasing trend throughout the week that declines on weekends. The other two sectors do not present the slowly increasing trend, but their occupancy/hour values also decline quickly on weekends. As with the preceding sectors, the residuals are larger for weekends, as seasonality is not followed on these days.

Sectors 8 and 9 present a trend throughout the year, which is constant and affected only by the festivities (Figure 3.106). They also have the same anomaly in July as sectors 5 and 6. It may be that the July anomaly is due to some works in this area. The trend levels become constant after August and decline a little at the end of December.

When focusing on the time range of March 2019, the trend continues to be similar to that of the preceding sectors, presenting a square wave form and large residual values on weekends.

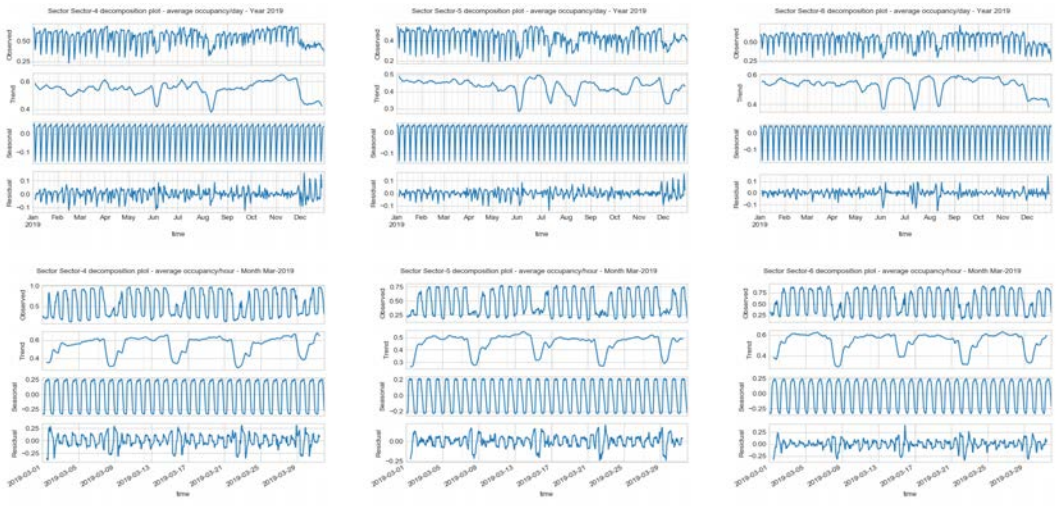


Figure 3.105: Decomposition of the Riyadh time series for sectors 4 (left), 5 (middle), and 6 (right). At top are the series for 2019, at bottom the series for March 2019.

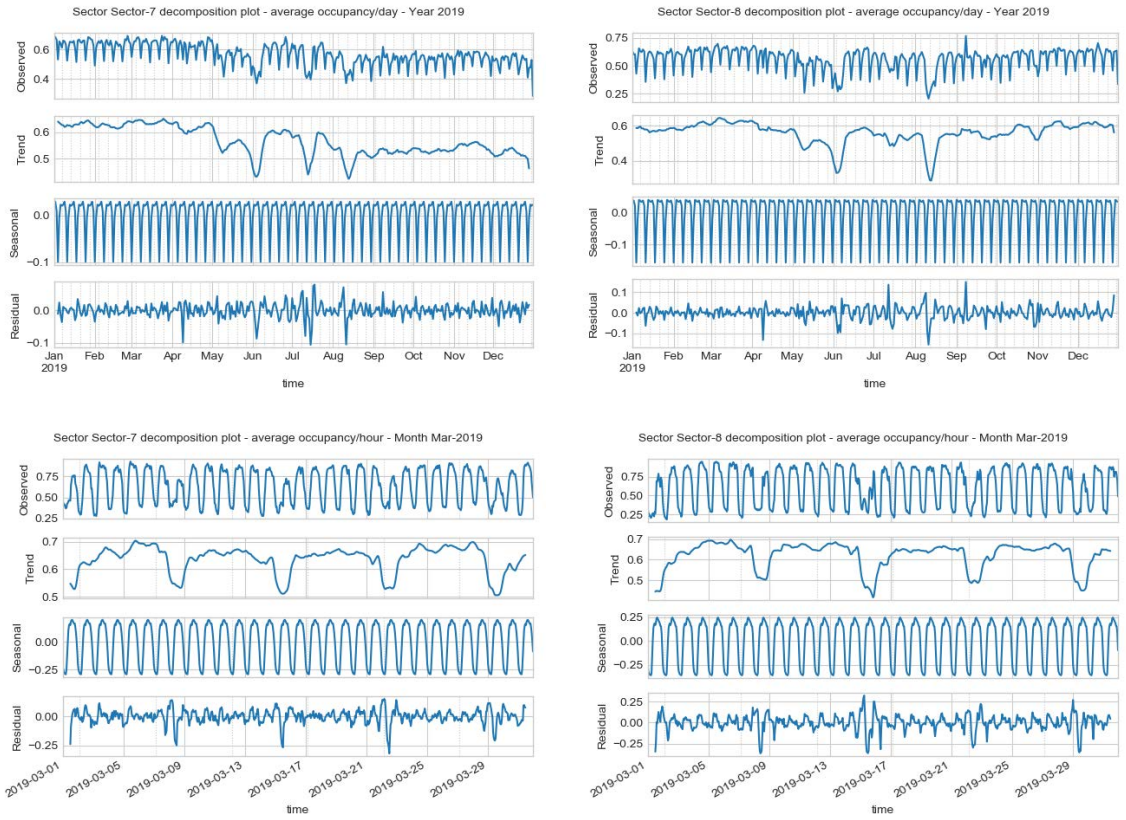


Figure 3.106: Decomposition of the Riyadh time series for sectors 7 (left) and 8 (right). At top are the series from 2019, at bottom the series for March 2019.

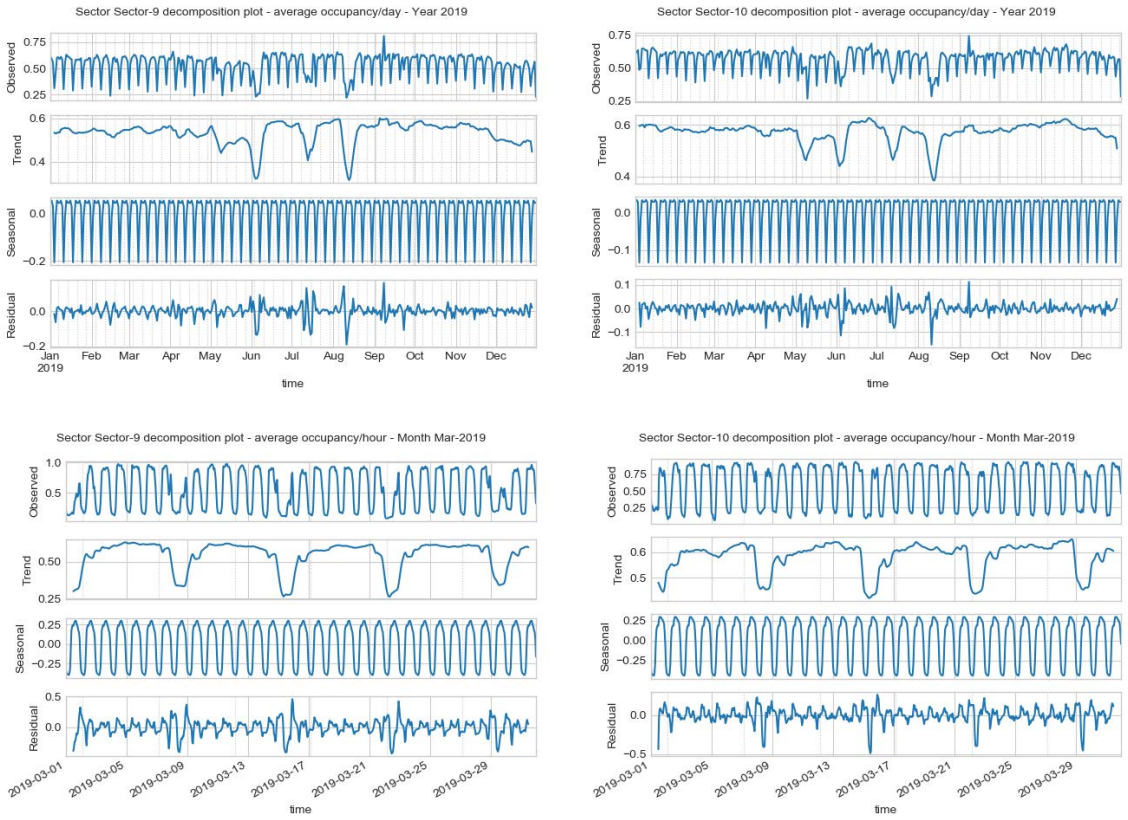


Figure 3.107: Decomposition of the Riyadh time series for sectors 9 (left) and 10 (right). At top are the series from 2019, at bottom the series for March 2019.

Finally, the trend levels for sectors 9 and 10 are the most constant of all the Riyadh sectors (see Figure 3.107). Both sectors present very similar behaviours, with large residuals on the holidays and in July. In contrast to the northern sectors, sectors 9 and 10 have no change in behaviour in December 2019.

Focusing on March 2019 (bottom row of Figure 3.107), the patterns are similar to the other sectors: high occupancy levels for 5 days then 1 or 2 days of low occupancy levels in which the residuals are large because trend + seasonality cannot capture this behaviour.

The ACF and PACF plots in Figure 3.108 show the autocorrelation and partial autocorrelation functions for Riyadh sectors 1 to 6. Similarly to other scenarios (e.g., Figures 3.77 and 3.53), there is a strong autoregressive pattern as the autocorrelation functions slowly tail off and the partial autocorrelation functions cut off at lag 2. Thus, the processes for all sectors seem to be AR, with important lags 1, 2 and 24 (or 25), where the last one indicating seasonality.

The ACF and PACF plots for sectors 7, 8, 9 and 10 (see Figure 3.109) have the same behaviour as the previous sectors.

Lag plots of the Riyadh sectors (see Figure FALTA) show a strong linear correlation in

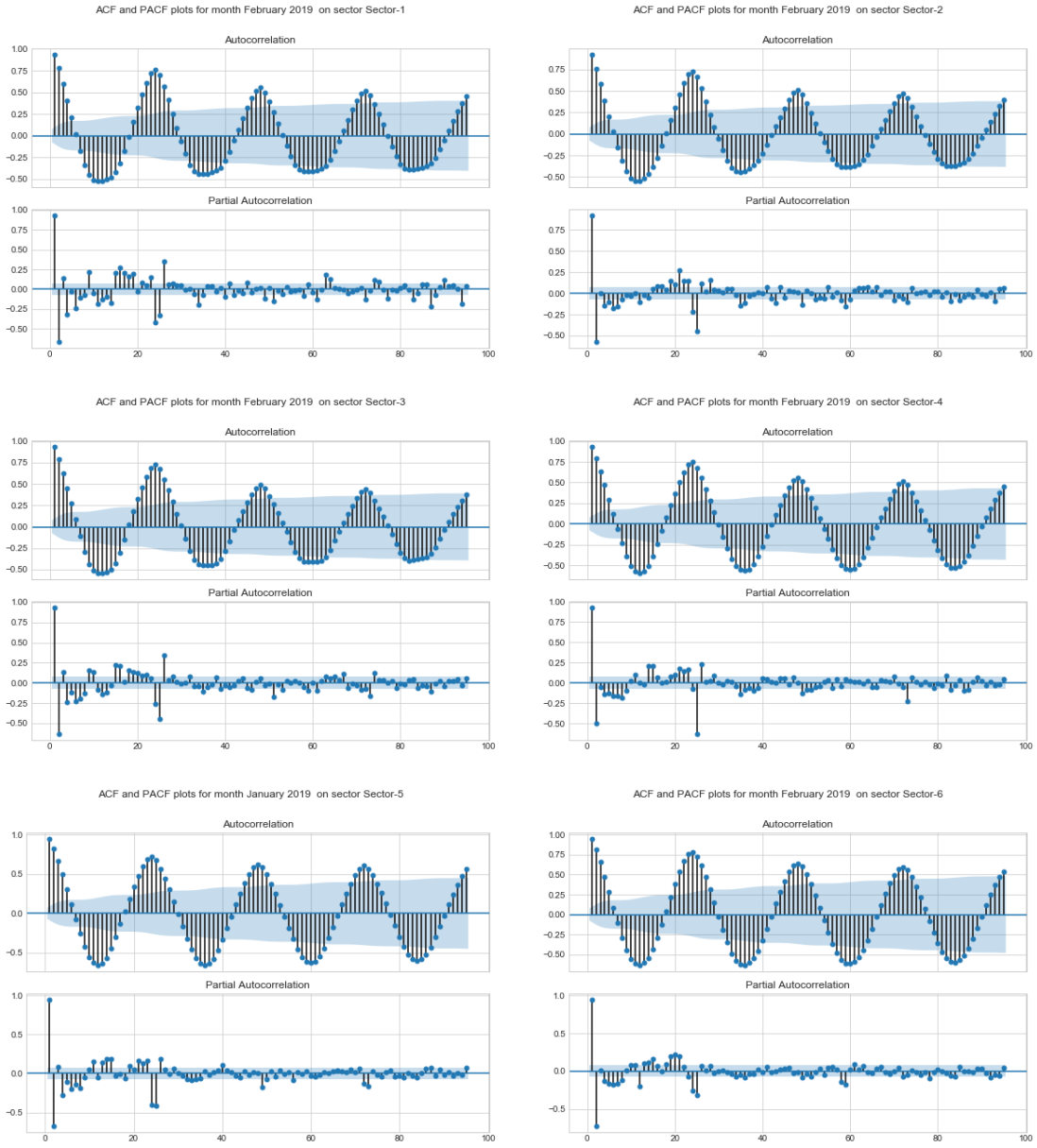


Figure 3.108: ACF and PACF plots for Riyadh sectors 1 to 6, February 2019. From top left to bottom right: sectors 1, 2, 3, 4, 5 and 6.

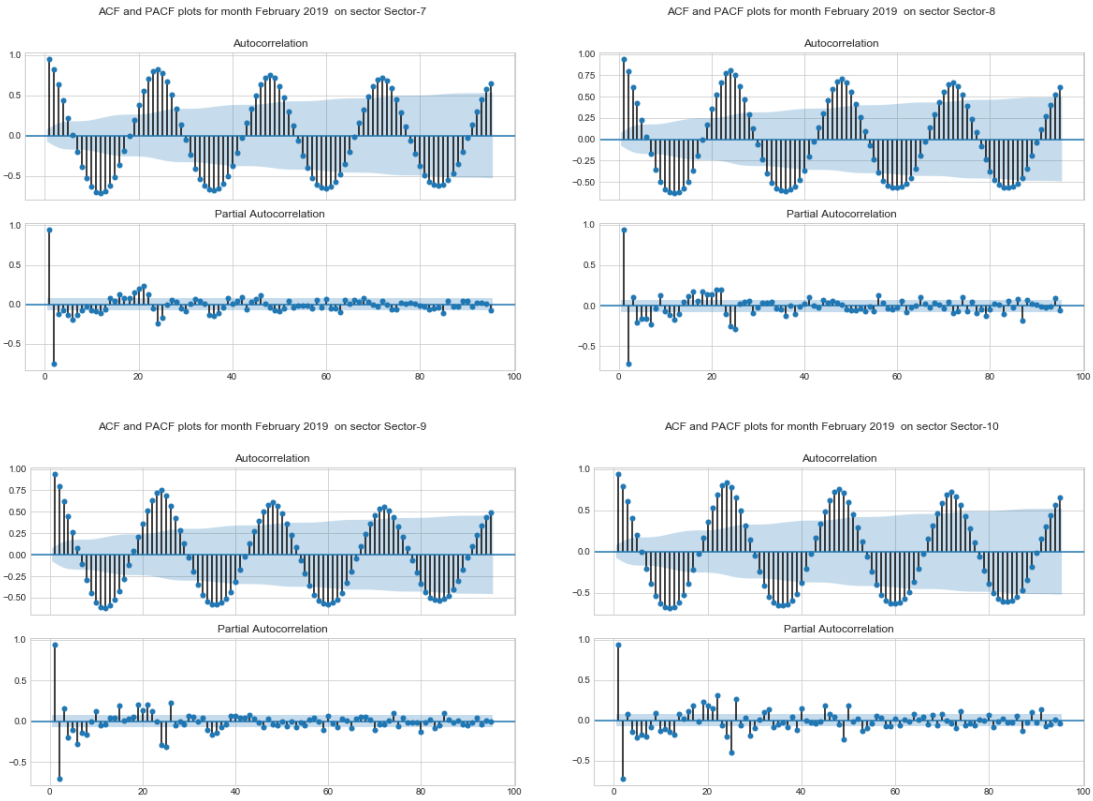


Figure 3.109: ACF and PACF plots for Riyadh sectors 7 to 10, February 2019. From top left to bottom right: sectors 7, 8, 9 and 10.

the first lag for all sectors.

All sectors show similar patterns in the frequency domain (see Figure 3.111). The predominant periods in the data are 24, 12, 84 and 168 (generally in that order). 24 and 168 denote the daily and weekly periods. The 12-hour period can be explained as the time between when occupancy/hour increases in the morning and decreases in the afternoon / at night. 84 is three and a half days, which constitute half of a week.

Time-space analysis

Taking a random week from the data, we can observe that all sectors behave similarly throughout the week (see Figure 3.112). It is only on weekends (days 01 and 02, that is, Friday and Saturday) that the sectors behave somewhat differently.

Cross-correlation visualizations confirm that all sectors have a strong linear dependency on all the other sectors for lags 0 to 5. The linear relationship fades as the sectors become farther away from each other. This can be observed in Figure 3.113, where sector 1 is more correlated in time with sector 2 and sector 3, and less correlated with sector 9 and sector 10. This points to a spatial correlation between the sectors and could be exploited to obtain better predictions.

3.6 Conclusions

At this point, it is necessary to summarize the different findings in the data.

In general, the larger the parking sector, the easier it is to detect patterns that repeat in time. We have seen that all parking sectors have a daily and/or weekly pattern in an autoregressive form (e.g., Figure 3.10). This information is of great value for creating a predictive model. Regrettably, sectors with low numbers of parking places have patterns that are harder to detect or their behaviour is chaotic (e.g., the Zipcar sector in Section 3.5.4).

It is interesting that most sectors have two behaviours, which are detected in a distribution with two peaks. These two behaviours usually occur as a result of different uses of the sectors throughout the days and weeks. One of the clearest examples is the city of Riyadh, where the sectors have totally different uses, depending on whether it is a holiday/weekend or workday/weekday (see Section 3.5.5 for detailed information).

Also, it is notable that sectors near to each other follow similar patterns, indicating they are slightly affected by their surroundings, and their occupancy levels are correlated in time (see Figure 3.113).

The analysis has identified some sectors with anomalies, which change the sector behaviours (e.g., the Willow Street sectors in 2019, check Section 3.5.4). Through simple modeling, such as trend + seasonality, these kinds of anomalies cannot be overcome.

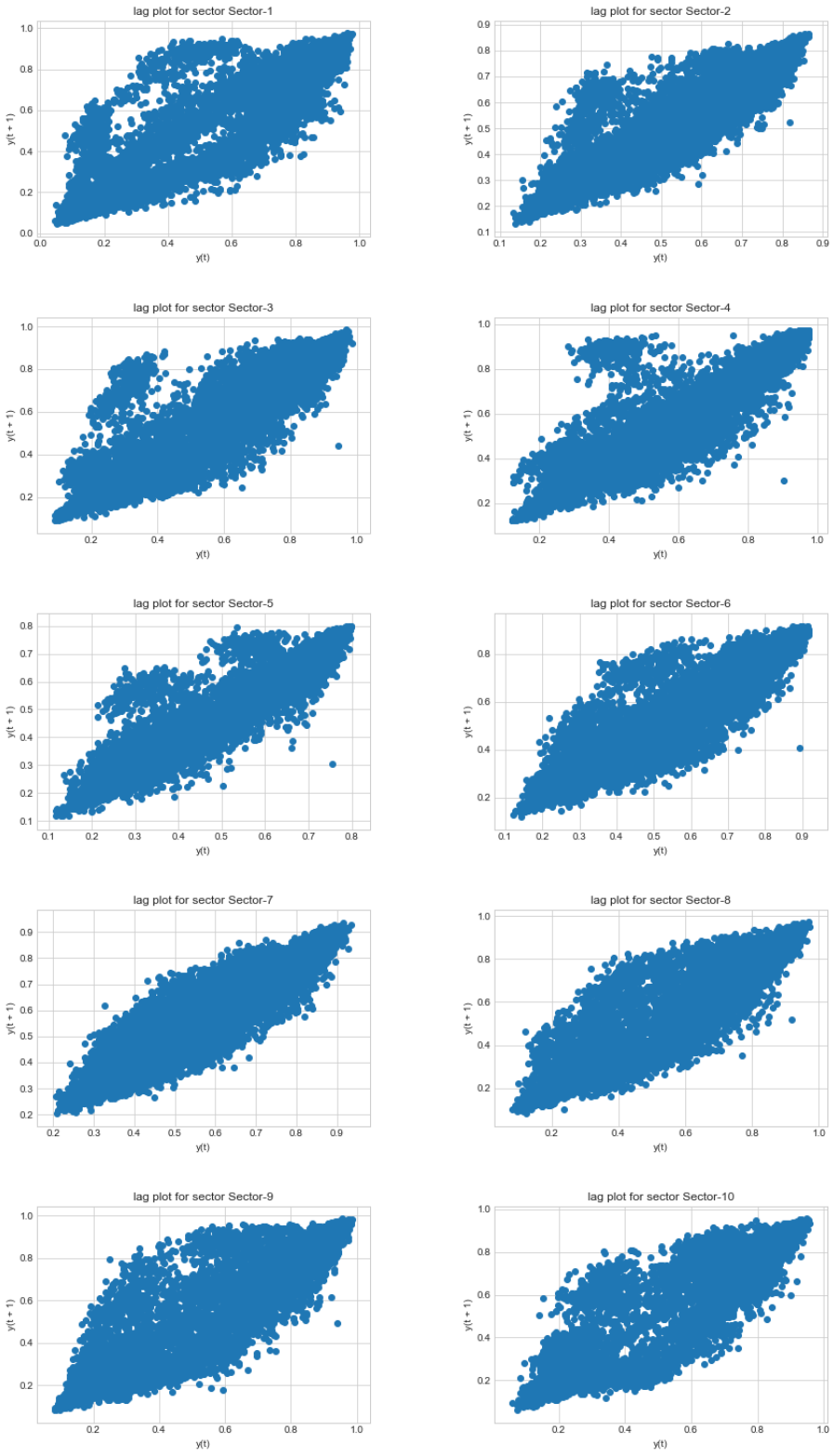


Figure 3.110: Lag plots for all Riyadh sectors. Top left to bottom right: sectors 1 to 10.

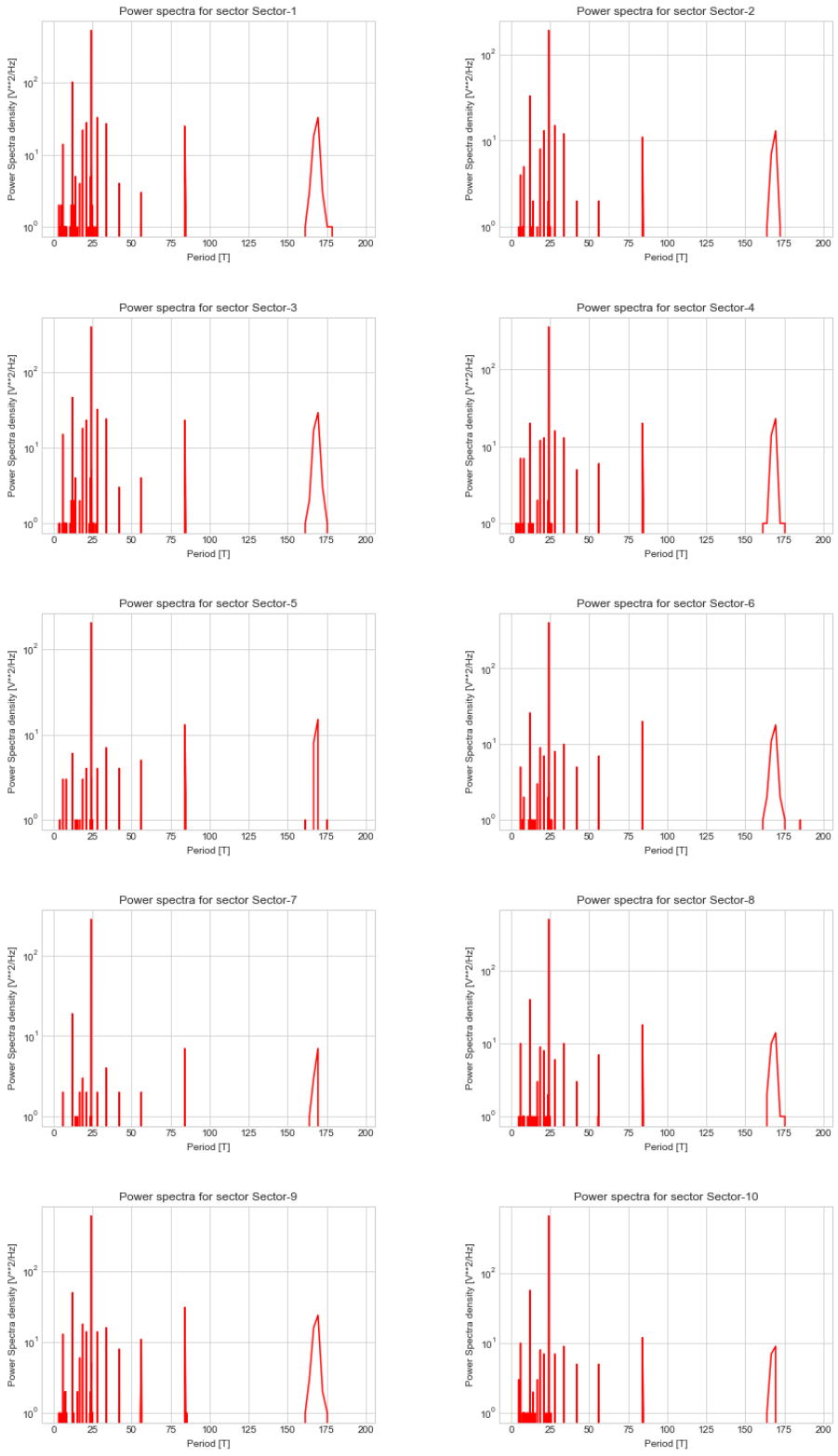


Figure 3.111: Riyadh occupancy/hour in the frequency domain. From top left to bottom right: sectors 1 to 10

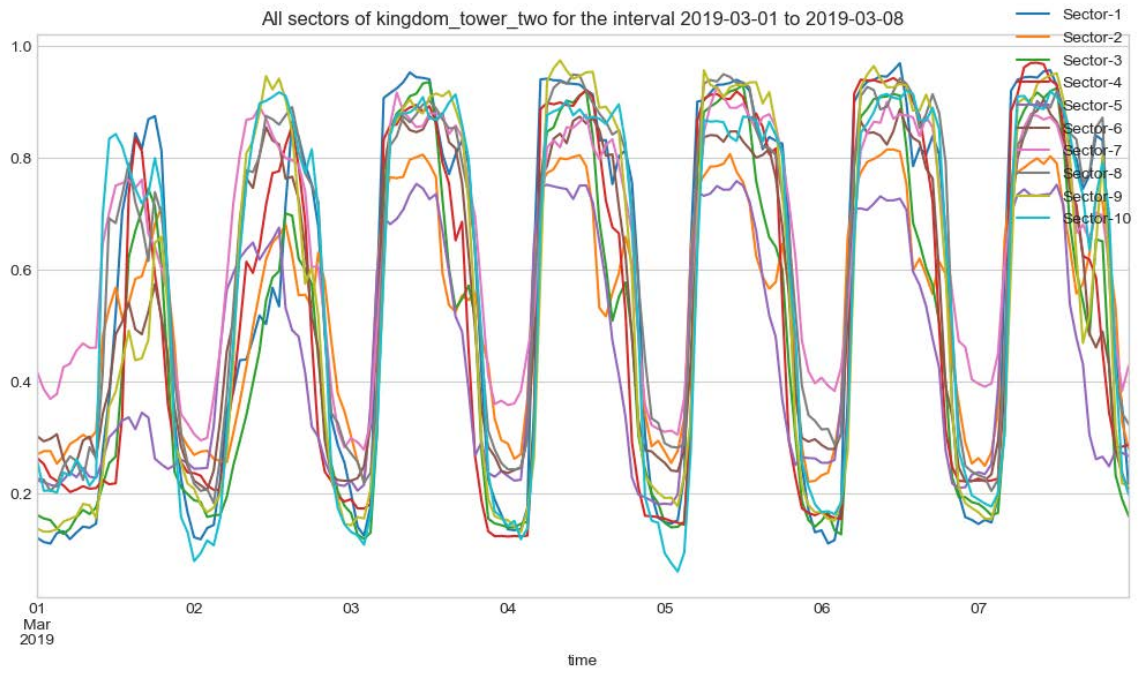


Figure 3.112: Riyadh sector realizations for the first week of March 2019, with the first day being Friday and the last Thursday.

Thus, it would be of great interest to develop models that can overcome these types of handicaps.

In closing, I provide here summaries of the important characteristics of each scenario.

3.6.1 Antwerp summary

The Pelikaanstraat sector has short stays because this sector is next to the train station and is used by taxis. Because at night the use of the sector changes, its distribution presents two differentiated peaks and large variance.

Zones 1 and 2 on Van Wesenbekestraat present correlations in their lags and similar behaviours. Zones 1 and 3 in the Van Schoonhovestraat sector also present correlations and similar patterns. Zone 2 exhibits chaotic behaviour.

3.6.2 Barcelona summary

The only sector that is in Barcelona scenario has short stays and high variance, but with a very defined pattern throughout the days, weekdays being different from weekends. Its distributions also present two different peaks.

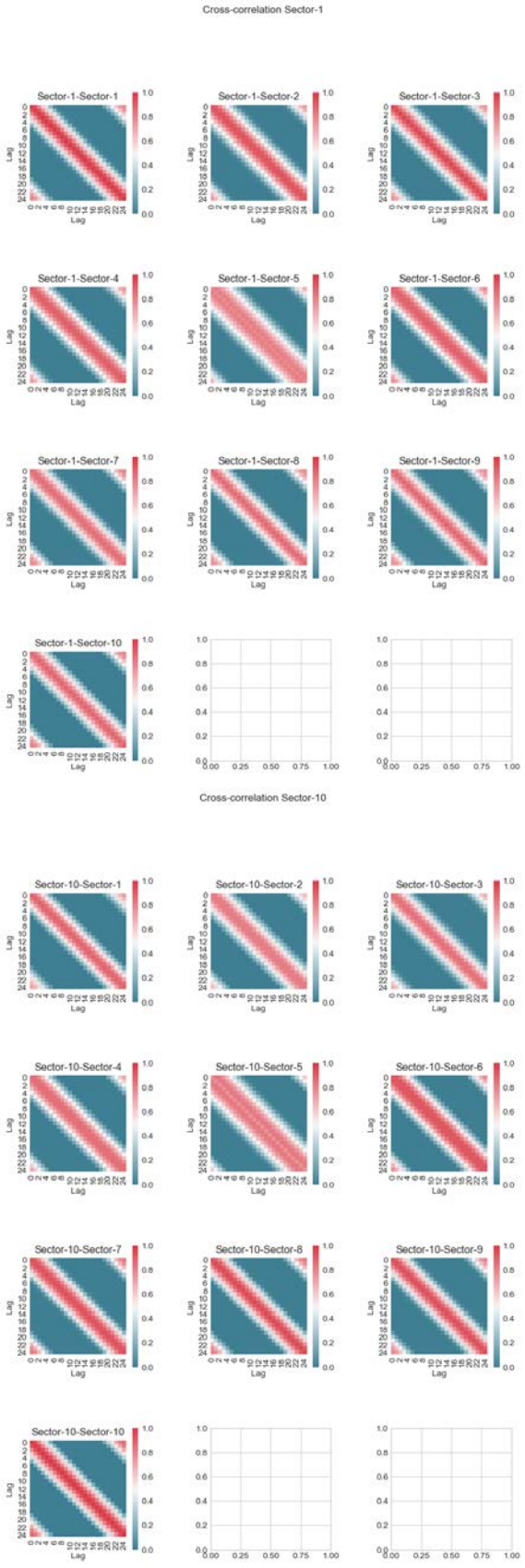


Figure 3.113: Cross-correlations for Riyadh sectors. Only sectors 1 and 10 are shown, as all of them are identical.

3.6.3 Wattens summary

The two sectors in Wattens (Inside and Outside) have similar patterns and behaviours, with two peaks in their distributions. Their behaviours are linked to their being a company's private parking facility and, thus, they follow patterns similar to office hours.

3.6.4 Los Angeles summary

The sectors in the Los Angeles scenario present two peaks in their distributions, as their behaviour in time is very different between weekdays and weekends. The Standard, Permit and Disabled sectors present correlations in time, but the Zipcar sector is chaotic in its behaviour.

The Los Angeles sectors present an anomaly that affects the occupancy levels during all of 2019, especially in the Permit sector.

3.6.5 Riyadh summary

The ten sectors in the Riyadh scenario have very similar patterns that fade as the distance between sectors increases. In addition, the sectors have two different behaviours: holidays vs. workdays.

Because the scenario covers a wide area, interesting patterns arise:

- Seasonal plots have detected behaviour changes for all sectors during May, indicating that people in the zone are working part-time that month and returning home earlier, while others are visiting the area at night for leisure. One possible explanation for this is that May coincides with the Ramadan holiday.
- An anomaly has been detected for some sectors in December 2019, for which no explanation has been found.

4 Related work

This chapter presents the research related to the main topics in this thesis and the state of the art during its development. Due to the lack of technology, the literature did not consider the modeling of parking behaviour until the emergence of Internet of Things (IoT) platforms. This technology made it possible to use approaches similar to those for traffic modeling, for which forecasting has been actively researched for years. This chapter first introduces the research into and evolution of Smart Parking, then presents the literature on time series forecasting models for traffic and parking from two different approaches: classical statistics and machine learning methods.

4.1 Smart parking

The area of Intelligent Transportation Systems (ITS) focuses on improving modes of transport and traffic management through innovation. In the case of parking, it provides systems like Parking Guidance Information (PGI), which delivers dynamic information about the state of parking zones and facilities to drivers using Virtual Message Signs (VMS) on the roads or via Internet (UK Government [2003]). Polak et al. [1990] demonstrated that PGI systems contribute to reducing overall traffic congestion by helping drivers find free parking spaces.

The classical methods for PGI consist of using four components: a parking monitoring mechanism, parking space information, telecommunications networks, and a control center. Figure 4.1 shows from left to right the relationship between the components: parking space information (usually VMS), the control center, and a parking state monitoring mechanism. All of them exchange information via a telecommunications network, usually Ethernet. The problem with these systems was that they could only be used for off-street parking systems; yet, even in this case, it was difficult to keep an exact count of the number of available parking spaces, as explained in the review by Kotb et al. [2017].

As a response to these problems, a new area called smart parking came into development, with a focus on the use of emerging technologies for providing services that improve management of off-street parking facilities and helping users find a parking spot.

The smart parking literature has been divided in regard to the nature of parking. For off-street parking, researchers concentrate on parking reservation and the efficient use of the parking facility infrastructures; while on-street parking studies focus on occupancy

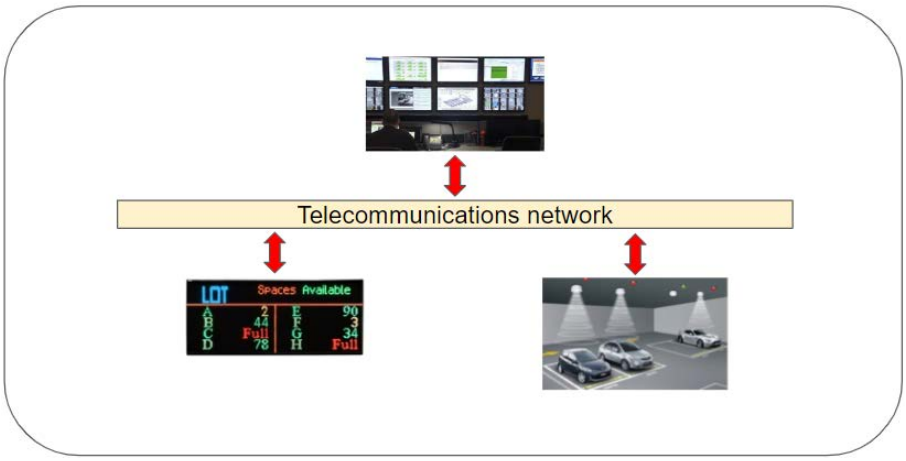


Figure 4.1: Parts that compose a PGI system.

prediction and estimating traffic caused by parking (Weinberger and Hampshire [2017]). A small part of the literature has explored the possibility of converting on-street parking in urban zones to fee-based parking, thus motivating the use of some off-street parking solutions for on-street parking, specifically monetary solutions. Different approaches have been explored for computing flexible prices for on-street parking slots, some of which are: taking into account the number of free spots available in a zone (Teodorovic and Lucic [2006]); the hour of the day (Shoup [2011]); the size of the vehicle (Nandekar et al. [2015]); and discounts based on the CO_2 emissions of the vehicle (Shoup [2016]).

Lin et al. [2017] provides an overview of the smart parking field from the year 2000 to 2016. First, it describes how the technology evolved from a need to capture and deliver to users two flows of information: traffic and parking (called "Information" in the mentioned study). Traffic information concerns pathways taken to find parking, parking availability, and riding to the desired slot. The parking information flow begins at the moment the parking slot status changes and the information is then delivered to the drivers. Bad management of these flows results in competition, which increases parking conflicts when demand for a spot is huge. Lin classifies the literature for smart parking under three themes:

1. The management and collection of the information. This includes information sensing, sensor connectivity, parking meters, crowdsensing and shared parking. In particular, it focuses on different existing sensing techniques to identify parking space status.
2. System deployment. This exploits and conducts a statistical analysis of the collected data. The system is composed of E-Parking (virtualization of the sensors and networks), reservation, guidance and monitoring of the information.
3. Service dissemination. This analyses the relationship between information and social features. It focuses on how to disseminate the information (where to publish and how) and how parking competition is caused by individual behavior (when

more than two drivers contend for the same parking spot).

The key pieces in current smart parking solutions are found in system deployment: software systems that collect, store and use the information; large-scale deployment (sensors and telecommunications at the city-scale); and prediction services.

Prediction services are related to characterizing system processes in order to obtain models. The aim of modelization is to recreate real world system processes as mathematical functions for creating simulation tools and predicting the outcomes of these systems in concrete situations. Modeling can be done with different approaches, one of them being the use of data that originates from the system under study. This approach is known in the literature as a data-driven approach, and its techniques are categorized into two groups: classical statistical methods and machine learning methods. Historically, modeling for traffic predictions appeared long before that of parking. Some of the earlier works are: Ahmed [1979] for free flow prediction; M. Hamed et al. [1995] for urban traffic; and Okutani and Stephanedes [1984] and Laetz [1990] for predictions in congested situations. The reason could be that traffic problems are more costly, related to safety and legal penalties (Kawashima et al. [1995]). In addition, technology plays an important role in obtaining measurements. In the words of Donald Shoup:

> *"Parking has been the most stagnant technology industry for years. Two main technologies can transform the parking paradigm: occupancy sensors and variable price parking meters."* (Shoup [2018]).

Prediction for parking analytics has always been a driving force behind predictive analytics for traffic, which has been an active area of research for many years. This in turn has motivated efforts to adapt promising traffic prediction methods to parking prediction.

4.2 Predictive modeling

Predictive modeling for parking and traffic became stronger with the emergence of IoT networks (Khanna and Anand [2016]), computational capabilities and data-driven applications (Royal Society [2017]). In the case of parking, the literature is recent but has learnt a lot from the traffic literature as can be observed in the evolution of the traffic research area: traffic and parking forecasting models are currently based on time series but some time ago prediction for traffic considered a broad domain of models. Both areas got an increasing number of publications thanks to technology advancements, for traffic began around the year 2000, and for parking around 2010. The massive deployment of parking sensor networks motivated the appearance of city-scale projects for parking (Atzori et al. [2010]), like SFPark in San Francisco (SFPark [2014]), LA Express park Los Angeles Times [2010], Melbourne and SmartSantander (Santander) (Lanza et al. [2016]). This led the way to time series forecasting modeling for parking.

4.2.1 Predictive modeling for traffic

The reviews of the short-term traffic prediction models found in Lint et al. [2007] and Vlahogianni et al. [2004] can be taken as the first efforts in the literature to classify the modeling approaches used for traffic prediction. These works compares the different approaches of the literature on the basis of: method; time horizon; scale (fixed location, route and network); environment (urban and freeway); and computational effort and accuracy (usually root mean square error (RMSE) and mean absolute percentage error (MAPE)). They also describes the most important variables predicted for a time interval: flow and intensity (vehicles/hour); density (vehicles/km); occupancy (percent of time a detector is occupied by vehicles); travel time (minutes); and mean speed (km/h). The proposed taxonomy, shown in Figure 4.2, classifies the prediction methods into:

- *Naïve methods.* These are models without any model assumptions. The prediction forecast could be the value of the instant measurement, an average of past values, or the result from clustering similar traffic patterns and averaging the most similar cluster and applying it to the current value.
- *Parametric methods.* These models find the parameters using data, and their structure is predetermined. They are traffic simulation models (macroscopic, microscopic, mesoscopic and three-phase traffic theory) and time series (linear regression, ARIMA family, Kalman filtering, ATHENA, SETAR and Gaussian Maximum Likelihood).
- *Non-parametric methods.* These are models whose number of parameters are not fixed in advance and neither is their structure. Instead, both are learned from data. They are k-nearest neighbor, local weighted regression, fuzzy logic, Bayesian networks, neural networks (NN), and rarely used methods like decision trees and support vector regression.

Vlahogianni and Karlaftis [2013] conducts a concise study and comparison between ARIMA methods and the NN family of models, demonstrating that in most cases they provide equivalent results. In addition, it presents statistical methods for NN that provide unbiased predictions, reporting that NN approaches in the literature focus only on accuracy. The study by Vlahogianni et al. [2014] is also interesting, as it gathers the previous literature on traffic predictions and presents key issues for future research, like the development of statistics for better machine learning models (detecting bias and comparing different models). It also recommends the use of emerging technologies like cloud computing, vehicle-to-vehicle communication and exogenous data like social network information.

This work treats the traffic and parking literature on data-driven models for prediction using the classification presented in Figure 4.3.

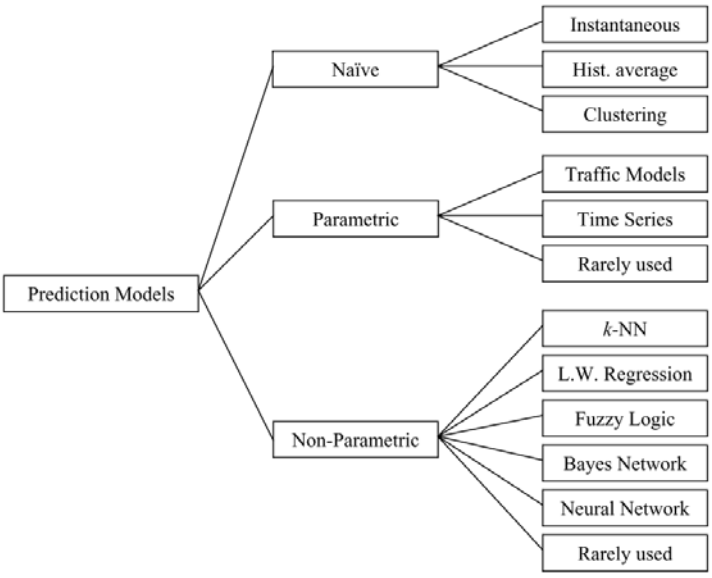


Figure 4.2: Taxonomy of prediction models. *Source: Lint et al. [2007]*

Classical statistical methods

The Box-Jenkins methodology (Box et al. [1976]) for the analysis and forecasting of time series is one of the first approaches to traffic forecasting. The resulting models, called ARIMA, assume that the underlying data process is linear and stationary. In Ahmed [1979], the Box-Jenkins methodology is used for predicting freeway traffic volume and occupancy. The result is that the best ARIMA model is a moving average of order 3, but it is not capable of dealing with changes in the variability of the traffic process, which is necessary for updating the model parameters online. In Nicholson and Swann [1974], spectral analysis is used to predict traffic flow volume with decent accuracy. The main disadvantage of the method is its inability to rapidly account for unforeseen changes that are not reflected in the previous data. A contemporaneous study used a Kalman filter to estimate traffic variables like number of vehicles and velocity for a single link using traffic flow Ghosh and Knapp [1978]. The data used in those studies was always from a single link (usually a tunnel) and an intersection, with a prediction time interval of 5 minutes.

Okutani and Stephanedes [1984] resumes the work by Ahmed [1979], without any solution that is more accurate than a simple moving average, and by Nicholson and Swann [1974], obtaining decent results but with problems in the model's adaptability to unseen changes. Their work presents an alternative method for predicting 15 minutes of traffic volume using more than one link with Kalman filters. This is the first work that uses seasonal data as input for the model: the data from the day before and from the same day of the week before. They achieved good accuracy but had some computational issues. This work demonstrated that traffic is a non-linear problem that cannot be re-

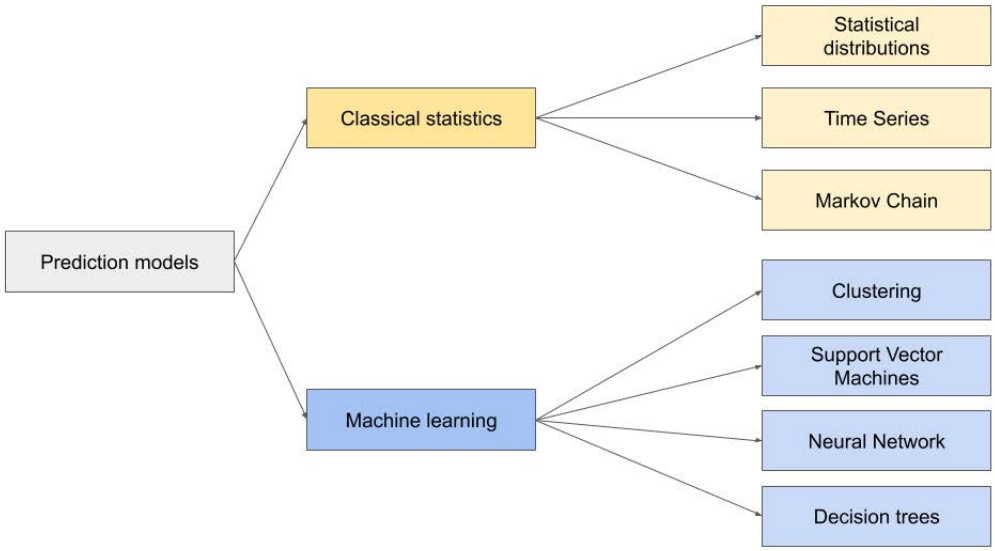


Figure 4.3: Prediction models classification.

solved with classical ARIMA methods. After this work, the Kalman filter was widely used for the estimation of traffic variables. One example is the estimation of traffic flow using origin-destination (OD) matrices, as shown in Antoniou et al. [2010] and Antoniou et al. [2016]. New solutions were provided using derivations and combinations of existing methods from the original ARIMA, like SARIMA used in Williams and Hoel [2003] and the combination of Kohonen self-organizing maps and ARIMA used in Van Der Voort et al. [1996]. Modern solutions use GPS data and STARMA models that exploit correlations between space and time (Safikhani et al. [2017]). A similar development occurred with the Kalman filter in Stathopoulos and Karlaftis [2003], where a multivariate formulation provided a solution for estimating traffic volume, travel speed and traffic density, thanks to data coming from loop detectors on the streets. Other solutions based on an unscented Kalman filter are used for freeway traffic estimation in Mihaylova et al. [2006], and a particle filter for travel time predictions in real-time is proposed in Chen and Rakha [2014].

The work by Vlahogianni et al. [2006] provides tools for detecting non-linearity and non-stationarity in univariate short-term time series while also characterizing the traffic data. In Lan and Kuo [2005], traffic time series are identified as chaotic (time series that could be interpreted as stochastic due to their behavior but that generated by a deterministic system), and tools are provided for identifying these types of time series.

The partial least squares method is suggested in Dauwels et al. [2014] for multiple forecasts of traffic variables, multiple links and multiple time horizons. Instead of using predictions for each link, they use a model for the whole network and also provide different forecasting horizons. They use a novel form of organizing the data as tensors instead of the usual matrix from cross-tabular data. This allows parallel computing to estimate the model, although they report scalability problems and that the model

cannot be computed for large networks.

Other solutions are based on Bayesian networks for computing the traffic flow of a link based on parent nodes, as seen in Sun et al. [2006], Osvaldo Anacleto and Albers [2013] and Castillo et al. [2008], where all proposed methods are based on OD matrix estimation. In 2017, Worldsensing adopted the solution provided by Scalabrin et al. [2017] for traffic speed prediction, while also providing anomaly detection using a Gaussian Mixture Model (GMM) for clustering the different link behaviors.

Machine learning methods

The first work that gave relevant results using NN for traffic problems can be found in Yu and Chen [1993]. They proposed a multi-layer perceptron (MLP) as a new approach to time series traffic data, because ARIMA only works under the assumption that the modeled process is stationary and linear, a requirement that is not satisfied by the traffic process. In fact, they proposed using a chaotic function to simulate the process that generates traffic patterns. They showed NN's capability of approximating complex continuous functions, as demonstrated in Hornik et al. [1989], but emphasized that not just any methodology can be followed for obtaining the best NN model architecture. Similarly, in Zhang et al. [1997], an NN was presented for traffic flow predictions. The data was generated from a macroscopic traffic model (higher-order continuum traffic model), and the results showed that the NN is capable of learning the generation process.

Baher Abdulhai et al. [1998] presented a new type of NN called time delay neural network (TDNN), the structure of which is optimized using a genetic algorithm (GA). It is capable of predicting the flow and occupancy of freeways with simulated and real data. The traffic flow data comes from target links upstream and downstream, which ensures the possibility of learning patterns affected by flow changes and shockwaves. TDNN architecture has a memory in each neuron that permits remembering previous layer outputs for N periods of time. Also, an in-depth study was conducted on how to treat traffic variables by examining the look-back interval, the prediction horizon, the spatial contribution, the input data resolution, and how these all affect prediction accuracy. The results showed that the prediction errors varied inversely with the extent of the spatial contribution. Longer prediction horizons had a mean regression effect, and they demonstrated that the aggregation level of the data has a direct effect on prediction horizon accuracy. Finally, they obtained acceptable results, which were compared with Zhang et al. [1997].

Different types of recurrent NN (Elman [1990]) are used in Ishak et al. [2003] for traffic flow prediction using data coming from target links upstream and downstream. These types of NN adjust their parameters by learning from data and from time sequences, but they are computationally costly to train while their structure is also difficult to optimize. The results were compared with non-linear regressions and showed that these types of NN achieved better performance.

The works by Baher Abdulhai et al. [1998], Ishak et al. [2003] and van Lint et al. [2005] focus on using recurrent NN for data with missing values caused by sensor failures. The recurrent NN used is called a state-space neural network (SSNN) (Zamarreño and Vega [1998]), and the data is imputed using exponential forecasts and spatial interpolation. They also use Bayesian regularization MacKay [1992] to train the NN.

In the work by Vlahogianni et al. [2005], an MLP approach is presented using a GA for meta-optimization (or hyper-parameter tuning) with the objective of providing short-term forecasting of traffic variables such as flow and occupancy. This solution is based on previous literature finding NN to be one of the best alternatives for modeling and predicting traffic parameters with time series. It also provides a methodology for optimizing the NN architecture. At the same time, Medeiros et al. [2006] present a work that they call AR-NN, where an NN is modeled using a statistical approach similar to those used in time series.

In order to improve traffic predictions, a methodology for identifying traffic patterns is developed in Vlahogianni et al. [2008]. It allows identifying traffic patterns in a network, the conditions by which these patterns occur, and their temporal evolution. The method uses Kohonen self-organizing maps (KSOM) to identify the clusters formed by these patterns.

Jin [2007] presents what is called a neuro-fuzzy decision tree (Neuro-FDT) that converts information from traffic into human-readable decisions using a fuzzy logic methodology.

A support-vector machine (SVM) is used in AiLing et al. [2002] to treat traffic flow prediction based on time series data for problems where traditional statistical methods like ARIMA fail. They emphasize that statistical learning theory (Vapnik [1999]) should be the way forward when traditional methods fail. In Wang and Shi [2013], SVM is used with a wavelet kernel to treat the traffic data with chaos time series theory.

The most recent literature focuses on using deep NN models (MLP, convolutional and recurrent networks with large numbers of hidden layers), which gained popularity due to the emergence of libraries that permit using GPUs for the training process. These are very well suited to the Big Data framework. “Deep” means that these models stack more than one hidden layer and thus enhance the NN’s pattern recognition capabilities, but the NNs also increase the complexity of these models, making them more difficult to understand. Some related works are: Yanjie Duan et al. [2016], where long short-term memory NN (recurrent NN (Hochreiter and Urgen Schmidhuber [1997])) is used to train 66 models (1 per link) for travel flow prediction; Fu et al. [2017], where ARIMA, LSTM and GRU (gated recurrent unit NN (Cho et al. [2014]), an evolution of LSTM) are compared for traffic flow forecasts; and Polson and Sokolov [2017], where a deep NN is used for traffic flow prediction. Some authors believe that data-driven traffic modeling needs to use spatial and temporal information together. From this point of view, the following works in the literature focus on capturing spatial-temporal relationships using a combination of convolutional NN and recurrent NN Yu et al. [2017], Cui et al. [2018], Cheng et al. [2018]. More recently, traffic forecasting has focused on graph convolution NN (GCNN) models (Bruna et al. [2013], Henaff et al. [2015]) and proposing various

improvements to this method (Defferrard et al. [2016], Kipf and Welling [2016]). The general idea of GCNN for traffic consists of capturing spatial-temporal patterns using a graph G= (V, E) based on the road network, where the vertices V are usually the nodes containing the traffic information (feature matrix X_t that evolves over time) and E is a set of edges that connects the nodes. The adjacency matrix A is used to represent the connection between roads, where a 0 value represents no link between roads and a 1 represents the existence of a link between the roads. The A matrix is used to obtain the Laplacian matrix L, where the eigenvalues and eigenvectors are extracted. At this point, it is possible to use convolutions in order to learn spatial patterns from the node information, and recurrent layers to learn temporal dependencies. This approach is used in Li et al. [2017], Yu et al. [2019a,b], Wang et al. [2018].

4.2.2 Time series forecast for parking

As mentioned before, the area of time series forecasting models for parking analytics has always been a driving force behind the same area related to traffic, which has been an active area of research for many years.

The usual methods for obtaining parking data are: sensors, VANET (vehicular ad-hoc network) and GPS for detecting the number of vehicles generating cruising traffic in a zone. The variables of interest are availability, reservation (Caicedo et al. [2012]) and occupancy (usually in the form of parking occupancy rate (OCCR)) (Vlahogianni et al. [2016]).

Classical statistical methods

The first works in this area focused on off-street parking (usually lots in malls and airports) to provide strategies for parking reservation and generate revenue from parking spots. In Teodorovic and Lucic [2006], a reservation system is optimized using fuzzy logic to take into account available space and provide different costs for the parking spaces. Another approach exploits the use of VANET by obtaining data from the communications between vehicles and sensors in lots. This is similar to Caliskan et al. [2007] and Klappenecker et al. [2010], who propose a Markov chain model that tells users the probability of an available parking space upon arrival.

An intelligent parking assistant is presented in Rajabioun et al. [2013]. This system finds the most suitable parking space based on user preferences and learned behavior, taking into account parameters like parking duration, arrival time, cost preference, driving time, lot parking rules and prices. They use data from on-street parking meters, off-street parking garages and free parking spaces. The needed infrastructures are: a device capable of connecting to the Internet with access to the vehicle's GPS; and a centralized server that orchestrates the information provided to users. They propose a prediction algorithm that fits the available historical parking data to a chi-square distribution, then delivers the prediction based on the value of real-time data. Another parking

recommendation system is presented in Wu et al. [2014], which uses VANET and a statistical distribution to model parking availability in different parking lots The system offers a guidance system for finding lots with available space.

In the work by Caicedo et al. [2012], a methodology for predicting real-time parking space availability is developed by simulating parking requests, estimating departures and predicting parking availability. The proposed algorithm uses historical data to characterize the parking requests (Poisson distribution), the parking duration time (gamma distribution), and the driver's decision on which lot to go to, then provides a parking availability prediction for the different lots in the area.

In Hampshire et al. [2011], an ordinary partial least squares model is presented for 10-minute forecasts of parking availability using sensor data.

A vector autoregressive model (multivariate ARIMA) is used in Rajabioun and Ioannou [2015] to predict the availability of parking at arrival time, taking into account temporal and spatial correlations. This is one of the first works to use large-scale sensor network data and it also provides an online update of the model parameters based on a recurrent algorithm using maximum likelihood estimation.

Machine learning methods

A fuzzy logic model is proposed in Chen et al. [2013] to handle the uncertainty of finding available parking in a desired zone at a desired time while taking into account departure time and possible uses of park and ride. It also computes the shortest travel path.

Smartphone data combined with an app is used in Koster et al. [2014], which employs parking maneuver recognition by means of the smartphone sensors (detecting when the car is leaving or entering the parking spot). With this information, plus the historical parking data contextualized according to geographic region, calendar effects, weather, real-time traffic information and special events, the authors propose building a model that can predict free parking spaces and consumption rates. The solution has two parts: recognition of parking maneuvers and calculation of parking availability. In the first part, the data from the smartphone is used to create a model based on k-medoids clustering (because finding the centroid in a time series is a difficult problem), with a dynamic time warping (DTW) metric that permits recognition of the car maneuvers (label data). The labeled data is then used to create a conditional random field (CRF) model to classify new series of sensor readings. In the second part, when the app requests information from the server about the availability of parking in a zone, the server performs two computations: it uses historical data and a Bayesian approach to compute the distance around a point for which the probability of finding a parking spot at the given time is sufficiently high. Then, it uses a hidden Markov model based on last reported status to calculate the probability of the currently known free spaces (within the calculated distance) being free at arrival time.

Zheng et al. [2014] is one of the first proposals that conducted a data analysis for a large-scale sensor network: the SFPark platform. They analyzed the data using clustering

based on the following algorithms: farthest first (FF), expectation maximization (EM) and support vector data description (SVDD). This allows finding interesting behaviors and detecting anomalies. Their results identify parking places that are overused or underused and sensors reporting apparently faulty data. Another work that utilized SFPark data is Richter et al. [2014], who had the objective of handling the coarse granularity of data with minimal effect on the accuracy of parking availability predictions while using the minimum number of models. They used a hierarchical clustering method with DTW (because data is time series) as the distance metric, and propose studying the accuracy in the following scenarios: one model per road segment (a road segment consists of more than one on-street parking space); one model per link for each day of the week; one model for weekdays; Saturday and Sunday; and, lastly, predictions using the results of spatial clustering and of temporal clustering. The results find that one model per day and link gives the best accuracy (80%), while the clustering scenarios give the worst (approximately 70% in both cases). However, the storage used for the models in the clustering scenario is 99% less than that of the one model per day and link scenario. The prediction is taken as the centroid of the cluster where the road segment is assigned.

In Zheng et al. [2015], data from SFPark and Melbourne's large-scale parking sensor networks are used to provide models for predicting parking OCCR ($\frac{number\ of\ slot\ currenly\ occupied}{number\ slots\ currently\ operational}$) by means of three different feature sets. The algorithms used are regression tree (RT), NN and SVR. The feature sets are: occupancy (0 or 1) based on day of the week and time of day as a float; occupancy based on past occupancies; and, lastly, a combination of the two feature sets. The results indicate that the RT achieved higher accuracy with the combined set of features.

One of the most cited studies in the recent literature on parking prediction is Vlahogianni et al. [2016], which used data from the SmartSantander project. They divided the sensor network into 4 different regions and proposed, for each region, a hazard-based model for predicting the occupancy time of a parking slot and an NN model (MLP) for predicting parking availability. The data is organized as 1-minute data and can provide up to 30 minutes of forecasts with different time horizons (1/5/15/30 minutes). The work focuses on the following research: parking prediction based on data from an IoT network sensor; the prediction accuracy of such models in terms of the forecasting horizon; the statistical properties of parking space duration; and the probability of having free space in a certain area. They also studied the average duration of a parking space being available for a given time period and occupancy as a percentage of parking slots occupied during a time period. The model inputs are the past values of the studied features (so the NN can predict occupancy) and exogenous variables like the day of the week. The methodology first uses the NN to predict parking occupancy, and then uses the output as an input to the hazard-based model (in combination with the time of day, time period (peak or off-peak) and weekday). In this way, it predicts the probability and number of spaces available in the desired zone. For the hazard-based model, a Weibull survival function is used, because the historical data indicates that the duration of a free parking space follows a Weibull distribution. The conclusions are that better NN structures

must consider more seasonality and take into account the traffic variables obtained from parking zones.

Blythe et al. [2015] proposes transforming the time series of parking space availability in order to reconstruct them in phase space, and then use the transformed data to train a wavelet NN (WNN) model. The continuous wavelet function of WNN serves as an activation function in the nodes of the hidden layer. The study reports good accuracy using survey data.

The work by Piovesan et al. [2016] provides analytics for parking data using an automated classification algorithm. This algorithm delivers information on the anomaly behavior of parking sensors and provides a clustering method using spatio-temporal patterns. As a result, it indicates that parking patterns can be detected by jointly using multiple metrics like parking event state duration (free or occupied) and the frequency of parking events. It also differentiates day of the week and hour of the day. They use different statistics and machine learning algorithms like expectation maximization (EM), Kohonen self-organizing maps (KSOM), k-means and density-based spatial clustering of applications with noise (DBSCAN).

Naïve Bayes and a decision tree are used in Hsueh et al. [2018] for predicting parking occupancy as a categorical variable (level: low, mid, high) at arrival time. The study is relevant because it uses three sets of different types of features: temporal (day of the week, time of the day (Tod) and occupancy rate); spatial (nearest parking streets zones, similar parking streets zones based on RMSE, similar parking streets based on RMSE with different ToD); and others (block id and hourly precipitation).

Another approach to the parking prediction problem is based on detecting traffic that results from cruising. Jones et al. [2018] uses data from smartphones (GPS, accelerometer, magnetometer, and gyroscope) in order to detect vehicles cruising for parking. The detection of cruising can be useful for updating probabilistic parking availability in the desired zone. Using GPS data, map matching is carried out via a hidden Markov model to localize the zones where the vehicle is located. Cruising is detected as a significant local minimum in the GPS trace relative to the distance to the destination. Data from the other sensors are used for training ML algorithms. The ML algorithms that ascertain whether or not the user is cruising are decision tree, K-NN, and SVM. One of the problems with this proposal is that accuracy depends on the number of users in the zone.

Ionita et al. [2018] takes a different approach as a parking recommendation system. Their objective is to model the whole city (San Francisco) by creating parking demand profiles. The profiles are based on parking occupancy, traffic, weather, events, parking revenue, fuel prices, distance to points of interest (POI) and amenity time in these places. For those areas without data, they create demand profiles using similarity functions for areas where data is available, then apply a clustering algorithm (K-means). Different machine learning models are tested on the clustered scenario and compared to the whole city scenario. Those models are decision trees, support vector machines, random forest, multilayer perceptron and gradient boosted trees. The one with the lowest RMSE in

the clustered scenario is gradient boosted trees.

Recent works have begun to use and compare deep learning approaches. Fedchenkov et al. [2018] uses data from sensors in private parking lots with a time-window interval of 10 minutes to create an LSTM model that will be used in vehicle navigation systems, specifically to recommend where to park based on the distance to POIs. Another interesting work is Badii et al. [2018], where different machine learning models are used to model the data from off-street parking, which is updated every 15 minutes (thus, they use this as a time step). They offer predictions of between 30 and 60 minutes. The exogenous data that they consider is day, month, day of the week, weekend, previous observation difference, observation difference of the week before, temperature, humidity, rainfall, hourly mean speed in nearby zones, vehicle flow in nearby zones, vehicle travel time and traffic density in nearby zones. The models used are support vector machines, ARIMA, NN with Bayesian regularization and RNN. Based on error metrics (MAPE, MSE, MASE), the model with the lowest error is the NN with Bayesian regularization. In Shao et al. [2018], a similar approach to Vlahogianni et al. [2016] is used to forecast parking OCCR. Historical data from 3,000 on-street sensors in the city of Melbourne are used to forecast parking availability and parking duration. A clustering approach is used to group the parking spot behavior by means of a K-means algorithm, with K=30. Then, for each cluster, an LSTM model is created. For the parking time duration, a nonlinear regression model is used. The exogenous variables are day of the week, time, occupancy rate, arrival time, departure time, duration time, longitude and latitude of the parking spot. A 10-minute time window is used for the inputs, and the forecasts cover a time interval of between 1 to 30 minutes. They perform two experiments: the first contrasts the cluster vs. non-cluster approach; the second compares LSTM with MLP. Based on error metrics (MAE, RMSE, MAPE, RRSE), the LSTM clustering approach has the least error.

Yang et al. [2019] is the first work for parking prediction that takes the graph approach in a similar way that recent works in traffic literature do. It combines graph CNN and LSTM in order to forecast parking OCCR in the city of Pittsburgh. The strategy is based on exploiting spatial and temporal relationships at the parking zone-level. The roadway network topology is extracted from the Google MAP API, and the proposed method is compared with LSTM, LASSO and historical mean. In order to accomplish the objective, it uses different sources of data: road network topology, traffic speed in parking zones blocks, parking occupancy computed from parking meter transactions and, finally, weather data. Time series are interpolated for a temporal resolution of 10 minutes. After comparing the proposed method with other models like LSTM, LASSO, historical average and latest observation, it outperforms them all in forecasts of 30 minutes into the future.

5 Theoretical aspects of time series algorithms

Given the problem of predicting parking occupancy presented in 1, the data analysis in 3, and the literature review presented in 4, we decided to use time series algorithms to provide forecasting values for parking occupancy.

The algorithms used (marked in red in Figure 5.1) are those based on ARIMA, as it is a popular algorithm in the literature with strong theoretical foundations. The weaknesses of this method are well known and, for that reason, this work also proposes using artificial neural network methods, which the traffic literature has begun using but have not yet been applied in parking prediction. The chosen methods belong to two different cultures of the data modeling research area as stated in Breiman [2001] and this will permit to explore a comparisson between the two approaches. With this in mind, this chapter aims to present the theoretical issues of both methods.

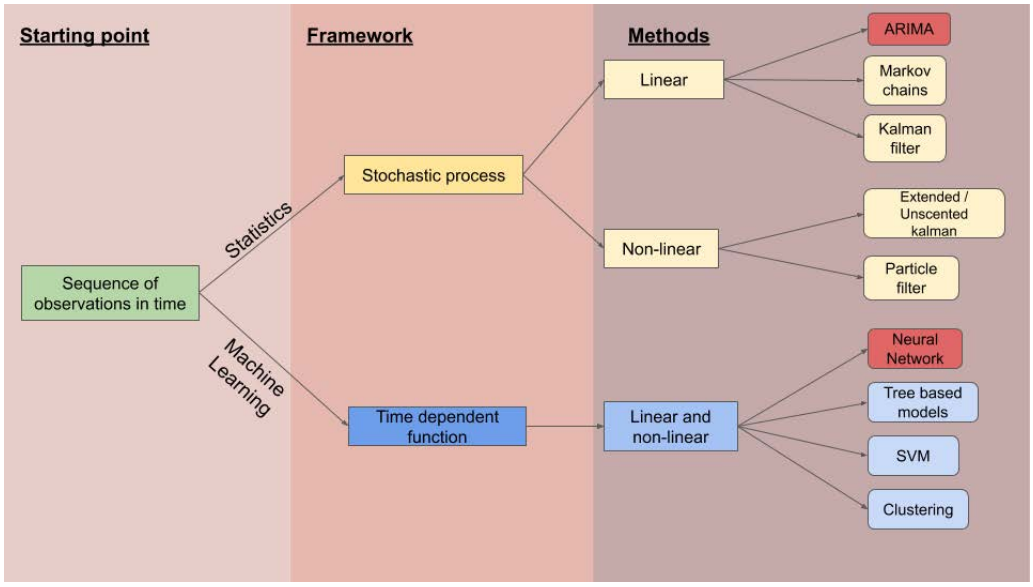


Figure 5.1: Time series modeling path and a subset of methods.

5.1 ARIMA

The autoregressive integrated moving average (ARIMA) model was popularized by Box and Jenkins Box et al. [1976]. It is a statistical tool for creating time series models that facilitate better understanding and future values of the data. ARIMA methods are comprised of three parts:

- Autoregression (AR). This defines the current time series value x_t as a function of its own past realizations up to time p, $x_{t-1}, x_{t-2}, ..., x_{t-p}$. It is assumed that this function is a linear function of the past values, an example of which can be observed in the lagged plots from Chapter 3. AR(p) defines an autoregressive model of order p, where p is the number of significant lags considered in the model (Equation 5.1). One of the requirements of AR(p) models is that the time series x_t must be stationary, a property explained in 3.2. It is assumed that the noise component is $w_t \sim wn(0, \sigma_w^2)$, that is, white noise ($wn$) with zero mean and standard deviation σ_w^2. For simplicity, theoretical AR models are assumed to have a mean value of 0.

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + ... + \phi_p x_{t-p} + w_t \tag{5.1}$$

  where x are the observations, w_t denotes white noise and the ϕ terms are the AR coefficients.

- Moving average (MA). The moving average part of the model assumes that the observed data at time t is a linear combination of the current and past white noise w_t. In other words, an MA model considers w_t to be the result of a linear combination of past regression errors. It is not necessary that this part of the model be stationary, due to the properties of w_t ($w_t \sim wn(0, \sigma_w^2)$). In order to decide on the number of significant q elements of an MA(q) model (Equation 5.2), one could extract this information from the ACF plot (examples of these visualizations can be found in Chapter 3, where the concept was introduced).

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + ... + \theta_q w_{t-q} \tag{5.2}$$

  where x_t denotes the value at instant t, w_t denotes white noise terms and θ_t are the MA coefficients.

- Integration (I). The purpose of this part of the ARIMA model is to achieve stationarity in a time series by differencing the series. Through differentiation, it is possible to eliminate trends (de-trending) and seasonality in the series. The operation consists of differencing each realization of the series from its last past observation, thus converting them into a series of differentiated values. For seasonality differencing, the operation consists of computing the difference between the values in a season and the values in the previous season. For example, when using seasonal difference in a monthly time series of yearly seasonality, then the first value of the series consists of $s_t = x_t - x_{t-12}$.

The differencing operation is done d times (with d being a parameter of the model) until the levels of μ become constant and independent of the time at which the data is observed. Notice that, in each differencing operation, the last time lag of the series is lost because of the operation. Another way to know if the series is stationary and needs to be differentiated is through the ACF plot. An ACF plot that decays slowly is an indication of non-stationarity in the series. Statistical tests exist for detecting whether a time series has stationarity or not, such as the augmented Dickey-Fuller test for unit roots and the Kwiatkowski-Phillips-Schmidt-Shin test for stationarity.

In order to know the value of p and q in ARMA(p,q) models, one method to accomplish this is through ACF and PACF plots. When an ACF visualization presents a sudden cut-off in the number of significant lags, this is an identity signature of MA(q) models. Moreover, the last significant lag is considered to be the value of the parameter q. Something similar happens with the PACF plot, where a cut-off in the visualization indicates the value of the parameter p of an AR(p) model.

Above, it is explained how to accomplish a constant mean. However, in order to achieve stationarity, it is also necessary to ensure that the variance is not dependent on time. One way to satisfy this requirement is by using logarithms. The benefits of using logarithms in a time series are twofold: they convert the exponential growth of the series into linear growth; and they transform the heteroscedasticity of the data into a constant. The former is due to the product rule of the logarithms; and the latter is because logarithms make larger values small while still maintaining the small ones, thus making the variance constant through time. A more general rule for data transformations for modeling can be found in Box and Cox [1982]

Modeling with the Box and Jenkins method consists of:

1. Making the mean and variance independent of t through transformations of the series, such as by differencing (order of I) and/or using the log transformation to stabilize the variance.
2. Visualizing ACF and PACF plots to determine the number for p and/or q.
3. Fitting the model and validating it by analyzing the residuals.

Estimating the parameters of an ARIMA model is usually done through the maximum likelihood function, but it can also be done by means of the least squares function and the method of moments. Detailed explanations about these methods can be found in Shumway and Stoffer [2017].

ARIMA can be extended for taking into account seasonality. In that case, extra parameters for seasonality are considered in the model, namely P, D and Q (in order to differentiate from the p, d and q parameters of ARIMA). Those are the terms for the seasonal AR, I and MA parts of the model, and they are called SARIMA, with the parameters (p,d,q) and (P,D,Q) (SARIMA(p,d,q)(P,D,Q)).

The limitations of ARIMA models come mainly from their assumptions and requirements. First of all, we have the assumption that the observations are a linear combina-

tion of past realizations, and their residuals do not always hold; thus, the model does not detect nonlinear patterns. Another weakness is the need for stationarity in the data. If the data presents trends or heteroscedasticity, it is necessary to transform it in order to remove those components; but this does not always achieve stationarity. Another weak point of ARIMA models is the difficulty in using exogenous variables. Two extensions exist, namely the autoregressive integrated moving average with exogenous (ARIMAX) variables model and the vector autoregressive moving average (VARMA) model, both of which try to manage this. While the former consists of adding the exogenous variables as new terms to the regression equation of an ARIMA model, the latter is applied when a set of time series are used as regressors towards a dependent variable and, thus, it extends ARIMA models to multivariate regression. Another important point is that is difficult to work with multiple seasons in ARIMA models, so one way to accomplish this is by means of a SARIMAX model using Fourier terms as exogenous variables, as in the work by Ludlow and Enders [2000]. Another way is to use an algorithm called TBATS, whose objective is to model data with complex seasonalities (multiple periodicities, not integer periods and/or long periods). Detailed information on this approach can be found in de Livera et al. [2011]. In 3, the data analysis revealed multiple seasonalities: daily and weekly. Thus, ARIMA methods may not be sufficient at providing good models for parking.

5.2 Artificial neural network

Artificial neural network (ANN), multi-layer perceptron (MLP) and shallow neural network are the names used in the literature to identify the same class of supervised models based on neural networks (NN). The standard form of these models follows a directed acyclic graph structure that emulates the biological connection between brain neurons through axons. The basic unit of the NN model is the neuron, also known as a perceptron. It appeared for the first time in Rosenblatt [1957] and has been evolving since then into networks of perceptrons.

In this work, the term ANN will be used when mentioning any form of algorithm based on a neural network. Shallow NN will be used when talking about the now classical neural network model that consists of one input layer, one hidden layer and an output layer. These were the starting blocks of modern neural network models when they consisted only of the feed-forward past computations (the name feed-forward comes from the fact that each layer's outputs are used as inputs for the next layer, and never the other way around). The parameters are tuned manually. Lastly, this work uses the term MLP when talking about shallow NN with one or more hidden layers. This enhances the model's ability to learn from complex feature spaces, as each layer's inputs are based on the previous layer's outputs. The key element that enabled the possibility of having stacks of hidden layers is an optimization algorithm that computes the model parameters. The optimization model is usually nonlinear, as MLP models achieve nonlinearity by using nonlinear functions that are applied to the computation of each of the neurons. Because

of its simplicity and computation speed with large sets of data, the standard method in use is gradient descent (or any optimizer derived from it, more information in Section 5.2.2.4).

ANN models can be trained in different frameworks:

- Supervised: The output of the model is compared with the real output (ground truth or dependent variable) by using a loss function. The discrepancy (loss) is used to tune the parameters in order to minimize the loss function value.
- Unsupervised. No ground truth is used to train the model. While this framework is used for clustering in machine learning models, unsupervised training is used in ANN for:
  - Autoencoders (proposed in the work by Kramer [1991]). The model input is used as the output, with the main idea being that the model learns to reconstruct the output.
  - Generative models (such as GAN Goodfellow et al. [2014]). A model is created from random noise that is able to generate synthetic data in such a way that a discriminator model cannot distinguish true data from synthetic data. In this case, the true data is used to train the discriminator and to compute the loss for the generator, but no ground truth is needed.
- Self-supervised. Little ground truth data is available and, because of this, creative ways of generating new data from the available data are used, i.e., data augmentation techniques such as modifying images by means of rotation, flipping them, or changing the direction of time in videos (i.e., playing the video backwards).
- Reinforcement learning. An agent (model) learns by interacting with an environment. From states and actions, rewards are computed and accumulated such that the agent develops a good policy towards an objective. The training of the model occurs through the enacted policy, and thus the agent maximizes the policy that grants it the maximum reward, i.e., the model inputs are the current state and actions, and it computes a function that approximates the policy that will provide the maximum reward.

5.2.1 ANN structure

The classical neural network model consists of an input layer, one hidden layer and one output layer (see Figure 5.2). This is known in the literature as a shallow neural network. The model is organized in layers that form an acyclic directed graph in which all nodes from preceding layers are connected to a node in the next layer. The input layer has an equal number of nodes serving as features in the data, with each node connected to all the nodes in the next layer (the hidden layer). The same organization is followed between the hidden layer and the output layer, where all the nodes in the hidden layer are connected to each of the nodes in the output layer. The number of neurons

in the hidden layer is a hyperparameter of the model (the width of the layer), but a reduced number of neurons in the hidden layer (a smaller number of neurons than in the preceding layer) forces the model to learn summarized features of the data, which can be interpreted as latent features. The output layer's number of neurons is constrained by the problem/data at hand; e.g., when facing a regression or a binary classification problem with a scalar dependent variable, only one node in the output layer is needed, as its output will consist of a scalar value. On the other hand, when the dependent variable is a vector like in a multiclass classification problem, then one node is needed for each of the classes.

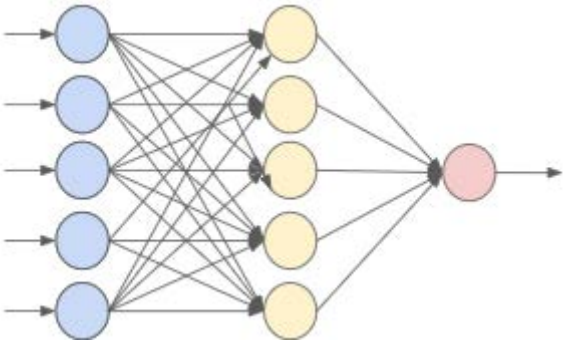


Figure 5.2: Shallow NN structure. Blue denotes the input layer, yellow the hidden units of the hidden layer, and red is the output layer.

MLP extends shallow neural network models, as they have multiple hidden layers (Figure 5.3) that constitute the model's depth. This enables the model to learn, in each hidden layer, new features based on the preceding layer's outputs, thus enhancing the model's capabilities as it learns complex features. This allows creating models that work as an end-to-end solution. This approach is not possible when using other machine learning methods, because these require manually creating new features to use as inputs, a process called "feature engineering". One of the main concerns about neural network procedures is that the features learned and how they are combined becomes obfuscated behind a large number of parameters and computations (and a lack of theory), thus causing them to be tagged as black boxes.

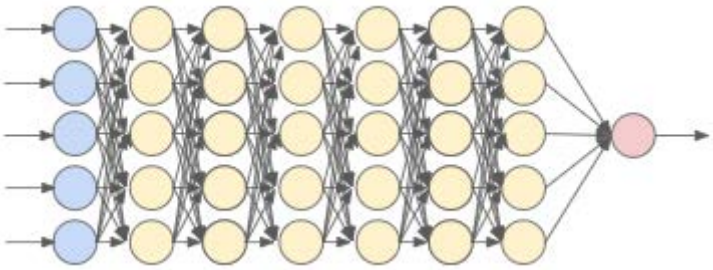


Figure 5.3: Multilayer perceptron structure. Blue denotes the input layer, yellow the hidden units of the hidden layer, and red is the output layer.

While only one input and output layer exist in the model and their numbers of nodes are somehow constrained by the data, the number of hidden layers is considered a hyperparameter of the model (the same happens with the number of neurons in each of the hidden layers), and its value cannot be known beforehand. Different approaches exist for finding the optimal value of the hyperparameters of a neural network model. Detailed information on these can be found in Section 5.2.2.7.

5.2.2 MLP

5.2.2.1 Computations

The computations for each node are shown in Equation 5.3. Each node in the hidden layer and output layer computes a weighted sum of their inputs (x), so the model parameters are the weights (w) and the bias (b) of this operation. Then, the result of the weighted sum is used as an argument of a nonlinear function called the activation function (σ). The ouput of the activation function (a) is the output of the neuron. Section 5.2.2.5 provides more information about the types of activation functions and their properties. The operations are schematized in Figure 5.4.

$$a = \sigma(\sum_{i=1}^{n} x_i * w_i + b) \tag{5.3}$$

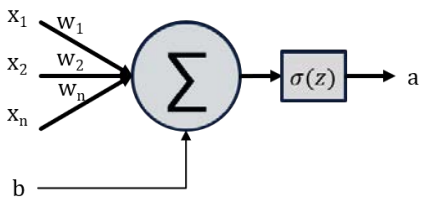


Figure 5.4: Computations at node level.

Zooming out and considering the operations between layers, the computed formula is shown in Equation 5.4 as a matrix formulation. Inputs are transformed by a linear operation using the weight matrix W, then translated by the bias vector **b** and, finally, a point wise transformation is applied (activation function **a**), thus giving nonlinearity to the operation. Here, it is made explicit that the computations for a hidden layer are projecting the features of the preceding layer into a new feature space, and the activation function plays a key role in this operation.

The goal is that the model learns to transform the input data through linear operations

and nonlinear transformation, so that the features at the output layer become linear.

$$\boldsymbol{a} = \sigma(W * \boldsymbol{x} + \boldsymbol{b}) \tag{5.4}$$

At the output layer, some considerations need to be taken into account, as the output of our model depends on the problem at hand. These considerations are:

- In a regression task, no activation function is needed, as the output layer is equivalent to a linear regression. The number of neurons must match the number of dimensions of the dependent variable.
- In a binary classification problem, a sigmoid activation function is the one to use, as it bounds the output to the interval [0,1]. This is equivalent to a logistic regression. The number of neurons must be one.
- In a multiclassification problem, the activation function to use is softmax, together with as many nodes as there are classes in the data, as this activation function maps the output from k nodes in such a way that the one with the higher value is close to 1 and the other values are closer to 0.

Also, one could instead consider using an output layer in order to directly use the outputs of the last hidden layer as input to any machine learning method such as support vector machine (SVM), k-nearest neighbors (KNN), etc.

Another point of consideration is the initial values of the parameters. The parameters of an ANN model can be initialized in a stochastic way by using either a normal or uniform distribution. However, it is necessary to take into account some considerations to achieve better results from the model and avoid problems such as exploding/vanishing gradients that usually appear when using either a large number of hidden layers or advanced architectures like recurrent neural networks (detailed information about this phenomenon is given in Section 5.2.2.9). In order to avoid such problems, it is necessary that the model's weights in each layer have a mean of 0 and a standard deviation of 1. With this in mind, there exist functions named initializers that provide a good initialization of the model's parameters, where the mean and the variance of the weights are constrained to each layer in the form of $W \sim \mathcal{N}(0, 1)$. They are:

- Glorot initialization (from Glorot and Bengio [2010]). The weights are initialized from a normal distribution such as $\boldsymbol{w} \sim \mathcal{N}(0, \sqrt{\frac{1}{fan\_in}})$, where $\boldsymbol{w}$ is the vector of weights for a model's neuron and fan_in is the number of inputs for the layer pertaining to the neuron. The bias parameters are initialized as $\boldsymbol{b} = 0$. This type of initialization has been empirically demonstrated to work well with classical activation functions such as the sigmoid and hyperbolic tangent (more information in Section 5.2.2.5).
- He initialization (from He et al. [2015]). The weights of the model are initialized from a normal distribution such as $\boldsymbol{w} \sim \mathcal{N}(0, \sqrt{\frac{2}{fan\_in}})$, where $\boldsymbol{w}$ is any weight

of the model and fan_in is the number of inputs to the current layer. The bias parameters are initialized as $\boldsymbol{b} = 0$. This initialization was designed with rectifier activation functions in mind. See Section 5.2.2.5 for detailed information.

Because the parameters of each node are randomly initialized, each one learns a different characteristic from the data at the end of the training.

5.2.2.2 Creating a model

The methodology for creating a model comprises three different stages, each of which uses a different part of the data. Data is split into three parts, where: the major part (usually 60%) is used for training; part of the remainder (30%) is used as the validation set; and what is left is used for testing the model by simulating new fresh data. Each stage of the model's creation consists of:

- Training. During this phase, the model parameters are adjusted according to the samples of the training data until achieving the minimum error possible.
- Validation. This data is used together with the training regime to tune the model hyperparameters.
- Testing. The last phase consists of evaluating the model goodness of fit to unseen data.

Some requirements in the data-splitting process are:

- All divisions of the data must be distributed similarly.
- Test data must be used as few times as possible. The optimal number of uses is one.
- A balanced number of samples is desirable for each category (if using categorical data) in the training set.

The reason for the data splitting is due to the nature of creating models in machine learning (and, hence, deep learning), as it consists of minimizing the value of a function (loss function) using a finite set of realizations from a process that generates infinite realizations. The goal is to use finite realizations in creating a model that can give accurate predictions based on modeled realizations that are not used in creating the model. This model process is known as generalizing the training data. The generalization of a model is constrained mainly by the quality of the data and by the model's complexity. If the data used to train the model is not representative enough, then we can expect the model's predictions to be biased in comparison with the expected value of the model's underlying process. This can be overcome by collecting more data from the underlying process. On the other hand, when dealing with complex models like neural networks with a large number of parameters and nonlinear transformations, the model parameters may be influenced by variance in the training data; and though it can give very accurate predictions about the training data, at the same time it will have a high error rate in the data that is not used for training. At the end, it is necessary to control the

degree to which the model parameters are fitted to the training data in order to control its generalization capabilities. In the case of neural networks, model complexity can be constrained using regularization techniques like dropout Srivastava et al. [2014] and early stop (more about regularization techniques in Section 5.2.2.8). Models with simple complexity (e.g., linear regression) tend to have high bias and low variance, while models with large complexity (e.g., neural networks) usually have low bias and high variance. This is known as the bias-variance trade-off, and detailed information is offered in Bishop [2006]. Thus, in order to assess the generalization capabilities of the model and detect overfitting in the training data, a test data set is used. Moreover, because of the large number of hyperparameters in a neural network, those are adjusted to the test set and become another way of overfitting the parameters to the data (this time to the training and test set). In order to prevent this, the model is evaluated after training and the hyperparameters are adjusted by means of the training and validation sets, thus limiting the test to evaluating the model based on unseen data.

5.2.2.3 Training

The training (or fitting) of the parameters takes place in two steps: forward propagation and backward propagation. The former carries the necessary computations from the inputs to the outputs, while the latter performs the computations needed for optimizing the model parameters. The backward propagation part (also called backprop) is driven by the backpropagation algorithm Rumelhart et al. [1986], Linnainmaa [1976].

In the forward propagation, the computations explained in Section 5.2.2.1 are applied through the model. Then, the output from the output layer is compared to the ground truth (supervised training, as mentioned in Section 5.2) through a loss function (further explanations on loss functions in Section 5.2.2.6). The objective of the model is to minimize the loss function, and the only way to accomplish this is through changes in the parameters (weights and bias) of the model.

The loss value uses the backprop algorithm as the starting point for the back propagation phase. The idea is to change the parameter values in such a way that the loss is minimized. In order to do so, it is necessary to compute the gradient of the loss function with respect to all the model parameters. Because all the computations on the feed-forward pass are continuous (including the activation functions), this can be accomplished using the chain rule of derivatives. Once the gradient is computed (Equation 5.5), the model parameters are updated using the gradient descent updating rule shown in Equation 5.6:

$$\boldsymbol{g} = \frac{1}{n}\nabla_\theta \sum_{i=1}^{n} l(f_\theta(\boldsymbol{x}_i;\boldsymbol{\theta}), y_i) \tag{5.5}$$

where ∇_θ is the gradient of the loss function l with respect to the parameters θ of the model. The arguments of l are the output of the model f_θ, with input $\boldsymbol{x}_i$ and the ground

truth y_i. The output $\boldsymbol{g}$ is the mean value of the gradient for the n data samples:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \boldsymbol{g} \tag{5.6}$$

where α is the learning rate (hyperparameter of the model, which usually uses a default value of 0.01), θ is the model parameters and $\boldsymbol{g}$ is the gradient computed using the previous Equation 5.5.

In practice, using the whole data set is not computationally feasible due to modern datasets being composed of possibly millions of data samples. In order to alleviate this, data is divided into mini-batches of size $m = 2^n$ that can fit into CPU and GPU memory. This characteristic of dividing the data set trains the data in three different ways:

- Batch gradient descent. All data samples are used simultaneously to compute the gradient and update the values of the model parameters. It is computationally costly, but the gradient estimation is less affected by noise.
- Stochastic gradient descent. For each data sample, the gradient is computed and the parameters are updated. It is fast, but the computed gradient at each step is subject to data noise.
- Mini-batch gradient descent. This is a trade-off between both of the previously explained methods, where "mini" batches of data with size 2^n are used to estimate the gradient and update the model parameters.

Regardless of the chosen strategy for computing the gradient and updating the parameters, the whole data set is used to train the model more than once. Each time that all the data samples are digested by the model during training is known as an epoch. Through the training of the model, an arbitrary number of epochs are needed to reach convergence into a minimum of the loss function. The number of epochs is conditioned by the data and the problem to solve is a hyperparameter of the model. A common strategy is to set a large number of epochs and define a gap between the training and the validation error, with the purpose of reaching an equilibrium between overfitting and generalization. Details about this technique are given in Section 5.2.2.8.

After a sufficient number of epochs, the loss function will achieve a minimal value. In fact, this minimal value is a local minima of the loss function due to the nonlinearities introduced by the function estimated by the model. However, research (Choromanska et al. [2015], Li et al. [2018]) shows that local minima in highly dimensional spaces can be considered good enough in the sense that their values do not differ too much between each other and, by generalization, nor do they differ from the global optimal value.

After each epoch, the model is evaluated on the basis of the validation set in order to assess and control its overfitting in the training data. Once the model is created, its goodness of fit is evaluated with unseen data from the test data set.

5.2.2.4 Optimizers

As mentioned in the previous section, the model parameters are fitted with the backpropagation algorithm, which, at its core, uses the gradient descent algorithm. In modern deep learning methods, it is common to use an evolution of the gradient descent, namely optimizers. This provides different strategies for improving aspects of the original gradient descent, like speed, avoiding saddle points and convergence of the loss function towards better local minima. In order to introduce the reader to the topic, this section provides short summaries of the relevant aspects of the most common optimizers in the literature. Detailed explanations about these evolutions are given in Ruder [2016].

Taking as Equation 5.5 as a starting point, the following methods have been proposed in the literature:

- *Momentum.* Its objective is to speed up convergence towards the minimum and avoid saddle points. In order to accomplish this, we use a velocity $\boldsymbol{v}$ (a matrix for the weights and a vector for the bias), whose values are updated using an exponential weighted moving average of past gradients. At each back propagation step, the velocity gets updated using Equation 5.7 and thus serves as a trade-off between the past and current values of the gradient vector. When the direction of the gradients in consecutive steps is the same, the velocity values get higher as past gradients are accumulated. Then, the parameters are updated with a larger value in the update step (Equations 5.6 and 5.8):

$$\boldsymbol{v_t} = \lambda * \boldsymbol{v}_{t-1} - \alpha * \boldsymbol{g} \tag{5.7}$$

  where $\boldsymbol{v_t}$ is the velocity vector at the current step t, $\boldsymbol{v}_{t-1}$ is the previous step velocity vector, λ is the momentum hyperparameter or weight decay (by default, its value is 0.9) and $\boldsymbol{g}$ is the gradient vector.

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \boldsymbol{v}_t \tag{5.8}$$

  where α is the learning rate, $\boldsymbol{\theta}_{t-1}$ is the previous values of the parameters and $\boldsymbol{v}_t$ is the current values of the velocity vector.

- *Nesterov Accelerated Gradient.* This is a modification of the momentum method, in which the velocity in the last step is used to compute a prediction of the parameter values (Equation 5.9). Then, the gradient is computed using these new parameters (Equation 5.10). The gradient will then be used to update the velocity (Equation 5.11), and from it the current values of the parameters are computed (Equation 5.12). The effect of these operations is a compromise between velocity and current gradient directions.

$$\tilde{\boldsymbol{\theta}}_t = \boldsymbol{\theta}_{t-1} + \lambda \boldsymbol{v}_{t-1} \tag{5.9}$$

  where $\tilde{\boldsymbol{\theta}}_t$is the prediction of the parameter values based on their previous values

$\boldsymbol{\theta}_{t-1}$ and the past velocity vector $\boldsymbol{v}_{t-1}$.

$$\boldsymbol{g} = \frac{1}{m}\nabla_{\tilde{\boldsymbol{\theta}}}\sum_{i=1}^{m} l(f(\boldsymbol{x}_i;\tilde{\boldsymbol{\theta}}), y_i) \tag{5.10}$$

where $\boldsymbol{g}$ is the gradient vector, m is the number of elements in the current mini-batch, f is the model with input $\boldsymbol{x}_i$ and parameters $\tilde{\boldsymbol{\theta}}$, l is the loss function and y_i is the ground truth output for the input $\boldsymbol{x}_i$.

$$\boldsymbol{v}_t = \lambda * \boldsymbol{v}_{t-1} - \alpha * \boldsymbol{g} \tag{5.11}$$

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \boldsymbol{v}_t \tag{5.12}$$

Another aspect that has been the subject of research into improving it is the learning rate. While in previous methods the learning rate maintained a constant value and this value was commonly reduced as the gradient descent algorithm moves further along in the iterations, modern optimizers use adaptive learning rates based on the magnitude of change in the gradient directions. Furthermore, instead of representing the learning rate as a scalar value, it is represented as a vector of learning rates for each dimension of the gradient (i.e., for each parameter of the model). The methods that use adaptive learning rates are:

- *Adaptive Gradient Descent (Adagrad).* This consists of accumulating the squared values of the gradient and using them to compute learning rates that will provide a great decrease in large gradient values and little decrease in the smaller ones. The effect is a trade-off between smaller and larger gradients. First, the current gradient step is computed following Equation 5.5, the squared gradient is accumulated using Equation 5.13 and then the update for the gradients is computed with Equation 5.14. Lastly, the values of the parameters are updated by using Equation 5.15. One of the major drawbacks of the Adagrad method is that the accumulated gradient gets bigger and bigger at each step, making the learning rates very small before arriving at a minimum of the loss function.

  $$\boldsymbol{r}_t = \boldsymbol{r}_{t-1} + \boldsymbol{g} \odot \boldsymbol{g} \tag{5.13}$$

  where the term $\boldsymbol{r}$ is the accumulated squared gradient at the current step t, and $\boldsymbol{g}$ is the current gradient.

  $$\Delta\boldsymbol{\theta} = -\frac{\boldsymbol{\alpha}}{\sqrt{\boldsymbol{r}_t} + \boldsymbol{\delta}} \odot \boldsymbol{g} \tag{5.14}$$

  where $\boldsymbol{\alpha}$ is the learning rate, $\boldsymbol{\delta}$ is a constant to avoid numerical issues and $\boldsymbol{r}_t$is the

accumulated squared gradient at the current step.

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \Delta\boldsymbol{\theta} \tag{5.15}$$

where $\boldsymbol{\theta}_{t-1}$ are the previous values of the parameters and $\Delta\boldsymbol{\theta}$ is the parameter update.

- *Root Mean Square Propagation (RMSprop):* This method overcomes the commented drawback of Adagrad using an exponential moving average of the previous squared gradients. With this approach, far away gradients at the current step have less influence when computing the learning rates solving the problem of not arriving at a minimum due to very small learning rates. The RMSprop method starts from the computed gradient using Equation 5.5, then, the accumulated gradient is computed using Equation 5.16. The change of the parameters is computed using Equation 5.17 and then applied to the current parameters values in order to compute the new values using Equation 5.19. Gradients can be computed using the NAG method, where they are computed using Equation 5.10, in this case, Equation 5.17 is replaced by Equation 5.18 and Equation 5.19, by Equation 5.20.

$$\boldsymbol{r}_t = \rho * \boldsymbol{r}_{t-1} + (1-\rho)\boldsymbol{g}_t \odot \boldsymbol{g}_t \tag{5.16}$$

where $\boldsymbol{r}_t$ is the accumulated exponential moving average of the squared gradient at the current step t, ρ is the decay rate (hyperparameter of the model, usually with a default value of 0.9), and $\boldsymbol{g}_t$ is the gradient computed at the current step.

$$\Delta\boldsymbol{\theta} = -\frac{\boldsymbol{\alpha}}{\sqrt{\boldsymbol{r}_t}+\boldsymbol{\delta}} \odot \boldsymbol{g}_t \tag{5.17}$$

where $\boldsymbol{\alpha}$ is the learning rate, $\boldsymbol{\delta}$ is a constant to avoid numerical issues, $\boldsymbol{r}_t$is the accumulated squared gradient at the current step and $\boldsymbol{g}_t$ is the current gradient step.

$$\boldsymbol{v}_t = \lambda * \boldsymbol{v}_{t-1} - \frac{\boldsymbol{\alpha}}{\sqrt{\boldsymbol{r}_t} + \boldsymbol{\delta}} \odot \boldsymbol{g}_t \tag{5.18}$$

where $\boldsymbol{v}_t$ is the velocity at the current step t, λ is the decay parameter for the past velocity values, $\boldsymbol{\alpha}$ is the learning rate, $\boldsymbol{\delta}$ is a constant to avoid numerical issues, $\boldsymbol{r}_t$is the accumulated squared gradient at the current step and $\boldsymbol{g}_t$ is the current gradient step.

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \Delta\boldsymbol{\theta} \tag{5.19}$$

where $\boldsymbol{\theta}_{t-1}$ are the previous values of the parameters and $\Delta\boldsymbol{\theta}$ is the parameter update.

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \boldsymbol{v}_t \tag{5.20}$$

where $\boldsymbol{\theta}_{t-1}$ are the previous values of the parameters and $\boldsymbol{v}_t$ is the parameter update through the current step velocity values.

- *Adaptive Moment Estimation (Adam).* This method defines momentum and the accumulated squared gradient as first- and second-order moments (hence the name adaptive moment estimation), and it further gives a solution to the biased estimate in the first steps of their estimation. The method begins with the gradient computed from a mini-batch of m examples using Equation 5.21. Then, the biased first-order moment and biased second-order moment are computed using, respectively, Equations 5.22 and 5.23. Then, both moments are bias-corrected using Equations 5.24 and 5.25. The parameter change is computed through Equation 5.26 and is then applied in order to update the parameter values using Equation 5.27.

$$\boldsymbol{g} = \frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_{i=1}^{m} l(f(\boldsymbol{x}_i; \boldsymbol{\theta}), y_i) \tag{5.21}$$

where $\boldsymbol{g}$ is the gradient vector, m is the number of elements in the current mini-batch, f is the model with input $\boldsymbol{x}_i$ and parameters $\boldsymbol{\theta}$, l is the loss function and y_i is the ground truth output for the input $\boldsymbol{x}_i$.

$$\boldsymbol{s}_t = \rho_1 * \boldsymbol{s}_{t-1} + (1 - \rho_1)\boldsymbol{g} \tag{5.22}$$

where $\boldsymbol{s}_t$ is the biased first-order moment, ρ_1 is the decay hyperparameter for past first-order moments and $\boldsymbol{g}$ is the gradient at the current step.

$$\boldsymbol{r}_t = \rho_2 * \boldsymbol{r}_{t-1} + (1 - \rho_2) * (\boldsymbol{g} \odot \boldsymbol{g}) \tag{5.23}$$

where $\boldsymbol{r}_t$ is the biased second-order moment at the current step t, ρ_2 is the decay hyperparameter for past second-order moments and $\boldsymbol{g}$ is the gradient at the current step.

$$\hat{\boldsymbol{s}}_t = \frac{\boldsymbol{s}_t}{1 - \rho_1^t} \tag{5.24}$$

where $\hat{\boldsymbol{s}}_t$ is the unbiased first-order moment, $\boldsymbol{s}_t$ is the biased first-order moment, ρ_1 is the decay hyperparameter for past first-order moments and t is the current step.

$$\hat{\boldsymbol{r}}_t = \frac{\boldsymbol{r}_t}{1 - \rho_2^t} \tag{5.25}$$

where $\hat{\boldsymbol{r}}_t$ is the unbiased second-order moment, $\boldsymbol{r}_t$ is the biased second-order moment, ρ_2 is the decay hyperparameter for past first-order moments and t is the

current step.

$$\Delta\boldsymbol{\theta} = -\boldsymbol{\alpha}\frac{\hat{\boldsymbol{s}}_t}{\sqrt{\hat{\boldsymbol{r}}_t} + \boldsymbol{\delta}} \tag{5.26}$$

where $\Delta\boldsymbol{\theta}$ is the update for the parameters, $\boldsymbol{\alpha}$ is the vector of learning rates, $\hat{\boldsymbol{s}}_t$ is the unbiased first-order moment, $\hat{\boldsymbol{r}}_t$ is the unbiased second-order moment and $\boldsymbol{\delta}$ is a small constant used for numerical stability.

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \Delta\boldsymbol{\theta} \tag{5.27}$$

where $\boldsymbol{\theta}_t$ is the new values for the parameters at the current step t and $\Delta\boldsymbol{\theta}$ is the update for the parameters.

Nowadays, deep learning practitioners usually make use of RMSProp and Adam optimizers as the default go-to algorithm for training their models.

5.2.2.5 Activation functions

Activation functions are the components that enable NN models to learn nonlinear patterns in the data by applying element-by-element nonlinear mapping to the outputs of the nodes. Traditionally, these functions have been monotonic, continuously differentiable and smooth with a monotonic derivative, but recent proposals like the rectifier family of activation functions do not fulfill some of these requirements, although they are able to provide properties that make their use desirable.

The most common activation functions are: the logistic sigmoid function defined in Equation 5.28, whose derivative is presented in Equation 5.29; the hyperbolic tangent function (tanh) shown in Equation 5.30, whose derivative is Equation 5.31; and the rectified linear unit (ReLU), whose formulation is presented in Equation 5.32 and its derivative is presented in Equation 5.33. Figures 5.5, 5.6 and 5.7 show the visualization of the common activation functions and their derivatives.

$$g(z) = \frac{1}{(1 + e^{(-z)})} \tag{5.28}$$

$$g'(z) = g(z) * (1 - (g(z)) \tag{5.29}$$

$$g(z) = \frac{(e^{(z)} - e^{(-z)})}{(e^{(z)} + e^{(-z)})} \tag{5.30}$$

$$g'(z) = 1 - g(z)^2 \tag{5.31}$$

$$g(z) = \max(0, z) \tag{5.32}$$

$$g'(z) = \begin{cases} 1 & \text{if } z >= 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.33}$$

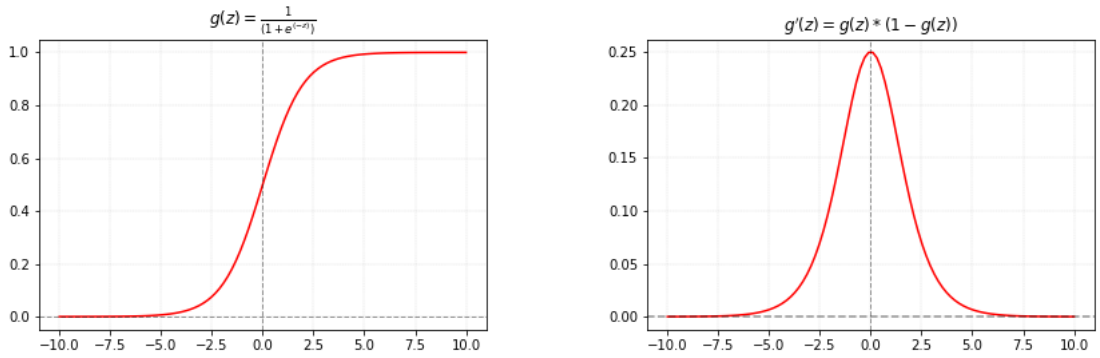


Figure 5.5: Logistic sigmoid function and its derivative.

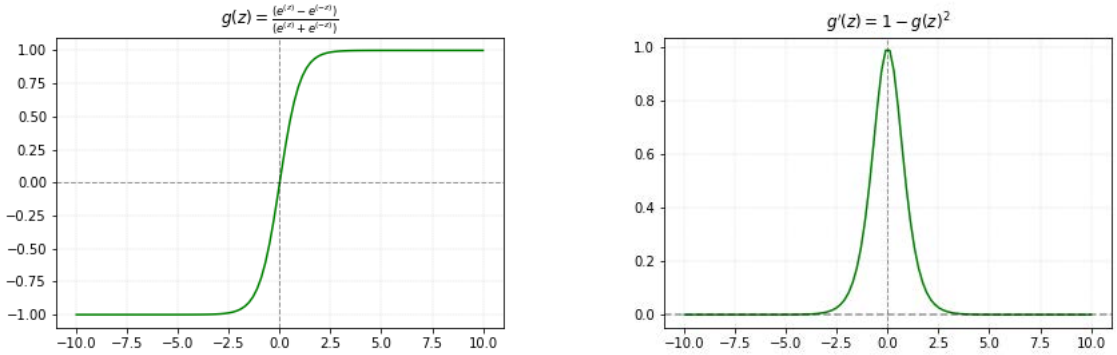


Figure 5.6: Hyperbolic tangent function and its derivative.

Problems when using functions like the logistic sigmoid and the hyperbolic tangent were detected and analyses in Hochreiter [1991], and these had an effect on the capacity of NN models to learn. The cause of the problems was detected in the derivatives of the activation functions used, as when the output of a node is very small or large, the logistic

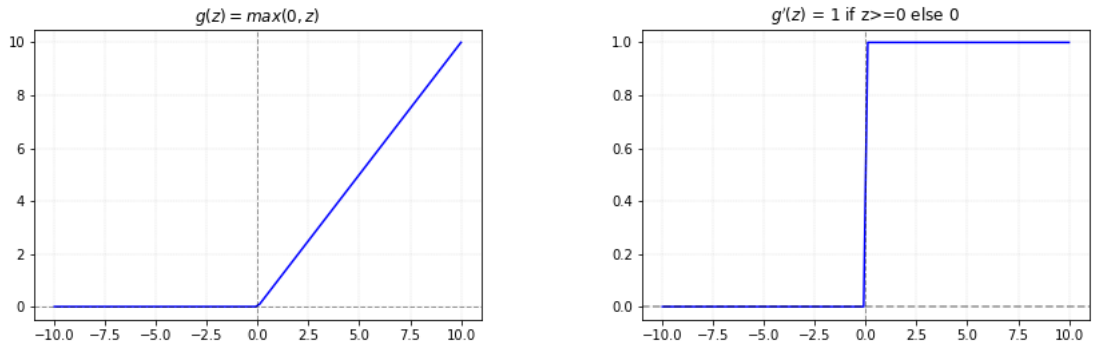


Figure 5.7: Rectified linear unit function and its derivative.

sigmoid (and, to a lesser extent, the tanh function also) gradients become saturated as their values grow very small or reach 0. This makes the training of the model very slow or even halts it. Because of this, it is preferable to use piecewise linear units like the rectified family of functions, to which the ReLU function belongs. ReLU was proposed in Glorot et al. [2011] to overcome the problem of oversaturated nodes and also to introduce sparsity into the models, as nodes that do not play a role (their output is 0) become inactive because their gradient will also be 0 (a phenomenon called dying ReLU); but for active nodes, the gradient will be 1, allowing the models to be trained faster. When using ReLU, it is recommended to use an initial value of 0.1 for the bias, as this avoids starting with inactive nodes. Notice that the ReLU derivative is not continuous at $z = 0$, which can be solved by assigning 0 or 1 to the output (depending on the software implementation).

Another common activation function that is not usually used in the hidden layers is the softmax activation function. Softmax nodes represent a probability distribution (as the sum of its outputs is 1) over a categorical variable of k categories; and, thus, it is commonly used at the output layer in conjunction with the cross-entropy loss function. It is also commonly used for multiclass classification and serves as a generalization of logistic regression models (based on the logistic sigmoid function). When softmax is used in an NN context, it is considered as a special layer called the softmax layer. A schema of the softmax layer can be seen in Figure 5.8.

The softmax function (Equations 5.34 and 5.35) maps an input vector to another vector, where its original values are squashed towards 0 – with the exception of the higher value, which is squashed towards 1. Also, the sum of the elements at the output equals 1 (as a probability distribution). This effect is a form of pointing to the higher element of the input vector and, as such, becomes something of an argmax function. Because of this, the softmax is considered a soft version of the argmax function, but with the possibility

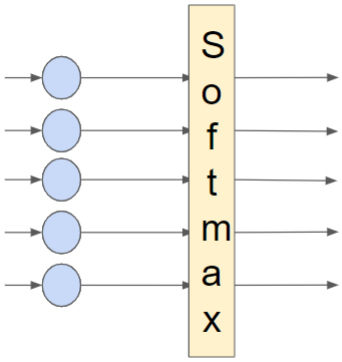


Figure 5.8: Softmax layer.

of letting the gradients flow because it has a continuous and smooth derivative.

$$softmax(a_i) = \frac{e^{a_i}}{\sum_{j=1}^{N} e^{a_j}} \tag{5.34}$$

where a_i is an element of the vector $\boldsymbol{a}$ and N is the number of vector elements.

$$\frac{\partial softmax(a_i)}{\partial a_j} = softmax(a_i) * (\delta_{ij} - softmax(a_j)) \tag{5.35}$$

where δ_{ij} is the Kronecker delta.

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

5.2.2.6 Loss functions

The goal of a loss function is to compute the deviation between the model output (given a specific input) and the real (or expected) output. Because the parameters of an NN are initialized randomly (see Section 5.2.2.1), the initial outputs of an NN start to become very different from the expected outputs, resulting in a large value from the loss function. As the training proceeds, the parameters are adjusted in order to minimize the loss function outputs.

Similar to activation functions, loss functions for NN must have the following properties:

- Their output is 0 when the model and expected outputs are equal.
- Their values are greater as the model outputs deviate more from the expected outputs.
- Smooth changes in the outputs when there are small changes in the inputs.

Loss functions are constrained by the objective of the problem, and thus they are different if the context is, for example regression, classification or metric learning.

In a regression context, common loss functions used are the L1 distance, whose formulation is presented in Equation 5.36, and the L2 distance, whose formulation is presented in Equation 5.37:

$$\mathcal{L}(x_i, y_i) =\mid y_i - f_\theta(x_i) \mid \tag{5.36}$$

where y_i is the expected output and $f_\theta(x_i)$ is the model output when the input is x_i.

$$\mathcal{L}(x_i, y_i) = (y_i - f_\theta(x_i))^2 \tag{5.37}$$

where y_i is the expected output and $f_\theta(x_i)$ is the model output when the input is x_i.

In a classification context, the most commonly used loss function is cross-entropy loss (when used together with a softmax activation at the output layer; if not, its name becomes negative log-likelihood loss). It originates from information theory, and it combines the entropy of the true distribution (defined in Equation 5.38) with the Kullback Liebler divergence (Equation 5.39), and the true distribution with an approximated distribution. In the NN context, the true distribution is fixed and represented through one-hot encoding of the expected category (or label), given an input. The approximated distribution is the model output with the same input. Remember that, by fitting the parameters, the second distribution changes in order to behave more like the true distribution, which remains fixed. While the entropy provides the expected number of information units to communicate the distribution p, the cross-entropy (Equation 5.40) provides the number of information units needed to communicate the distribution p by using the distribution q. Thus, the closer they are, the smaller the number of information units needed. Because these distributions in the NN context are seen as one-hot encoders for realizations of categorical variables, cross-entropy loss can be simplified, as shown in Equation 5.41. From p, only one entry has a value of 1 (the one representing the category) and the other entries have a value of 0.

$$H(p) = -\sum_{i=1}^{n} p(x_i) * log(p(x_i)) \tag{5.38}$$

where p is a probability distribution, x_i is a realization of a random variable X and n is the number of samples.

$$D_{KL}(p \parallel q) = \sum_{i=0}^{n} p(x_i) * log(\frac{p(x_i)}{q(x_i)}) \tag{5.39}$$

where p and q are probability distributions, x_i is a realization of a random variable X

and n is the number of samples.

$$H(p,q) = H(p) + D_{KL}(p \parallel q) = -\sum_{i=0}^{n} p(x_i) * log(q(x_i)) \tag{5.40}$$

where p and q are probability distributions, x_i is a realization of a random variable X and n is the number of samples.

$$H(p,q) = -log(q(x_i)) \tag{5.41}$$

where p and q are probability distributions, x_i is a realization of a random variable X and n is the number of samples.

5.2.2.7 Hyperparameter optimization

NN models are known for their large numbers of hyperparameters defining the models. These cannot be learnt from data , and thus it is necessary to use meta-learning approaches or heuristics. Examples of hyperparameters are the number of hidden layers, number of neurons per layer, type of activation function, which optimizer to use, and optimizer parameters, among others. Because the hyperparameter defines the architecture of the model, the term neural architecture search has been coined as the research area into methods of finding the best NN architecture for a given problem. Common strategies proposed in the literature are:

- Random search. Proposed in the work by Bergstra and Bengio [2012], this consists of randomly trying different combinations of the hyperparameters for a number of n iterations and comparing the model results in the validation set for each iteration. In 60 iterations, there is a 95% probability that a good model arises from this approach.
- Neural architecture search by reinforcement learning. Proposed in the work by Zoph et al. [2016], this consists of letting a model learn through reinforcement learning how to create other models by combining the different parts of an NN model. As a result, the model learns to create NN models with high accuracy.
- Neural architecture search by evolution. Presented in the work by Real et al. [2017], the idea here is to use evolutionary algorithms to discover network architectures that work best for a given problem.

5.2.2.8 Regularization

As mentioned in Section 5.2.2.2, regularization techniques are needed for fitting the parameters when dealing with NN models, as their great capabilities for adaptation can prevent the model from generalizing due to the training data being overfitted. Regularization was first proposed to overcome ill-posed problems, which arise in statistics when

the matrix of regressors present collinearity (linear dependence on their vectors). More recently, regularization is used in any technique that increases a model's capabilities of completing a task based on unseen data. Different approaches are used in NN to prevent overfitting:

- Parameter norm penalties. Controlling the magnitude of the parameters by adding a term to the loss function. This term is also called the regularization term. One of the more common regularization techniques used in regression is the Tikhonov regularization (L^2 norm), which consists of adding the term $\lambda(\boldsymbol{w}^T\boldsymbol{w})$, where $\boldsymbol{w}$ is the vector of the parameters. The idea is to control the norm of the weights through the regularizer hyperparameter λ.
- Data augmentation. When dealing with inputs such as images, it is possible to increase the input data by manipulating it. This acts as a regularization technique, as more available data makes the model able to learn from more examples , and thus it becomes better at working with unseen data. This is one of the main methods used in self-supervised learning.
- Dropout. Proposed in Srivastava et al. [2014], this consists of randomly deactivating neurons (multiplying their output by zero) in each layer during their training in order to make the NN model become a bagged ensemble of an exponentially greater number of neural networks (a technique proposed in Breiman [1994]). Thus, instead of a large NN model, a model of submodels is obtained.
- Early stop. This consists of stopping the iterative procedure of model fitting and is based on the increase in error between training and the validation set used during training, as this is the most common sign of overfitting in the training data. Detailed explanations on its effect as a regularization technique is explained in Bishop [2006].

5.2.2.9 The vanishing gradient problem

One problem that arises in very deep NN models and some specialized architectures like the recurrent neural networks (RNN, Section 5.2.3) is the problem of vanishing and exploding gradients. As the number of matrix operations (multiplications) become larger in the forward pass, it is possible that the gradient in the backward pass suffers from one of the following issues:

- Vanishing gradient. The recurrent matrix multiplications cause the gradients to become very small; thus, the update of the weights at each iteration also becomes very small and therefore leads to the training procedure becoming very slow or impossible.
- Exploding gradient. The recurrent matrix multiplications cause the gradients to become very large; thus, the update of the weights at each iteration also becomes very large and therefore leads to the model being unable to learn.

This effect happens because the largest eigenvalue in the matrix of weights becomes very small or very large when the optimal is to maintain a value near one at each of the successive multiplications. Overcoming this requires using aforementioned techniques like weight initialization, rectified linear unit activation functions, optimizers, etc. Also, in the case of the exploding gradient, techniques like clipping gradient are used, where the magnitude of the gradient is limited to a certain value in order to prevent it from exponentially increasing (usually dividing the gradient by its magnitude if a threshold is exceeded). The work by Hochreiter [1991] presents a deep analysis of the problem.

5.2.3 Recurrent NN: Long short-term memory and gated recurrent unit

When modeling sequence data using NN methods, MLP falls short because it is not capable of dealing naturally with some of the characteristics of this kind of data. The main features that characterize sequence data are:

- Elements of a sequence have a meaning based on order, causing an element's value to be determined by the elements that precede and/or succeed it.
- Sequence data is not of fixed length, as what happens in the case of natural language processing.

One possible way to model such data with MLP is to use a sliding window. Problems arise when applying such methods, as it is difficult to know the optimal window size. Also, a window of fixed size constrains the model to detecting only patterns that occur within the window span. Thus, one could consider using a very large window, let us suppose of size n, for which each neuron in the first hidden layer would need n parameters, thus making the problem computationally unfeasible. Moreover, if the sequences are of variable length, then there will be problems with the endings of the sequences, as some special value will be needed to fulfill the sequence length.

Recurrent neural networks (RNN) are NN models whose goal is to model sequence data and provide natural fixtures that deal with the characteristics of this type of data. RNNs are characterized by sharing their weight parameters across time. This is accomplished by connecting the neurons to themselves, thus forming cycles in their graph. This enables the model to capture time dependencies, as neurons receive new input and the previous neuron's output at each time step, then use these to provide the current output. Figure 5.9 shows a folded RNN unit (with a cycle) and the RNN unit unfolded in time.

The loop formed by connecting the RNN units to themselves enables the model to have some sort of memory, which is called "state" or "hidden state". The sequential information is preserved in the RNN's hidden states, and the new samples that are processed have a cascade effect. This cascade allows the RNNs to determine relationships between contiguous and non-contiguous events in the sequence. The hidden state is used by the RNN units as a summary of the relevant aspects of the past sequences up to the current step t. Figure 5.10 shows an RNN model with one input layer, one hidden layer

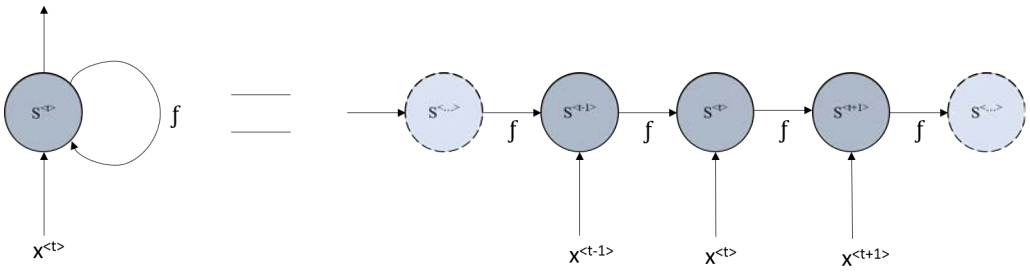


Figure 5.9: RNN unit.

and one output layer. Each element of the input layer is considered a sequence. Because of this, the expected input of recurrent models are tensors of three dimensions, where their components are defined by the number of samples (batch of data), the sequence length, and the number of different types of sequences that are considered (features).

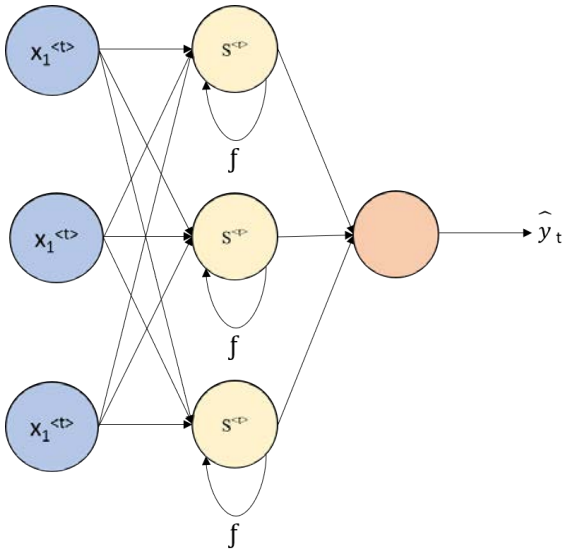


Figure 5.10: RNN structure.

One of the advantages of using a parameterized hidden state is that it is a random variable, which allows RNN models to provide an efficient parameterization of the joint distribution of observations. On the other hand, it must be taken into account that this leads to the assumption that the same parameters can be used for different time steps. Thus, the conditional probability distribution of the input variables at time $t+1$, given variables at time t, is stationary (stationarity has been introduced in 3.2).

5.2.3.1 RNN computations

RNN computations differ slightly from those presented in Section 5.2.2.1. Although there exist different procedures for using RNN models, this section focuses on presenting

a possible approach to time series data.

At each time step, a row of data that represents the previous value sequence until the current step (included) is submitted to the RNN. Then, the output is compared (using the loss function) with the next value in the sequence in order to adjust the parameters. The weight of each arc that connects the neurons to themselves preserves the information from the past sequences , and thus the output at the current step will take into account the previous information. At the initial step, the hidden state of all RNN units has a value of 0. Then, at each step, their value is updated based on the previous states and the current inputs. The computations in an RNN unit are shown in Equations 5.42 and 5.43, where the former shows how the hidden state is computed at the current time step, and the latter is the computation made for obtaining the output of the RNN unit based on the current hidden state.

$$\boldsymbol{h}_t = \sigma(W * \boldsymbol{h}_{t-1} + U * \boldsymbol{x}_t + \boldsymbol{b}) \tag{5.42}$$

where $\boldsymbol{h}_t$ is the current hidden state, W is the matrix of weights for the last hidden state that is represented as $\boldsymbol{h}_{t-1}$, $\boldsymbol{U}$ is the matrix of weights for the inputs at the current time step that is represented as $\boldsymbol{x}_t$, $\boldsymbol{b}$ is the bias parameters and σ any activation function (usually, tanh is used).

$$\hat{\boldsymbol{y}}_t = V * \boldsymbol{h}_t + \boldsymbol{c} \tag{5.43}$$

where $\hat{\boldsymbol{y}}_t$ is the output of the RNN unit at the current step t, h_t is the current hidden state, V is the weights of the output layer and $\boldsymbol{c}$ is a bias parameter.

5.2.3.2 Training in RNN

The training procedure of an RNN model is the same as the one presented for MLP models (Section 5.2.2.3), with a slight modification to the computations because RNN requires taking into account the hidden-to-hidden recurrent connections. The modification is needed for the computations because the current output in RNN depends not only on the current input, but also on past hidden states. Therefore, the gradient also flows in time as it computes the contribution to the loss at each time step. Because it travels through time, the name of the algorithm is recoined as backpropagation through time (BPTT).

One of the major problems with BPTT is that longer sequences can result in the gradient vanishing or exploding easily, similarly to what happens in very deep MLP. This effect is explained in the previous Section 5.2.2.9. In order to overcome the problem of vanishing/exploding gradients, the literature proposes other methods based on RNN, of which the most used are long short-term memory (LSTM) and gated recurrent unit (GRU).

5.2.3.3 LSTM

LSTM NN is one of the first popular proposals for solving the problem with long-term dependencies. It was introduced in Hochreiter and Urgen Schmidhuber [1997], where the vanishing problem is presented in detail, as well as how to overcome it using gates that control the hidden state of the model units. GRU NN (proposed in Cho et al. [2014]) is an evolution of LSTM in which the structure is fine-tuned in order to create less complex models with faster computations.

In the LSTM proposal, the cells (the name given to the units due to the greater complexity of the computations compared to those of traditional RNN units) are composed of a mechanism of gates (operations) that control the flow of information in the hidden state. The cells also incorporate another flow of information known as the cell state. The cell state is used to maintain the flow of the most useful information up to the current step. The use of the cell state and the hidden state is why the model is named long short-term memory, as the former acts as the long-term memory and the latter as the short-term memory. The full schema of an LSTM cell can be seen in Figure 5.11.

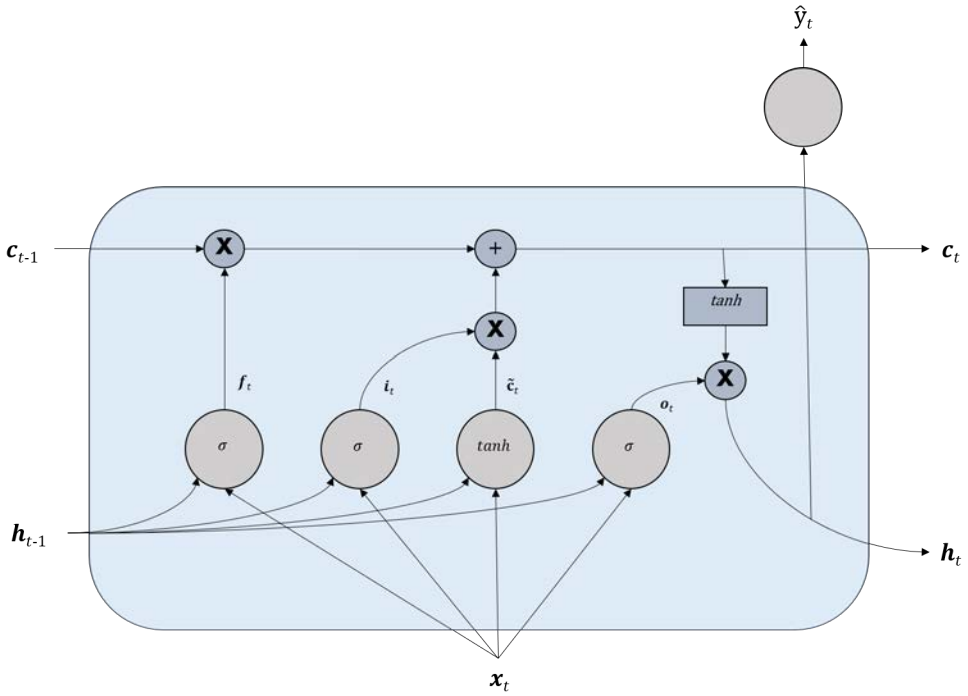


Figure 5.11: LSTM cell.

The inner computations of an LSTM cell are designed to control the cell state through a gating system. This system uses linear interactions to modulate the flow of information inside the cell based on the condition of the hidden state and the current sample. Gates consist of computations between the hidden state and the current input, followed by a sigmoid activation function. This guarantees that the outputs behave similarly to a

step function, as their value will be between 0 and 1. In an LSTM cell, there are three different gates, each with a specific goal:

- The forget gate. This computes which parts of the last cell state need to be forgotten, based on the current input and previous hidden state.

$$\boldsymbol{f}_t = \sigma(U^f * \boldsymbol{x}_t + W^f * \boldsymbol{h}_{t-1} + \boldsymbol{b}^f) \tag{5.44}$$

  where U^f is the matrix of weights of the forget gate for the current input $\boldsymbol{x}_t$, W^f is the matrix of weights of the forget gate for the previous hidden state $\boldsymbol{h}_{t-1}$, and $\boldsymbol{b}^f$ is the bias vector of the forget gate. σ denotes the sigmoid activation function.

- The input gate. This decides which parts of the current input and previous hidden state are relevant in order to update the current cell state.

$$\boldsymbol{i}_t = \sigma(U^i * \boldsymbol{x}_t + W^i * \boldsymbol{h}_{t-1} + \boldsymbol{b}^i) \tag{5.45}$$

  where U^i is the matrix of weights of the input gate for the current input $\boldsymbol{x}_t$, W^i is the matrix of weights of the input gate for the previous hidden state $\boldsymbol{h}_{t-1}$, and $\boldsymbol{b}^i$ is the bias vector of the output gate. σ denotes the sigmoid activation function.

- The output gate. This filters the parts from the current cell state that are going to be used as current hidden state/output based on the values of the current input and previous hidden state.

$$\boldsymbol{o}_t = \sigma(U^o * \boldsymbol{x}_t + W^o * \boldsymbol{h}_{t-1} + \boldsymbol{b}^o) \tag{5.46}$$

  where U^o is the matrix of weights of the output gate for the current input $\boldsymbol{x}_t$, W^o is the matrix of weights of the output gate for the previous hidden state $\boldsymbol{h}_{t-1}$, and $\boldsymbol{b}^o$ is the bias vector of the output gate. σ denotes the sigmoid activation function.

At each time step, the forget gate uses Equation 5.44 to compute which information is no longer useful in the cell state, specifically by using the previous hidden state and the current input. At the same time, candidates for new information are computed using Equation 5.47, and their relevant parts are selected by means of the input gate, whose Equation is 5.45. At this point, the cell state is updated, as described in Equation 5.48. Once the cell state is updated, part of its information is filtered using the output gate (Equation 5.46). This information will be used as the cell's output and as its new hidden state. Equation 5.49 describes the process of computing the new hidden state/output.

$$\tilde{\boldsymbol{c}}_t = tanh(U^c * \boldsymbol{x}_t + W^c * \boldsymbol{h}_{t-1} + b^c) \tag{5.47}$$

where $\tilde{\boldsymbol{c}}_t$ is the candidate cell state, $tanh$ is the tanh activation function, U^c is the matrix of weights of the candidate cell state for the current input $\boldsymbol{x}_t$, W^c is the matrix of weights of the candidate cell state for the previous hidden state $\boldsymbol{h}_{t-1}$, and $\boldsymbol{b}^c$ is the bias vector of the candidate cell state.

$$\boldsymbol{c}_t = \boldsymbol{f}_t \times \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \times \tilde{\boldsymbol{c}}_t \tag{5.48}$$

where $\boldsymbol{c}_t$ is the current cell state, $\boldsymbol{f}_t$ is the output of the forget gate at the current time step, $\boldsymbol{c}_{t-1}$ is the previous cell state, $\boldsymbol{i}_t$ is the output of the input gate, and $\tilde{\boldsymbol{c}}_t$ is the candidate cell state computed using Equation 5.47.

$$\boldsymbol{h}_t = tanh(\boldsymbol{c}_t) \times \boldsymbol{o}_t \tag{5.49}$$

where $\boldsymbol{c}_t$ is the current step cell state, *tanh* is the tanh activation function, and $\boldsymbol{o}_t$ is the output from the output gate.

5.2.3.4 GRU

Cho et al. [2014] proposed GRU as a modification of LSTM. The objective was to reduce the complexity of the cells and increase the speed of the computations. In order to accomplish this, the cell state is merged into the hidden state. In addition, the forget gate and the input gate are merged into a new gate called the update gate. Lastly, a reset gate is used, the function of which is the same as that of the output gate of the LSTM cell. The schema of the GRU cell can be seen in Figure 5.12.

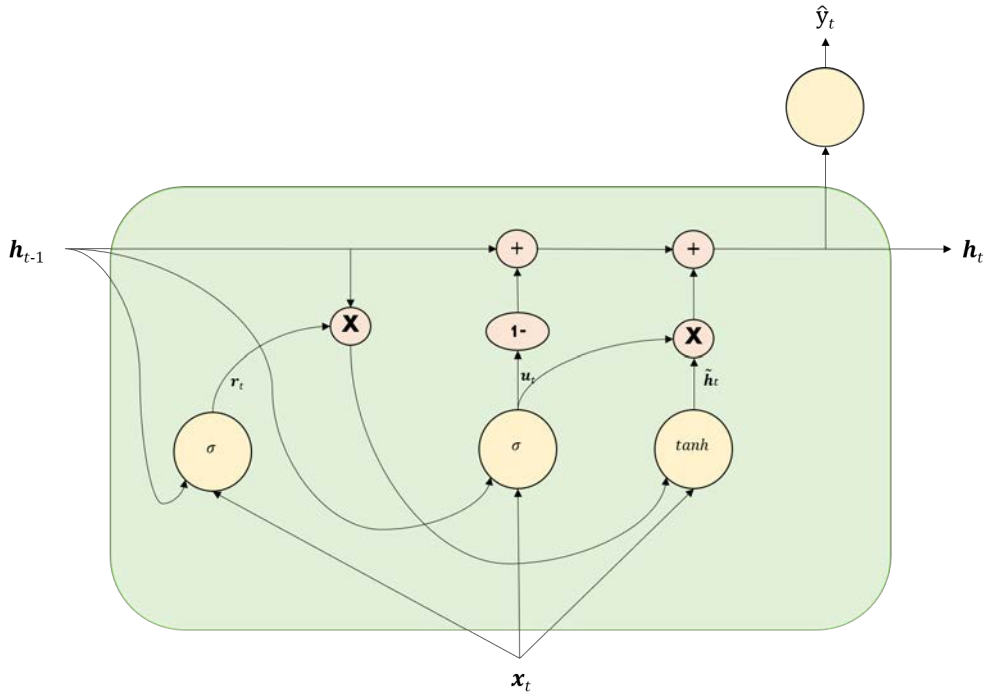


Figure 5.12: GRU cell.

The inner mechanism in a GRU cell starts with the computations at the reset gate (Equation 5.50), whose output will be used to filter the previous hidden state information in order to compute the new hidden state candidate. At the same time, the update gate is computed using the current input and the previous hidden state by means of Equation 5.51. The new candidate hidden state is computed by using the current input and an

element-wise multiplication between the reset gate output and the previous hidden state (this element-wise multiplication selects the elements of the previous hidden state and which values will be reset), as stated in Equation 5.49. Once the candidate hidden state for the current step is computed, the output of the update gate is used to decide which information to keep from the previous hidden state and which information to include in the new hidden state, as shown in Equation 5.53.

$$\boldsymbol{r}_t = \sigma(U^r * \boldsymbol{x}_t + W^r * \boldsymbol{h}_{t-1} + \boldsymbol{b}^r) \tag{5.50}$$

where $\boldsymbol{r}_t$ is the output of the reset gate, U^r is the matrix of weights of the reset gate for the current input $\boldsymbol{x}_t$, W^r is the matrix of weights of the reset gate for the previous hidden state $\boldsymbol{h}_{t-1}$, and $\boldsymbol{b}^r$ is the bias vector of the reset gate. σ stands for the sigmoid activation function.

$$\boldsymbol{u}_t = \sigma(U^u * \boldsymbol{x}_t + W^u * \boldsymbol{h}_{t-1} + \boldsymbol{b}^u) \tag{5.51}$$

where $\boldsymbol{u}_t$ is the output of the update gate, U^u is the matrix of weights of the update gate for the current input $\boldsymbol{x}_t$, W^u is the matrix of weights of the update gate for the previous hidden state $\boldsymbol{h}_{t-1}$, and $\boldsymbol{b}^u$ is the bias vector of the update gate. σ stands for the sigmoid activation function.

$$\tilde{\boldsymbol{h}}_t = tanh(U^h * \boldsymbol{x}_t + W^h * (\boldsymbol{r}_t \times \boldsymbol{h}_{t-1}) + \boldsymbol{b}^h \tag{5.52}$$

where $\tilde{\boldsymbol{h}}_t$ is the candidate hidden state, U^h is the matrix of weights of the candidate hidden state for the current input $\boldsymbol{x}_t$, W^h is the matrix of weights of the candidate hidden state for the previous hidden state $\boldsymbol{h}_{t-1}$, $\boldsymbol{r}_t$ is the output of the reset gate, and $\boldsymbol{b}^h$ is the bias vector of the update gate. σ stands for the sigmoid activation function.

$$\boldsymbol{h}_t = (1 - \boldsymbol{u}_t) \times \boldsymbol{h}_{t-1} + \boldsymbol{u}_t \times \tilde{\boldsymbol{h}}_t \tag{5.53}$$

6 Computational experiments – ARIMA

This chapter characterizes the parking sectors that were presented in Chapter 3. Specifically, the characterization presented was conducted using the ARIMA method, of which the theoretical issues have been discussed in Chapter 5. The objective of this modeling is to provide hourly forecasts of parking sector occupancy using their past occupancy values. The technique of using the same variable to provide forecast values is known in the time series literature as endogenous forecasting. In addition, the error is quantified at different forecasting horizons with a two-fold objective: first, in order to study how the forecasts decay as the forecasting horizon becomes longer; and second, to compare the ARIMA models with other models created by using other methodologies.

The chapter is organized in the following manner. First, it presents the design of experiments in order to provide structure and context about the modeling. Then, it introduces the goodness-of-fit metrics used to compare the candidate models and evaluate the errors in the chosen model. Next, it is explained the ARIMA modeling process. Following, it explains the step-by-step modeling process for each sector of each city. Lastly, the chapter closes with a summary and conclusions about the important facts and observations uncovered through the modeling.

6.1 Design of experiments

An experiment is considered to be the modeling process for each sector in each city. The outcome of an experiment is: the model that is proposed from a set of considered candidate models, in accordance with the methodology used; the error of its forecasts at different time horizons; and any relevant observation during the process. Models are compared using the goodness-of-fit metrics described in the following section. Table 6.1 provides a summary of the experiments considered and comments on how the idiosyncrasy of the data can have an effect on the modeling process, usually in the form of a question that would be desirable to take into account.

Questions that arise from the comments on the experiments are:

- How different are the models of the sectors based on their idiosyncrasies?
- Is it possible for an ARIMA model to handle multiple seasonalities?
- Do nearby sectors share the same model?

City	Sector	Experiment name	Comments
Antwerp	Pelikaanstraat	ARIMA_antwerp_pelikaanstraat	Sectors present different sizes, usage and locations.
	Van Wesenbekestraat	ARIMA_antwerp_vanwesenbekestraat	
	Van Wesenbekestraat Zone 2	ARIMA_antwerp_vanwesenbekestraat2	
	Van Schoonhovestraat Zone 1	ARIMA_antwerp_vanschoonhovestraat1	
	Van Schoonhovestraat Zone 2	ARIMA_antwerp_vanschoonhovestraat2	
	Van Schoonhovestraat Zone 3	ARIMA_antwerp_vanschoonhovestraat3	
Barcelona	Regular	ARIMA_barcelona_regular	At least two seasonalities: eight hours and daily.
Wattens	Inside	ARIMA_wattens_inside	Both sectors belong to the same zone and have the same usage.
	Outside	ARIMA_wattens_outside	
Los Angeles	Standard	ARIMA_LA_standard	2018 and 2019 have different occupancy values because of street works.
	Permit	ARIMA_LA_permit	
	Disabled	ARIMA_LA_disabled	
	ZipCar	ARIMA_LA_zipcar	
Riyadh	Sector-1	ARIMA_riyadh_sector1	All sectors are interconnected and present high cross-correlation values.
	Sector-10	ARIMA_riyadh_sector10	

Table 6.1: Table of experiments for ARIMA.

- Is it possible to overcome temporal changes?
- Is a model really necessary for each sector?
- How can sectors be characterized by their time series?

6.2 Goodness-of-fit metrics

When dealing with time series forecasting (regression) in this work, the metric used to compare the models and validate its results is the root mean square error (RMSE). The equation for this is presented in Equation 6.1. When using RMSE, the model with the lowest value is the best of the set. For ARIMA models, the RMSE is used to quantify the forecast errors when compared with the out-of-sample data, which is the data not used to estimate the model parameters.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{6.1}$$

where n is the number of samples, y is the ground truth and $\hat{y}$ is the forecast of the model for the sample i.

In order to compare different ARIMA models fitted to the same data, the most common metrics are the Akaike Information Criterion (AIC) (Equation 6.2) and the Bayesian Information Criterion (BIC) (Equation 6.3), as they indicate the relative quality of the models for a given set of data. Both metrics are similar, but BIC penalizes more of those models with a larger number of parameters. AIC models can be compared using the formula $exp(\frac{AIC_1 - AIC_2}{2})$, whose result is interpreted as how much more probable it is that the model with AIC_2 will minimize the information loss when compared with the

AIC_1 model. When using these criteria, the model with the lowest value is preferred; but if the difference between the values is very small, the simplest model is chosen in accordance with the principle of parsimony.

$$AIC = ln\hat{\sigma}_k^2 + \frac{n+2k}{n} \tag{6.2}$$

where $\hat{\sigma}_k^2$ stands as the value of the variance estimated by maximum likelihood, k is the number of parameters of the model and n the number of observations used to fit the model.

$$BIC = ln\hat{\sigma}_k^2 + \frac{k * ln(n)}{n} \tag{6.3}$$

where $\hat{\sigma}_k^2$ stands as the value of the variance estimated by maximum likelihood, k is the number of parameters of the model and n the number of observations used to fit the model.

Another interesting metric is the value of the likelihood function. The likelihood function is used to estimate the parameters of the ARIMA models through the maximum likelihood estimation (MLE) method. Equation 6.4 shows the calculation of this value.

$$\hat{l} = -\frac{n}{2}ln(2\pi) - \sum_{t=2}^{n}[\frac{1}{2}ln(\sigma_t^2) + \frac{(X_t - \mu_t)^2}{2\sigma_t^2}] - \frac{1}{2}ln(\sigma_z^2) - \frac{X_1^2}{2\sigma_z^2} \tag{6.4}$$

where $\hat{l}$ stands as the maximum loglikelihood value for an ARIMA model. The logarithm is used when dealing with probabilities, because it is easier to perform the operations and is less computationally costly due to the large number of decimals involved when using the ARIMA methodology

6.3 ARIMA modeling methodology

In order to develop the ARIMA models, the Box-Jenkins methodology has been used. This approach has well-differentiated stages that guide the researcher through creating an ARIMA model. A schema of the methodology stages is shown in Figure 6.1.

The first step in the Box-Jenkins methodology involves identifying the time series as stationary. One of the required assumptions of ARIMA models (and in fact, of any time series model) is the need for stationarity, which means that the mean and variance of our observations do not change as time goes on. In other words, their values are constant over time. Stationarity can be identified through visualization tools, such as box plots for observing the evolution of mean and variance over time or by using statistical tests that exploit the fact that a non-stationary process presents a unit root in its characteristic equation. One example of the latter is the Dickey-Fuller test. The cause of a non-stationary time series could lie in the mean or the variance. When the

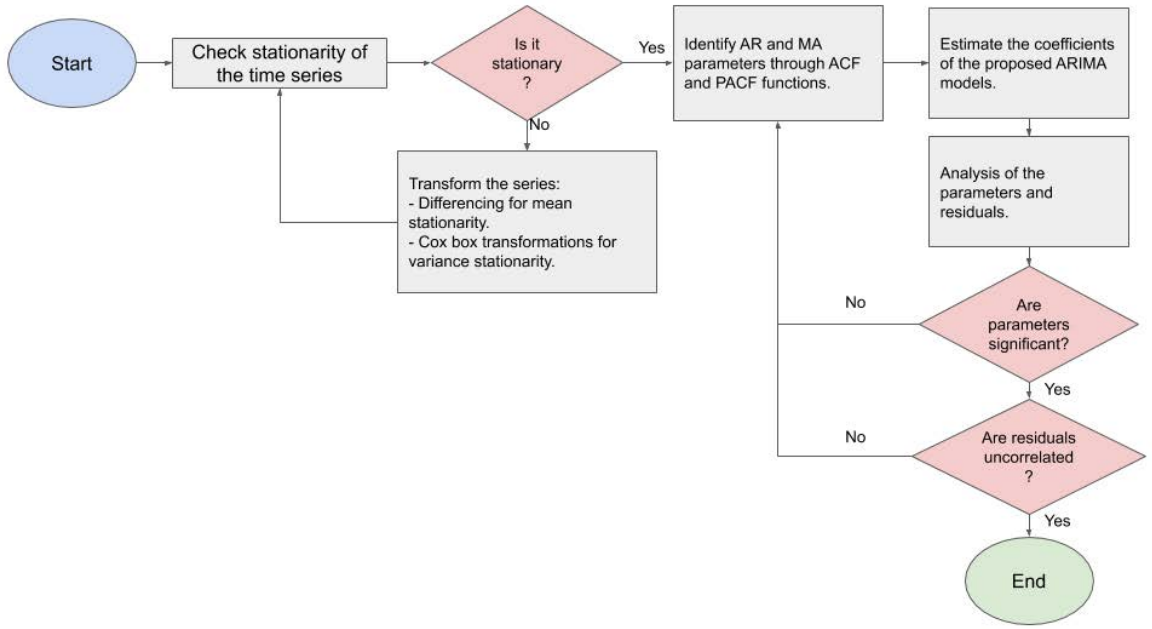


Figure 6.1: Box-Jenkins methodology schema.

mean is not constant over time, the time series values will increase or decrease as time goes on (trend), and this can be solved by using the difference operation. The difference operation is identified in an ARIMA model as the integrated order of the model, or $I(\cdot)$. If the variance is the cause of non-stationarity, the Cox-Box transformations Box and Cox [1982] can be used to transform the series to stationary.

Once the time series are stationary, the next step is to identify the autoregressive $AR(\cdot)$ and moving average $MA(\cdot)$ orders of the model. This step can be done through the graphic visualization of the ACF and PACF. The significant lags in the ACF indicates the order of the moving average part, while the significant lags in the PACF indicates the order of the autoregressive part. It is possible that the visualizations show a pattern of high correlations at equally spaced lags, thus indicating seasonality in the observations, and this is modeled by incorporating seasonal coefficients into the model. When seasonal coefficients are used, then we use the the Seasonal ARIMA (SARIMA) model, for which the parameters are identified as $SAR(\cdot)$, $SI(\cdot)$ and $SMA(\cdot)$. At this step, more than one model can be identified.

The next step is to fit the parameters and compute the AIC, BIC and parameter significance through a t-test. As mentioned in the previous section, AIC and BIC values are used to relatively compare different models that have been identified in the previous step. The t-test for parameter significance helps identify coefficients that are not needed in the model, as they do not have an effect on the amount of information captured by the model. One rule of thumb for identifying whether a parameter is significant is to divide the parameter value by its standard deviation; if the ratio is larger than two, then the parameter is significant.

The last step involves analyzing the residuals of the previously selected model in order to test their normality. If the model is able to capture the relevant information in the observations, the distribution of the residuals is a Gaussian distribution. The normality of the residuals can be studied through different plots such as: the Q-Q plot; the ACF and PACF residual plots of the residuals; the density plot; and the homoscedasticity plot over time. Another way of testing whether the residuals fulfill the normality condition is through statistical tests like the Ljung-Box Q test, in which the null hypothesis H_0 is that there are no correlations in the residuals while the alternative hypothesis is that the residuals are correlated and, thus, normality has not been established. With the information from the residual plots, it is possible to rethink the number of model parameters and try new combinations in order to normalize the residuals. A model without normality in its residuals does not imply that the model forecasts are useless, just that it will not be able to always give accurate forecasts and that, if it is used, one must provide explanations about the underlying process that generates the data.

As mentioned above, some of the Box-Jenkins stages can be approached through visualizations like checking the stationarity and identifying the model parameters. Part of this work has been presented previously in Chapter 3.

6.4 Antwerp ARIMA models

6.4.1 Pelikaanstraat sector

This section includes the experiment corresponding to the Pelikaanstraat sector of Antwerp, namely *ARIMA_ antwerp_pelikaanstraat*.

Daily box plots for the Antwerp sector of Pelikaanstraat (see Figure 6.2) present the behaviors of the mean and variance through the first two weeks of March 2019 after applying various differentiations to the data: non-differentiated; one regular; one seasonal (daily); and one regular plus one seasonal differentiation together. It can be observed that the mean has different levels for weekdays and weekends, meaning that perhaps the series are non-stationary. This behavior was already observed in the seasonal plots presented in Chapter 3.5.1. When applying one regular differentiation to the series, the mean and variance seem to become stationary. With one seasonal differentiation, the results are similar to the non-differentiated series. For the one regular plus one seasonal differentiation, the results are similar to the one regular differentiated series. The non-differentiated series must also be considered, since the long-term plots in Section 3.5.1 indicate that the mean is constant over the years.

One thing that must be taken into account when using differentiation in time series is that, for each order of differentiation, the mean and variance change and there exists the probability of over-differentiating the series, thus adding noise to the series. As a general rule of thumb, differentiation is performed until the ACF decays rapidly towards zero. Another rule is that the optimal order of differencing is the one with the lowest standard

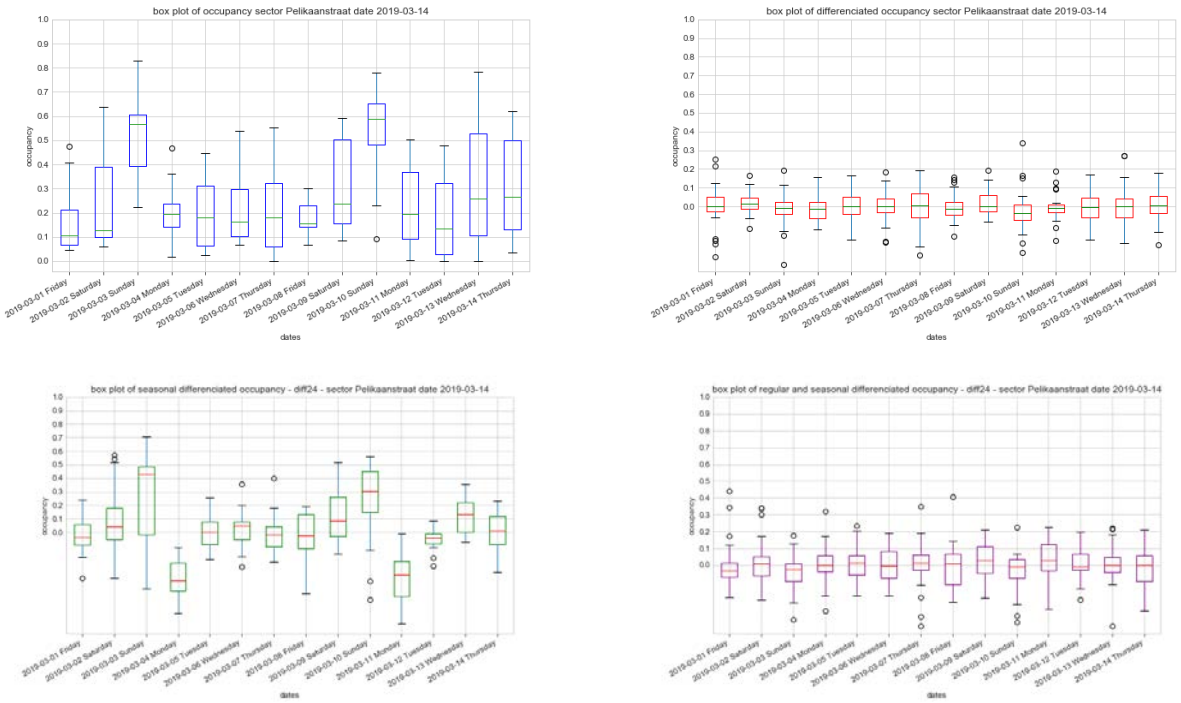


Figure 6.2: Box plots of the Antwerp Pelikaanstraat sector for the first two weeks of March 2019. Top left shows the non-differentiated series, top right the differentiated one, bottom left presents the seasonal differentiation (24 lags) and bottom right is the series with one regular differentiation and one seasonal differentiation together.

Difference order	Mean	Variance
No differentiation	0.3570	0.0524
One regular	0	0.0140
Two regular	0	0.0288
One seasonal	-0.0008	0.0507
Two seasonal	0.0002	0.1492
Regular plus seasonal	0	0.0236
Two regular, one seasonal	0	0.0538
One regular, two seasonal	0	0.0701
Two regular, two seasonal	0	0.0706

Table 6.2: Pelikaanstraat sector mean and variance when using different orders of differentiation.

Model	Log likelihood	AIC	BIC
$ARIMA(1,0,0)$	7949.008	-15892.016	-15870.215
$SARIMA(0,1,2)(0,1,1)_{24}$	8304.040	-16600.081	-16571.022
$SARIMA(0,1,0)(0,1,1)_{24}$	8121.553	-16239.106	-16224.577

Table 6.3: Comparison of the Antwerp Pelikaanstraat sector ARIMA model scores.

deviation or variance. Table 6.2 shows the mean and variance of the Pelikaanstraat sector under different orders of differentiation. The Pelikaanstraat sector shows the minimum variance when using one regular differentiation and, in the case of seasonal differentiation, also when using one regular plus one seasonal.

The order of the coefficients of an ARIMA model can be determined by the visualizations of the ACF and PACF functions for the whole series. Figure 6.3 shows the Pelikaanstraat ACF and PACF of the data with different transformations. From the plots, the following models can be considered: AR(1) and constant mean with the non-differentiated series; AR(2) and/or MA(2); and SAR(1) and/or SMA(1) for the one regular plus one seasonal differentiated series. Because it is not clear at all, the MA(2),SMA(1) is the one considered because it seems to have a sharp cut-off in the autocorrelation plot. Lastly, we consider a third model with the form of SMA(1) for the one regular plus one seasonal differentiated series. Modeling the data this way means that the observations are realizations based on the occupancy of the day before at the same hour, plus a random value coming from white noise.

Notice that some of the series present higher autocorrelation values at the lags that coincide with a one-week period, that is, 168 hours.

Table 6.3 shows a summary of the results of the fitted models for the Pelikaanstraat sector. From the models considered, $SARIMA(0,1,2)(0,1,1)_{24}$ is the one with the best AIC and BIC scores (the lower, the better) and higher likelihood.

All the parameters are significant, with their significance being obtained by computing

Figure 6.3: Pelikaanstraat ACF and PACF plots for the non-differentiated, one differentiated, seasonal differentiated, and one regular plus one seasonal differentiated series.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
0.1033	0.1360	0.1562	0.1737	0.1881	0.1994

Table 6.4: Antwerp Pelikaanstraat ARIMA RMSE.

the t-value of each parameter (ratio of the parameter by its standard error). This is then used to obtain the p-value for testing the null hypothesis $H_0 : \beta_i = 0$ for $i = 1, ..., n$ where n is the number of observations and β is the parameter value. The mathematical description of the model is presented in equation 6.5. Notice that, despite the simplicity of $ARIMA(1, 0, 0)$, the model scores are not far from the proposed one.

$$\hat{x_t} = -0.1587_{(0.008)} w_{t-1} - 0.1341_{(0.009)} w_{t-2} - 0.9509_{(0.003)} w_{t-24} + w_t, \tag{6.5}$$

with $\sigma^2 = 0.0121$. The lower values in parenthesis denote the corresponding estimation of the standard errors for each parameter. w_{t-n} represents the white noise values for a number of n lags considered.

The residuals are analyzed by means of the visualizations presented in Figure 6.4. The visualizations show that not all information is captured by the data, as there are correlations in the residuals. The density plot of the residuals does not follow a normal distribution, a fact that is also confirmed by the visualizations of the Q-Q plot and the Ljung-Box p-values plot. The ACF and PACF plots show that there is correlated information in the residuals between the first 24 lags. The conclusions are that the forecasts of the proposed model need to be viewed with caution, as not all the assumptions have been fulfilled.

The parameters of the model have been estimated using data from February 2019 until January 2019. February, March and April data is used for testing the model forecasting for out-of-sample data. The model forecast has been evaluated with different forecasting horizons, as one of the research questions of this work is to study how the accuracy of the models decays as the forecasting horizon becomes longer. Specifically, Table 6.4 presents the RMSE of the model at each of the considered forecasting hours, from one to six. In the case of the ARIMA models, it is necessary to use a multi-step strategy in order to obtain forecasts for consecutive time steps, that is, the forecast of the first time step (next hour) is used as input for the model in order to obtain the forecast for the next time step (forecast for two hours from the current step). This has an effect on the accuracy, as the uncertainty increases at each step.

Figure 6.5 shows the one-step forecasts for the ARIMA model proposed for the Pelikaanstraat sector of Antwerp. The top one presents the forecasts for the entire time interval of the test data. The bottom shows 48 hours of forecasts. Notice that the forecasts sometimes seem to fall behind the real observations.

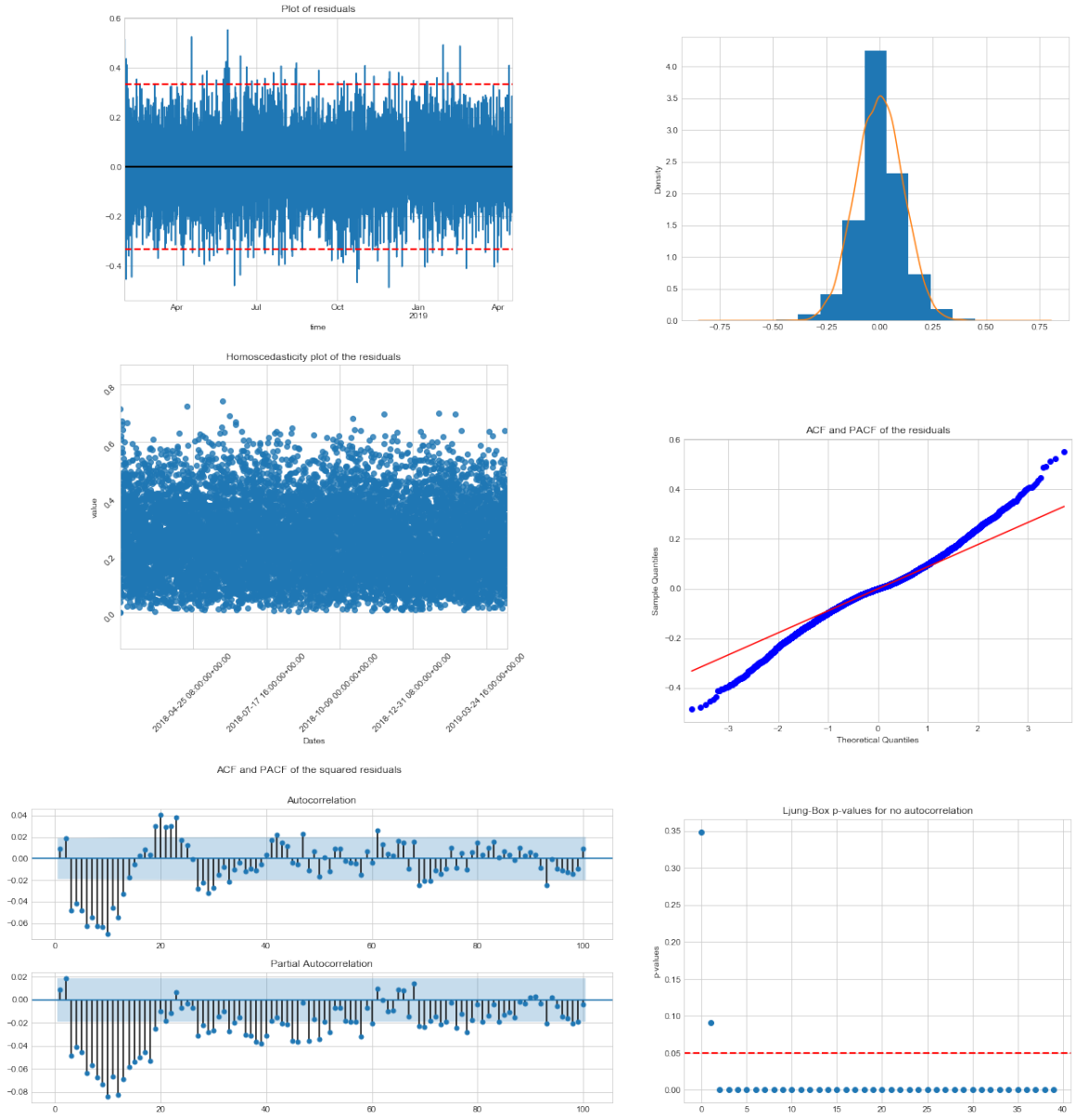


Figure 6.4: Antwerp Pelikaanstraat residual analysis of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

Figure 6.5: Antwerp Pelikaanstraat ARIMA forecasts. Blue denotes the observed occupancy, red the occupancy forecasted by the model, and green is the confidence interval.

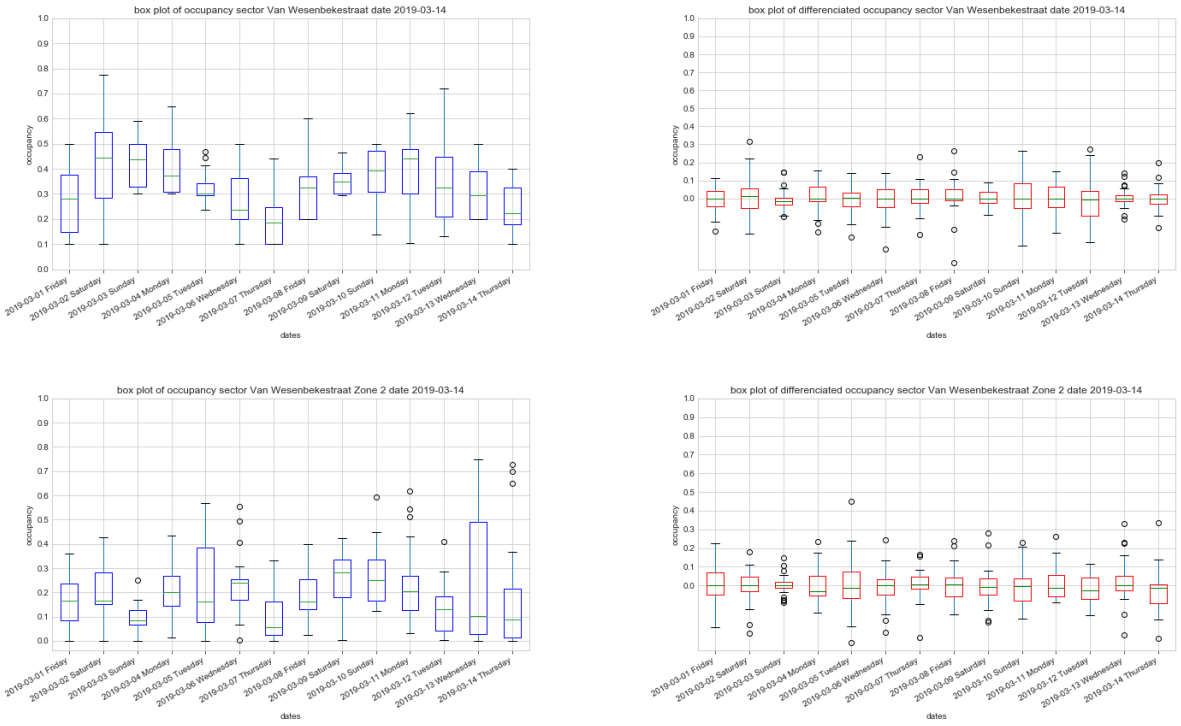


Figure 6.6: Daily box plots of the first two weeks of March for the sectors Van Wesenbekestraat Zone 1 (top) and Van Wesenbekestraat Zone 2 (bottom).

6.4.2 Van Wesenbekestraat sectors

This section includes the experiments corresponding to the two sectors of Antwerp located on Van Wesenbekestraat Street. The corresponding experiments are: *ARIMA_ antwerp_vanwesenbekestraat* and *ARIMA_ antwerp_vanwesenbekestraat2*.

The Van Wesenbekestraat sectors daily box plots are presented in Figure 6.6. Zone 1 presents a cyclical weekly behavior through the first two weeks of March 2019. Occupancy levels hit bottom on Thursdays, and the maximum values are on Saturdays and Sundays. After a regular differentiation, the series presents a constant mean of value zero with a more or less constant variance for each of the days. Zone 2 presents a similar pattern to the one from Zone 1, especially in the second week of March. The data with one regular differentiation presents patterns similar to Zone 1.

Figure 6.7 shows the ACF and PACF of the series for Van Wesenbekestraat Zone 1 and Zone 2 when non-differentiated and after a one regular plus one seasonal (daily) differentiation. Both sectors present very similar plots and, because of this, the candidate models are the same for each of them. Possible models are: AR(1) for the non-differentiated series, with a constant trend (mean); and AR(2), MA(2) and SMA(1) for the one regular plus one seasonal differentiated series. As the plots are similar to those of the Pelikaanstraat sector, no further differentiation order is considered.

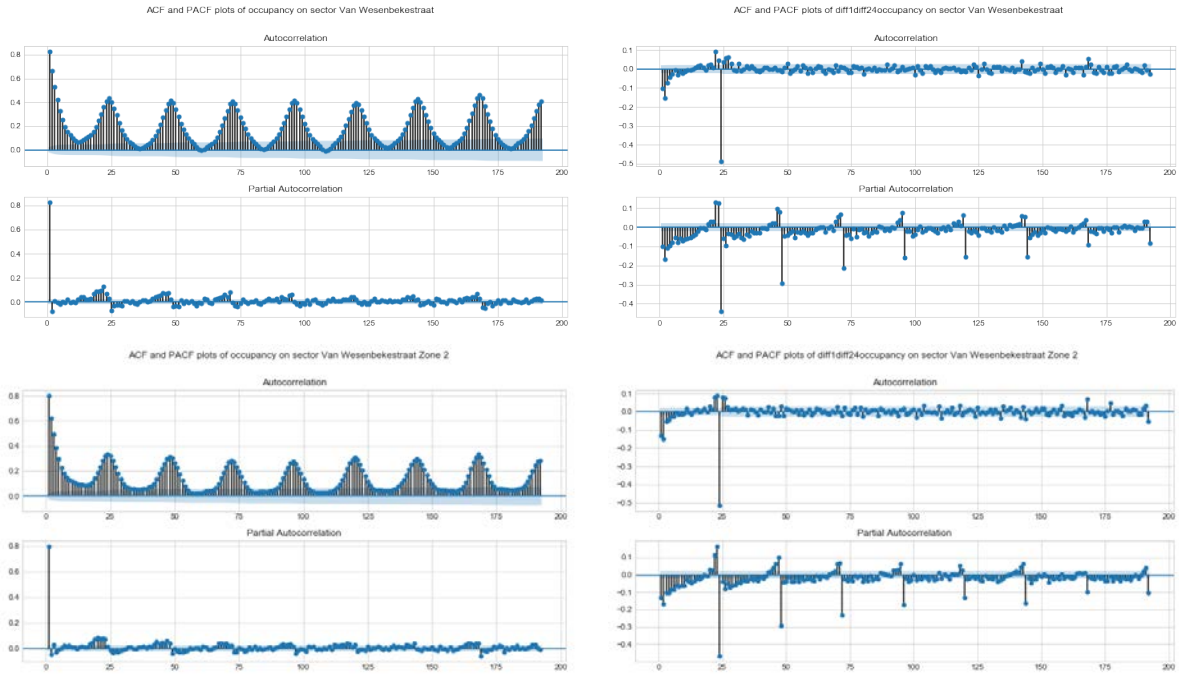


Figure 6.7: Van Wesenbekestraat Zone 1 (top) and Zone 2 (bottom) ACF and PACF plots for the non-differentiated series (left) and the regular plus seasonal differentiated series (right).

Table 6.5 presents the values of the mean and variance under different orders of differentiation for the regular plus seasonal parts. The series achieve the lowest mean and variance with one regular differentiation and with one regular plus one seasonal differentiation.

Table 6.6 shows a summary of the results of the fitted models for the sectors in Van Wesenbekestraat Zones 1 and 2. From the models considered, the $SARIMA(2,1,2)(0,1,1)_{24}$ is the one with better AIC and BIC scores (the lower, the better) and higher likelihood. Notice that the model $ARIMA(1,0,0)$ with a constant trend also has a good score.

From Equation 6.6, we can see that some of the parameters are not significant. Specifically, all parameters of the regular part are non-significant, as their standard deviations are higher than their values and, thus, the null hypothesis $H_0 : \beta_i = 0$ for $i = 1, ..., n$ cannot be rejected. Something similar happens with the candidate model for the sector in Van Wesenbekestraat Zone 2, as can be seen in Equation 6.7. The second parameter of the MA(2) part is non-significant.

$$\begin{aligned}\hat{x_t} = &-0.1560_{(2.492)}x_{t-1} + 0.6698_{(1.857)}x_{t-2} - 0.0967_{(2.489)}w_{t-1} \\ &- 0.8949_{(2.482)}w_{t-2} - 0.9655_{(0.003)}w_{t-24} + w_t,\end{aligned} \tag{6.6}$$

with $\sigma^2 = 0.007$. The lower values in parenthesis denote the corresponding estimate of the standard errors for each parameter. w_{t-n} represents the white noise and x_{t-n} the

Difference order	Van Wes. Zone 1 - Mean	Van Wes. Zone 1 - Variance	Van Wes. Zone 2 - Mean	Van Wes. Zone 2 - Variance
No differentiation	0.4333	0.0263	0.2822	0.0212
One regular	0	0.0089	0	0.0085
Two regular	0	0.0181	0	0.018
One seasonal	0.0002	0.0296	0.001	0.0282
Two seasonal	-0.0004	0.0878	-0.0003	0.0838
Regular plus seasonal	0	0.0151	0	0.0157
Two regular, one seasonal	0	0.0333	0	0.0356
One regular, two seasonal	0	0.0451	-0.0001	0.0476
Two regular, two seasonal	0	0.0988	0	0.1089

Table 6.5: Table of mean and variance values for the sectors on Van Wesenbekestraat Street under different orders of differentiation for the regular and seasonal parts.

Sector	Model	Log likelihood	AIC	BIC
Van Wesenbekestraat Zone 1	$ARIMA(1,0,0)$	9045.552	-18085.103	-18063.725
Van Wesenbekestraat Zone 1	$SARIMA(2,1,2)(0,1,1)_{24}$	9714.117	-19416.235	-19373.494
Van Wesenbekestraat Zone 2	$ARIMA(1,0,0)$	9351.470	-18696.941	-18675.562
Van Wesenbekestraat Zone 2	$SARIMA(2,1,2)(0,1,1)_{24}$	9740.899	-19469.798	-19427.057

Table 6.6: Model scores for the Antwerp sectors in Van Wesenbekestraat Zone 1 and Van Wesenbekestraat Zone 2.

past realizations of the series. n indicates the number of lags for each parameter.

$$\begin{aligned}\hat{x_t} = &-0.1625_{(0.054)}x_{t-1} + 0.6465_{(0.042)}x_{t-2} - 0.0861_{(0.051)}w_{t-1} \\ &- 0.9001_{(0.050)}w_{t-2} - 0.9739_{(0.003)}w_{t-24} + w_t,\end{aligned} \tag{6.7}$$

with $\sigma^2 = 0.0069$

The mathematical expression for the model $ARIMA(1,0,0)$ with trend is presented in Equations 6.8 and 6.9 for Zones 1 and 2, respectively. Both estimated parameters (mean and one autoregressive coefficient) are significant.

$$\hat{x_t} = 0.8297_{(0.006)}x_{t-1} + 0.0737_{(0.003)} + w_t, \tag{6.8}$$

with $\sigma^2 = 0.0082$.

$$\hat{x_t} = 0.8_{(0.007)}x_{t-1} + 0.0565_{(0.002)} + w_t, \tag{6.9}$$

with $\sigma^2 = 0.0077$

Considering the results obtained from both models, we propose a third one that tries to combine the best parts of the previous models. The third model is a $SARIMA(1,0,0)(0,0,1)_{24}$ with no trend. The equation of the third model is presented

Sector	Model	Log likelihood	AIC	BIC
Van Wesenbekestraat Zone 1	$SARIMA(1,0,0)(0,1,1)_{24}$	9705.391	-19404.783	-19383.412
Van Wesenbekestraat Zone 2	$SARIMA(1,0,0)(0,1,1)_{24}$	9707.417	-19408.833	-19387.463

Table 6.7: Scores of the proposed third model for the Antwerp sectors of Van Wesenbekestraat Zone 1 and Van Wesenbekestraat Zone 2.

in 6.10 for Zone 1 and in 6.11 for Zone 2. Both models present very similar values for their parameters, and all of them are significant. With the obtained results, the same model can be proposed for both sectors.

$$\hat{x_t} = 0.7530_{(0.007)}x_{t-1} - 0.9578_{(0.003)}w_{t-1} + w_t, \tag{6.10}$$

with $\sigma^2 = 0.007$.

$$\hat{x_t} = 0.7530_{(0.007)}x_{t-1} - 0.9604_{(0.003)}w_{t-1} + w_t, \tag{6.11}$$

with $\sigma^2 = 0.007$.

Table 6.7 presents the scores of the proposed model, which are very close to the model $SARIMA(2,1,2)(0,1,1)_{24}$.

Figure 6.8 presents the residual analysis of the model $SARIMA(1,0,0)(0,1,1)_{24}$, which is chosen for the sector in Van Wesenbekestraat Zone 1. The residual analysis shows that the model fails to capture some of the patterns from the first lags. While the density plot of the residuals seems to follow a normal distribution, the Q-Q plot shows that in fact the distribution has "heavy tails", meaning that extreme values are not captured. The Q-statistic confirms that the residuals do not follow a normal distribution.

Figure 6.9 presents the residual analysis of the model $SARIMA(1,0,0)(0,1,1)_{24}$, which is chosen for the sector in Van Wesenbekestraat Zone 2. Similarly to Zone 1, the residual analysis shows that the model is not able to capture all the patterns that exist in the data. From the ACF and PACF visualizations, we can see that correlations remain in the first twenty-four lags. The Q-Q statistic confirms that the residuals do not follow a normal distribution.

The models have been fitted using data from December 2018 until the end of October 2019. November and December 2019 data have been used to evaluate the model with out-of-sample data. Table 6.8 presents the RMSE of the model at different steps up to six, using a multi-step approach. It can be seen that the accuracy of the model decays as the forecasting steps increase.

Figure 6.10 presents the out-of-sample predictions with one-step forecasts vs. Figure 6.11, which presents the real occupancy. For Van Wesenbekestraat Zone 1, the model provides forecasts that are more conservative, as their values fall short when forecasting

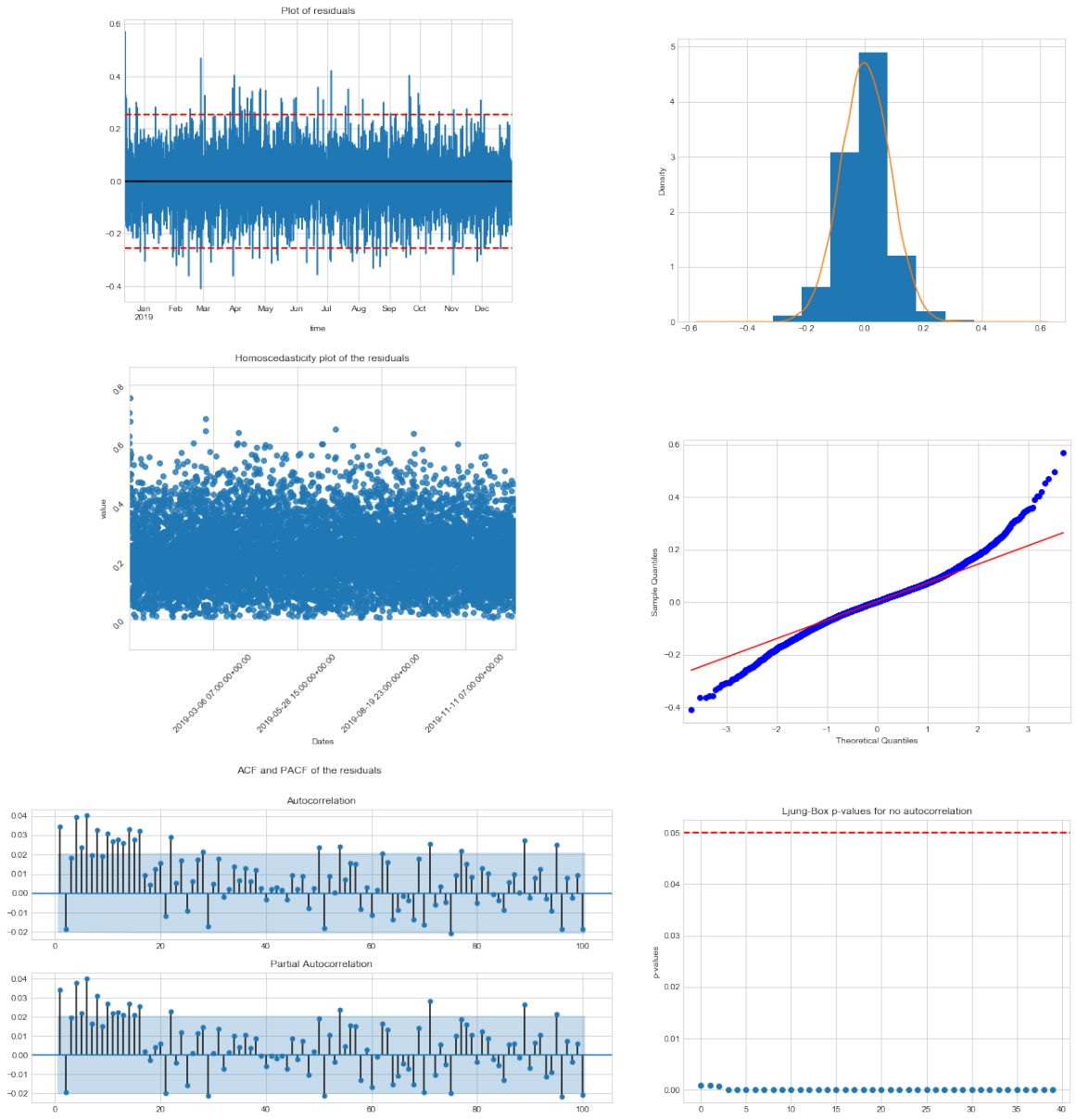


Figure 6.8: Antwerp Van Wesenbekestraat Zone 1 residual analysis of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots, and Q-statistic p-values plot.

Sector	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
Van Wes. Zone 1	0.0778	0.0997	0.1073	0.1110	0.1130	0.1146
Van Wes. Zone 2	0.0824	0.1039	0.1130	0.1190	0.1221	0.1233

Table 6.8: Antwerp Van Wesenbekestraat Zone 1 and Zone 2 ARIMA RMSE at different forecasting horizons.

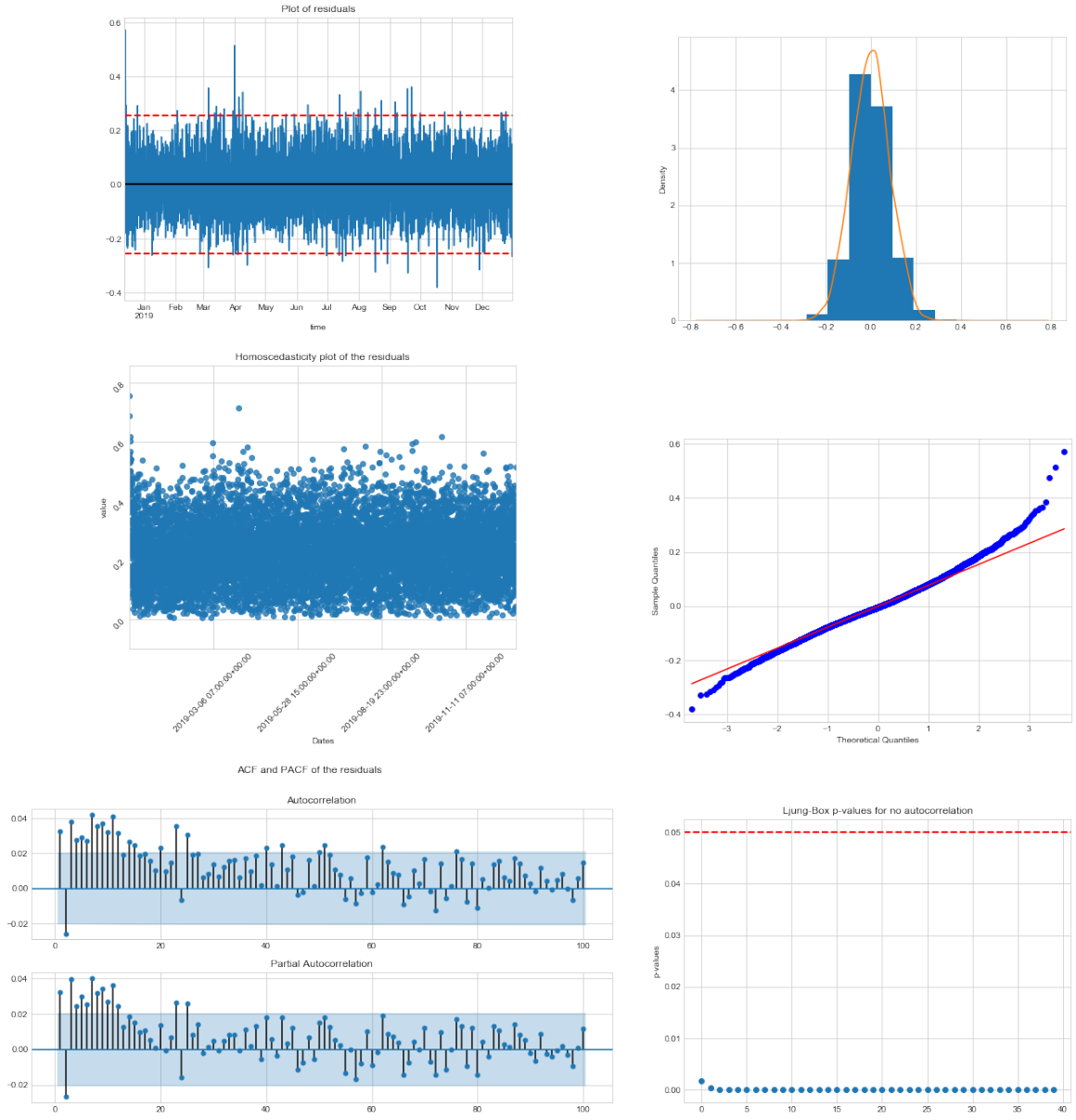


Figure 6.9: Antwerp Van Wesenbekestraat Zone 2 residual analysis of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots, and Q-statistic p-values plot.

for low- or high-occupancy levels. For Van Wesenbekestraat Zone 2, the model seems to be repeating the most recent input.

6.4.3 Van Schoonhovestraat sectors

This section includes the experiments corresponding to the three sectors of Antwerp located on Van Schoonhovestraat Street. The corresponding experiments are: *ARIMA_ antwerp_vanschoonhovestraat1*, *ARIMA_ antwerp_vanschoonhovestraat2* and *ARIMA_ antwerp_vanschoonhovestraat*3.

Figure 6.12 presents the daily box plots of the sectors on Van Schoonhovestraat Street for the first two weeks of March 2019. Remember from Chapter 3.5.1 that the Van Schoonhovestraat sectors of Zones 1 and 3 behave very similarly and have a similar number of parking places, while the sector in Zone 2 behaves erratically because of the low number of parking places and their use. The levels of parking occupancy for Van Schoonhovestraat Zones 1 and 3 are higher for weekends than for weekdays, similarly to what has been observed in previous parking sectors; while for the series with one regular differentiation, the mean becomes zero and the variance is reduced. For Zone 2, it is difficult to identify any pattern because of the limited number of places, which is three in total. The differentiated series pushes the mean towards zero.

In order to estimate the number of parameters of a SARIMA model, we use visualizations of the ACF and PACF functions. Figure 6.13 shows the ACF and PACF of the data for the Van Schoonhovestraat sectors with no transformation and with one regular differentiation plus one seasonal (daily) differentiation. Similarly to the previous parking sectors, the non-differentiated series presents an AR(1) model with trend. It is possible to also consider the AR(2), MA(2) and SMA(1) models for the one regular differentiation plus one seasonal differentiation model. Notice that the sectors of Zones 1 and 3 present higher autocorrelation values at the lags that coincide with a one-week period, that is, 168 hours. Lastly, the model $SARIMA(1,0,0)(0,1,1)_{24}$ is considered to be a merge of the two models proposed and is similar to what was done with the Van Wesenbekestraat models. The Van Schoonhovestraat Zone 2 sector presents the typical random walk signature that consists of a long tail for the ACF and an abrupt cut-off at the first lag of the PACF. The mathematical form of a random walk is $x_t = x_{t-1} + w_t$, where x_t is the actual value of the time series, x_{t-1} is the previous value and w_t is white noise, i.e., the stochastic component of the series. Such a process represents an $ARIMA(0,1,0)$ model. A random walk model is a special case of the $ARIMA(1,0,0)$, in which the parameter has value 1.

Table 6.9 presents the mean and variance values for the time series of the Van Schoonhovestraat sectors under different orders of differentiation for the one regular plus one seasonal part. Similarly to those sectors on Van Wesenbekestraat, the minimum values of the time series' mean and variance are achieved with the one regular differentiation and the one regular plus one seasonal differentiation for all three sectors.

Figure 6.10: Antwerp Van Wesenbekestraat Zone 1 ARIMA forecasts. Blue denotes the observed occupancy, red the occupancy forecasted by the model and green is the confidence interval. Top presents the entire out-of-sample interval of data while bottom presents out-of-sample predictions for two days.

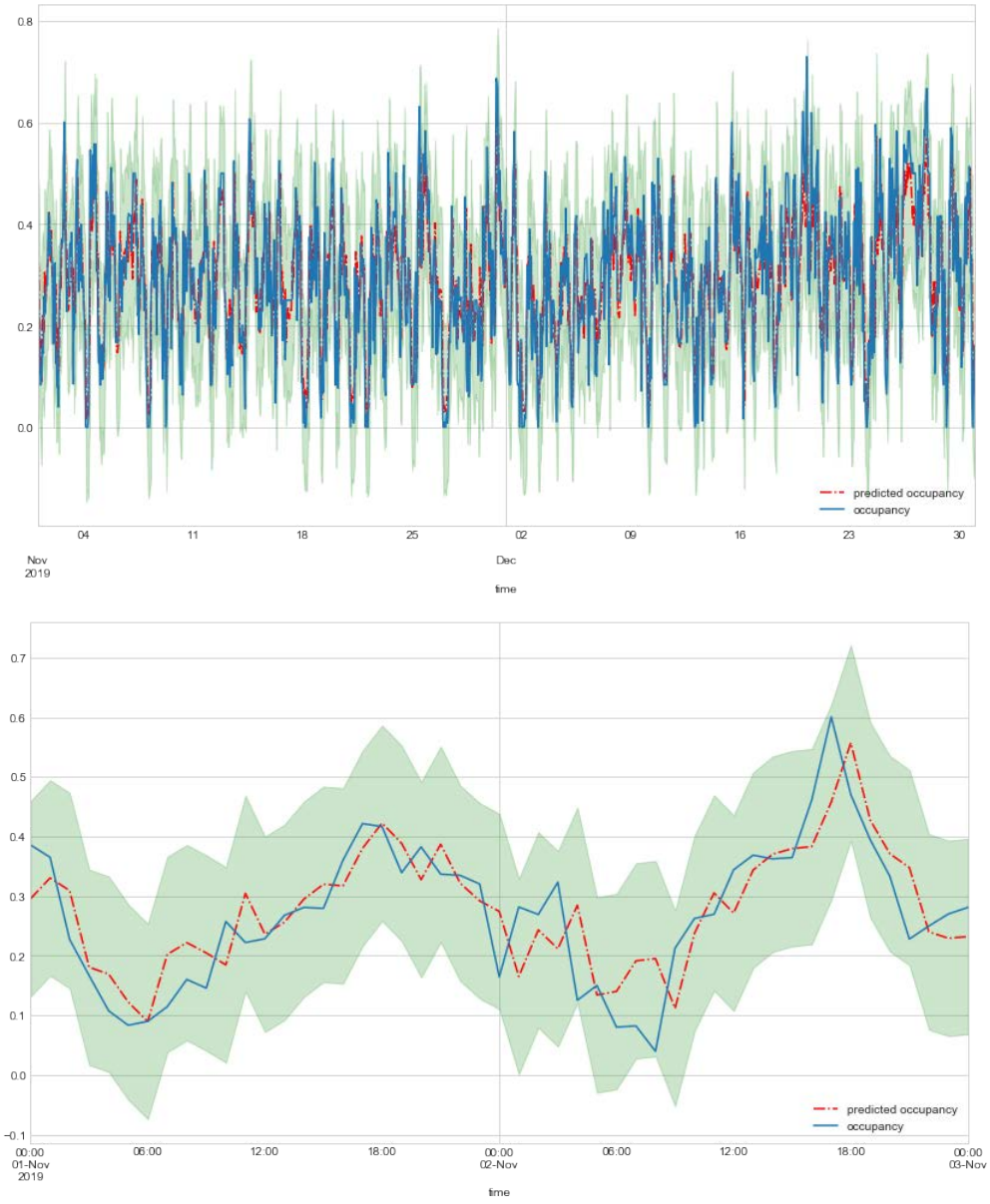


Figure 6.11: Antwerp Van Wesenbekestraat Zone 2 ARIMA forecasts. Blue denotes the observed occupancy, red the occupancy forecasted by the model and green is the confidence interval. Top presents the entire out-of-sample interval of data while bottom presents out-of-sample predictions for two days.

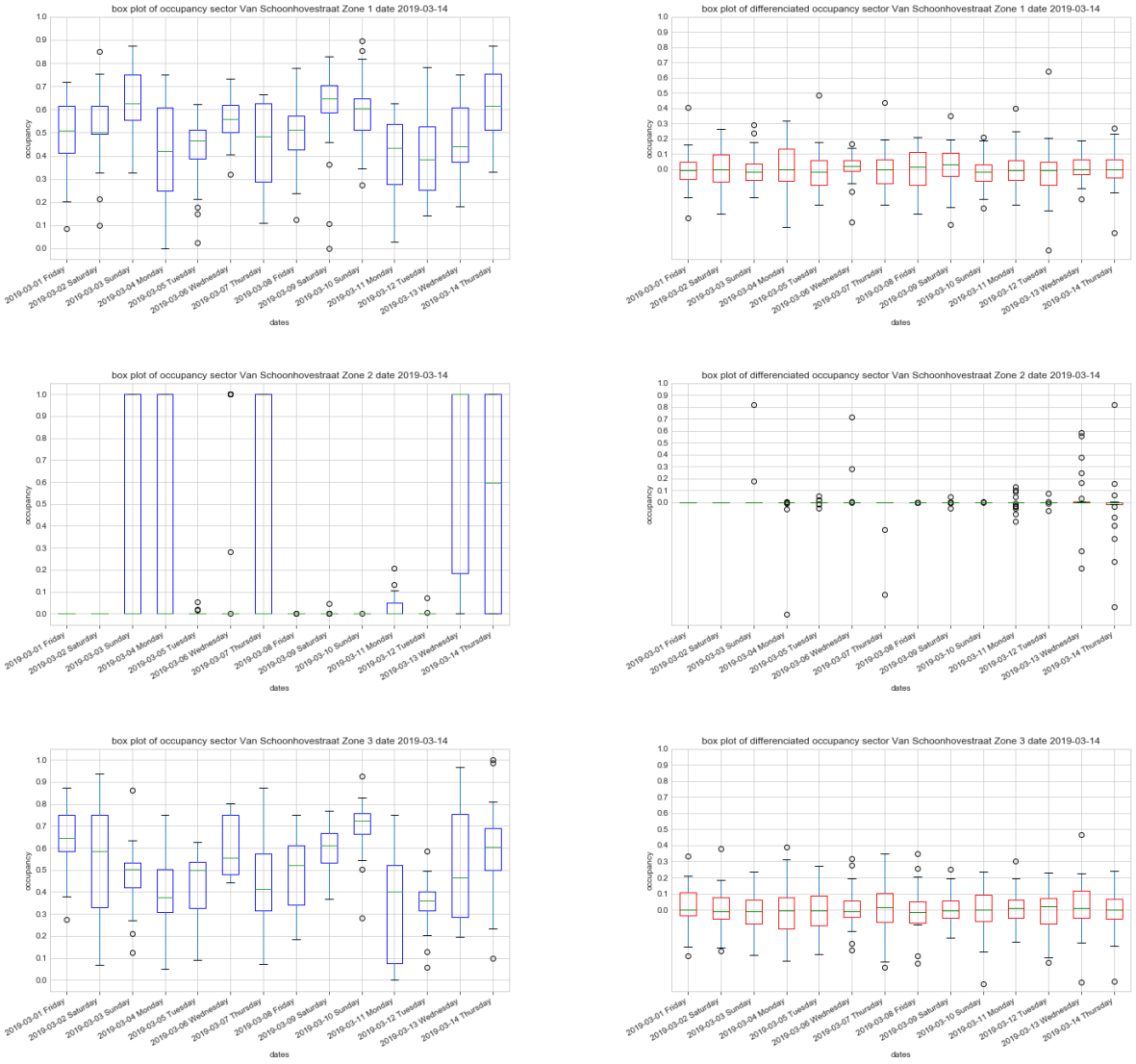


Figure 6.12: Box plots of the Antwerp Van Schoonhovestraat sectors. First row for Zone 1, second row for Zone 2 and the last row for Zone 3. For each sector, the first column shows the non-differentiated time series and the second column shows the time series with one regular differentiation.

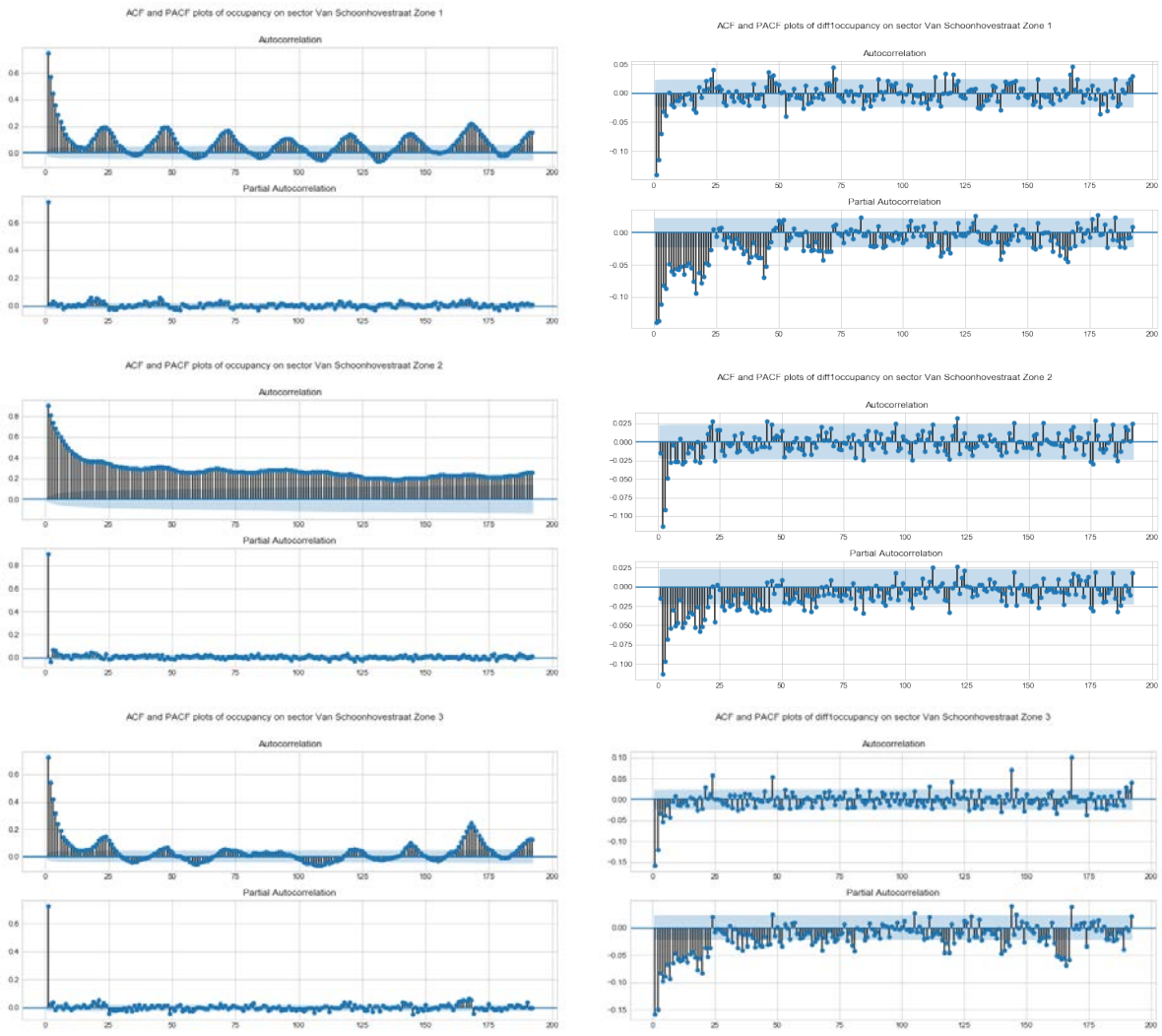


Figure 6.13: Van Schoonhovestraat ACF and PACF plot for the non-differentiated series and the one regular plus one seasonal differentiated series. From top to bottom, the plots represent Zones 1, 2 and 3.

Difference order	Van Sc. Z1 - Mean	Van Sc. Z1 - Variance	Van Sc. Z2 - Mean	Van Sc. Z2 - Variance	Van Sc. Z3 - Mean	Van Sh. Z3 - Variance
No differentiation	0.5183	0.0266	0.2763	0.0717	0.5337	0.0368
One regular	0	0.0133	0.0001	0.0139	0	0.0201
Two regular	0	0.0303	0	0.0282	0	0.0465
One seasonal	-0.0003	0.0428	0.0013	0.0932	0.0002	0.0627
Two seasonal	-0.0007	0.1281	0.0003	0.2737	0	0.1825
Regular plus seasonal	-0.0001	0.0255	0	0.0273	0	0.0378
Two regular, one seasonal	0	0.0594	0	0.055	0	0.0888
One regular, two seasonal	0	0.0763	0	0.0816	0.0002	0.1132
Two regular, two seasonal	0	0.1771	0	0.1638	0	0.2668

Table 6.9: Table of mean and variance values for the sectors on Van Schoonhovestraat Street under different orders of differentiation for the regular and seasonal parts.

Sector	Model	Log likelihood	AIC	BIC
Van Schoonhovestraat Zone 1	$ARIMA(1,0,0)$	5990.677	-11975.354	-11954.619
Van Schoonhovestraat Zone 1	$SARIMA(2,1,2)(0,1,1)_{24}$	6051.533	-12091.067	-12049.617
Van Schoonhovestraat Zone 1	$SARIMA(1,0,0)(0,1,1)_{24}$	6045.303	-12084.605	-12063.880
Van Schoonhovestraat Zone 2	$ARIMA(1,0,0)$	5519.204	-11032.408	-11011.673
Van Schoonhovestraat Zone 2	$SARIMA(2,1,2)(0,1,1)_{24}$	5519.894	-11027.789	-10986.339
Van Schoonhovestraat Zone 2	$SARIMA(1,0,0)(0,1,1)_{24}$	5485.387	-10964.773	-10944.048
Van Schoonhovestraat Zone 2	$ARIMA(0,1,0)$	4525	-9049	-9042
Van Schoonhovestraat Zone 3	$ARIMA(1,0,0)$	4511.671	-9017.343	-8996.608
Van Schoonhovestraat Zone 3	$SARIMA(2,1,2)(0,1,1)_{24}$	4540.294	-9068.587	-9027.138
Van Schoonhovestraat Zone 3	$SARIMA(1,0,0)(0,1,1)_{24}$	4544.193	-9082.387	-9061.662

Table 6.10: Score comparison of the ARIMA models for the Antwerp Van Schoonhovestraat Street sectors.

Table 6.10 shows a summary of the results of the fitted models for the sectors on Van Schoonhovestraat Street. All the models considered have similar scores. Because of this and, in following the principle of parsimony, the model with less complexity is the one proposed, which in this case is $ARIMA(1,0,0)$.

The equations of the proposed models are 6.12, 6.13 and 6.14. The Van Schoonhovestraat Zones 1 and 3 models are very similar to the Van Wesenbekestraat models, which perhaps reflects the high cross-correlation between the time series of both models, as shown in Chapter 3.5.1. This indicates that it is possible to use a single model for both sectors. Notice that the random walk model scored close to the other models considered for the sector in Zone 2.

$$\hat{x_t} = 0.75_{(0.007)}x_{t-1} + 0.1295_{(0.004)} + w_t \tag{6.12}$$

with $\sigma^2 = 0.0116$. The lower values in parenthesis denote the corresponding estimates of the standard errors of each parameter. x_{t-n} represents the previous lags values for a number of n lags considered.

$$\hat{x_t} = 0.9032_{(0.007)}x_{t-1} + 0.0268_{(0.003)} + w_t \tag{6.13}$$

with $\sigma^2 = 0.0132$

$$\hat{x_t} = 0.7269_{(0.008)}x_{t-1} + 0.1457_{(0.004)} + w_t \tag{6.14}$$

with $\sigma^2 = 0.0173$

Sector	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
Van Sch. Zone 1	0.1042	0.1259	0.1367	0.1423	0.1459	0.1489
Van Sch. Zone 2	0.1310	0.1727	0.1975	0.2109	0.2207	0.2271
Van Sch. Zone 3	0.1307	0.1540	0.1642	0.1710	0.1731	0.1738

Table 6.11: Antwerp Van Schoonhovestraat Zone 1, Zone 2 and Zone 3 ARIMA RMSE at different forecasting horizons.

The residual analysis of the sector in Van Schoonhovestraat Zone 1 is presented in Figure 6.14. The residuals clearly do not follow a normal distribution, as can be seen in the Q-Q plot. Some significant patterns are also not captured, as seen in the ACF and PACF residual plots. The Q-statistic confirms the lack of normality for residuals at different lags.

Figure 6.15 presents the residual analysis of the sector in Van Schoonhovestraat Zone 2. The sector's erratic behavior and lack of patterns are once more noticeable in the residual plots, where we can see that the residuals have high levels of correlations and do not follow a normal distribution.

From the visualizations in Figure 6.16, it is possible to see that the model proposed for the sector in Van Schoonhovestraat Zone 3 does a better job than the previously considered modelsbecause the residuals are close to following a normal distribution, as stated in the density plot, the homoscedasticity plot and the Q-Q plot. However, the ACF and PACF of the residuals show that some seasonal correlations are not captured well. The Ljung–Box test statistic shows that normality in the residuals is accomplished by a larger number of lags when compared to previous Antwerp models.

The data interval for estimating the parameters of the models is from February 2019 to October 2019; while data from November until the end of December was used for testing the models' goodness of fit to unseen data. 6.11 presents the RMSE of the models for each sector at different steps up to six, using a multi-step approach. It can be seen that the accuracy of the model decays as the forecasting steps increase.

Figures 6.17,6.18 and 6.19 presents the out-of-sample predictions with one-step forecasts vs. the real occupancy. It is possible to see that the forecasts fall short for the highest and lowestoccupancy values. What is more, the models' forecasts simply replicate the most recent observations.

6.5 Barcelona ARIMA models

6.5.1 Regular sector

This section includes the experiment that corresponds to the Regular sector of Barcelona, namely *ARIMA_ barcelona_regular*.

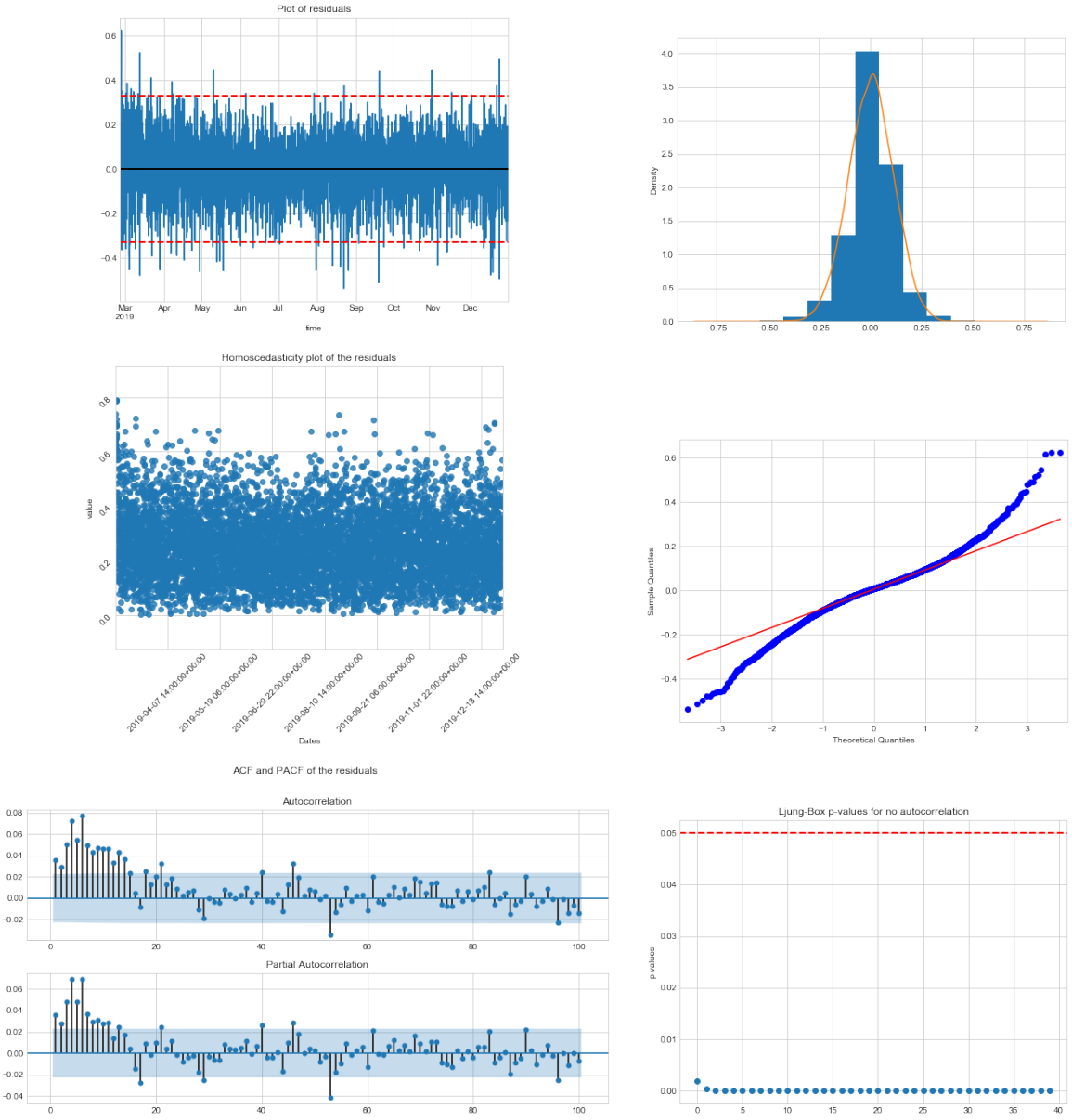


Figure 6.14: Antwerp Van Schoonhovestraat Zone 1 residual analysis of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots, and Q-statistic p-values plot using the Ljung–Box statistical test.

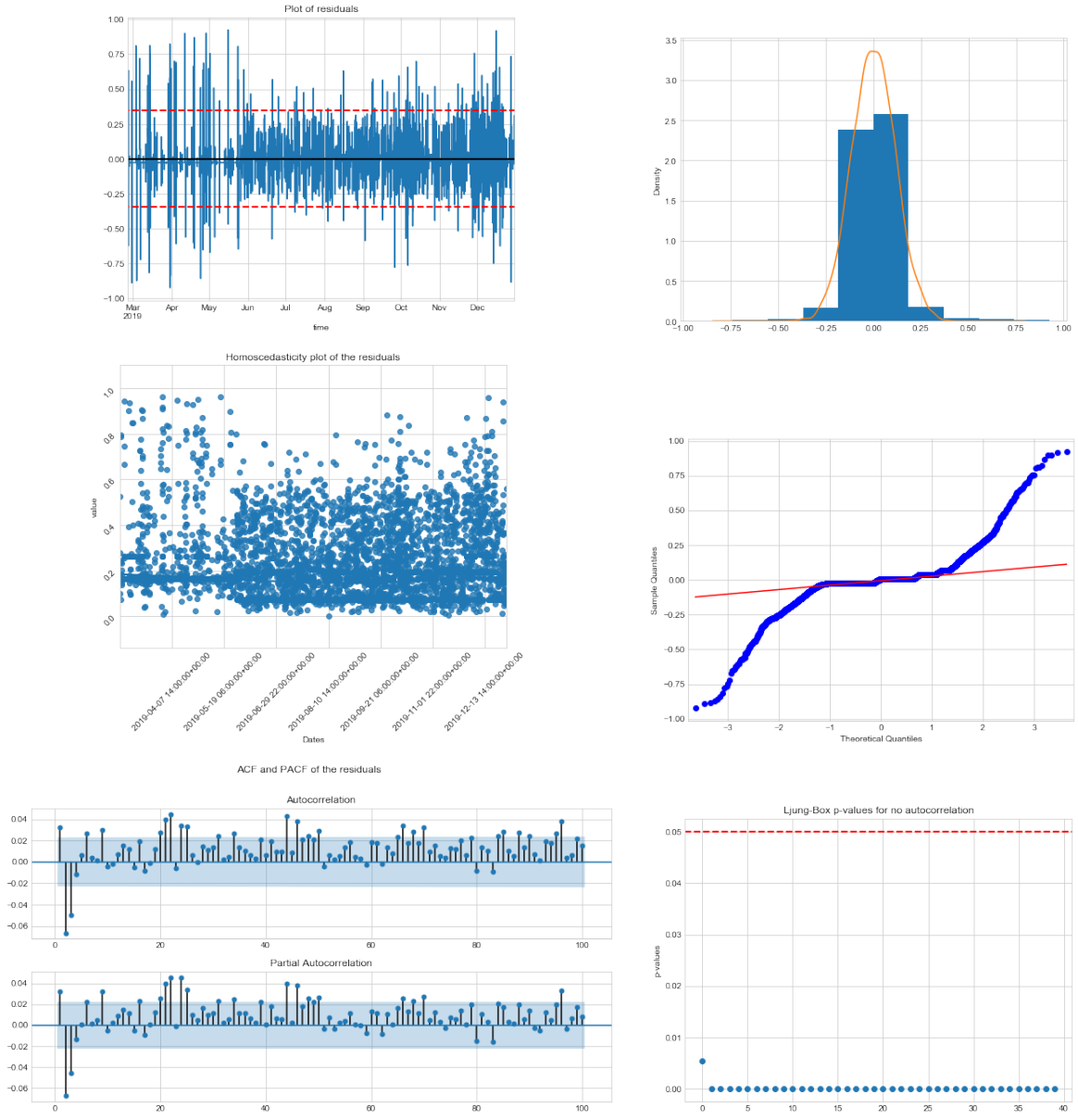


Figure 6.15: Antwerp Van Schoonhovestraat Zone 2 residual analysis of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot using the Ljung–Box statistical test.

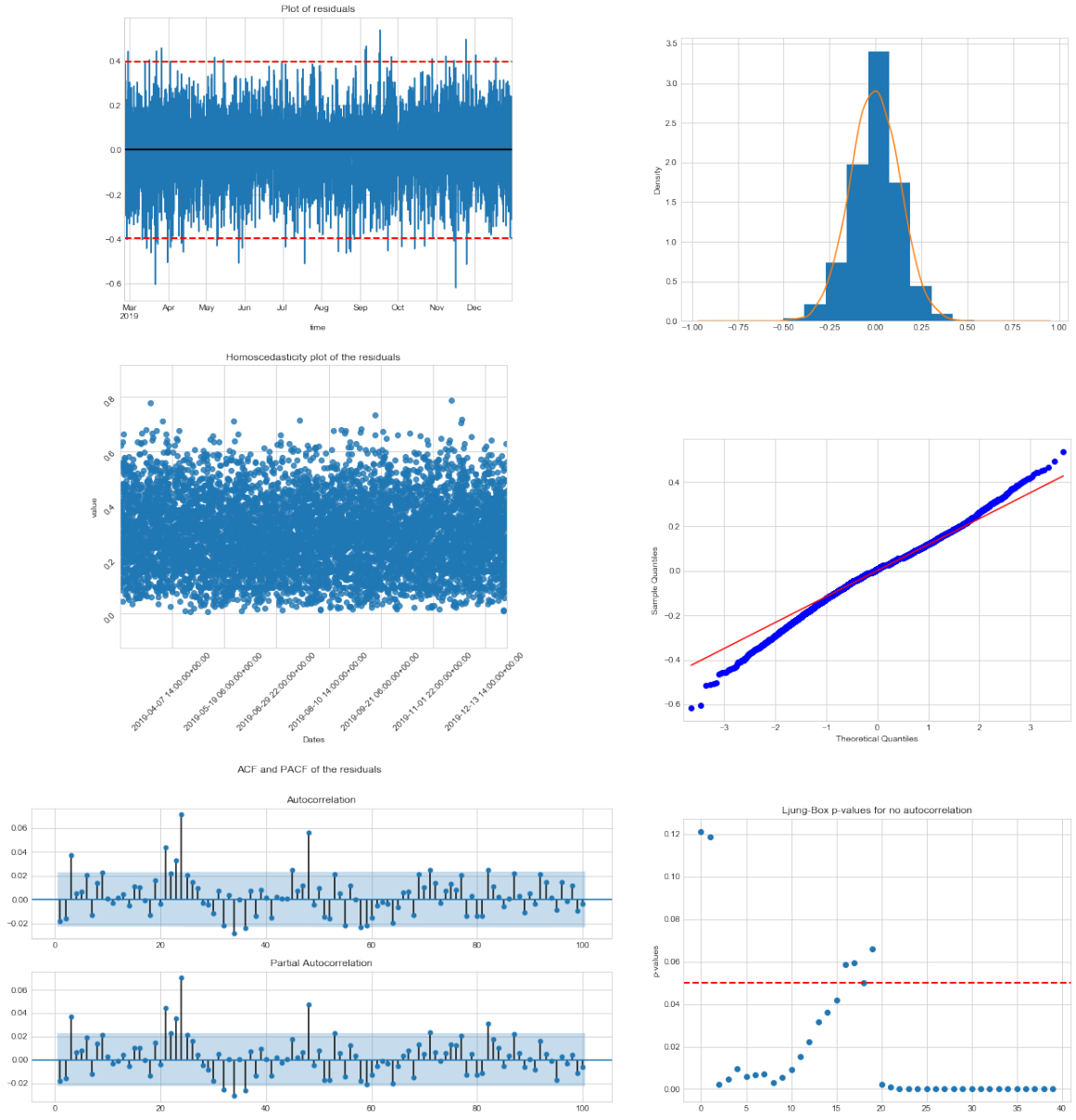


Figure 6.16: Antwerp Van Schoonhovestraat Zone 3 residual analysis of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot using the Ljung–Box statistical test.

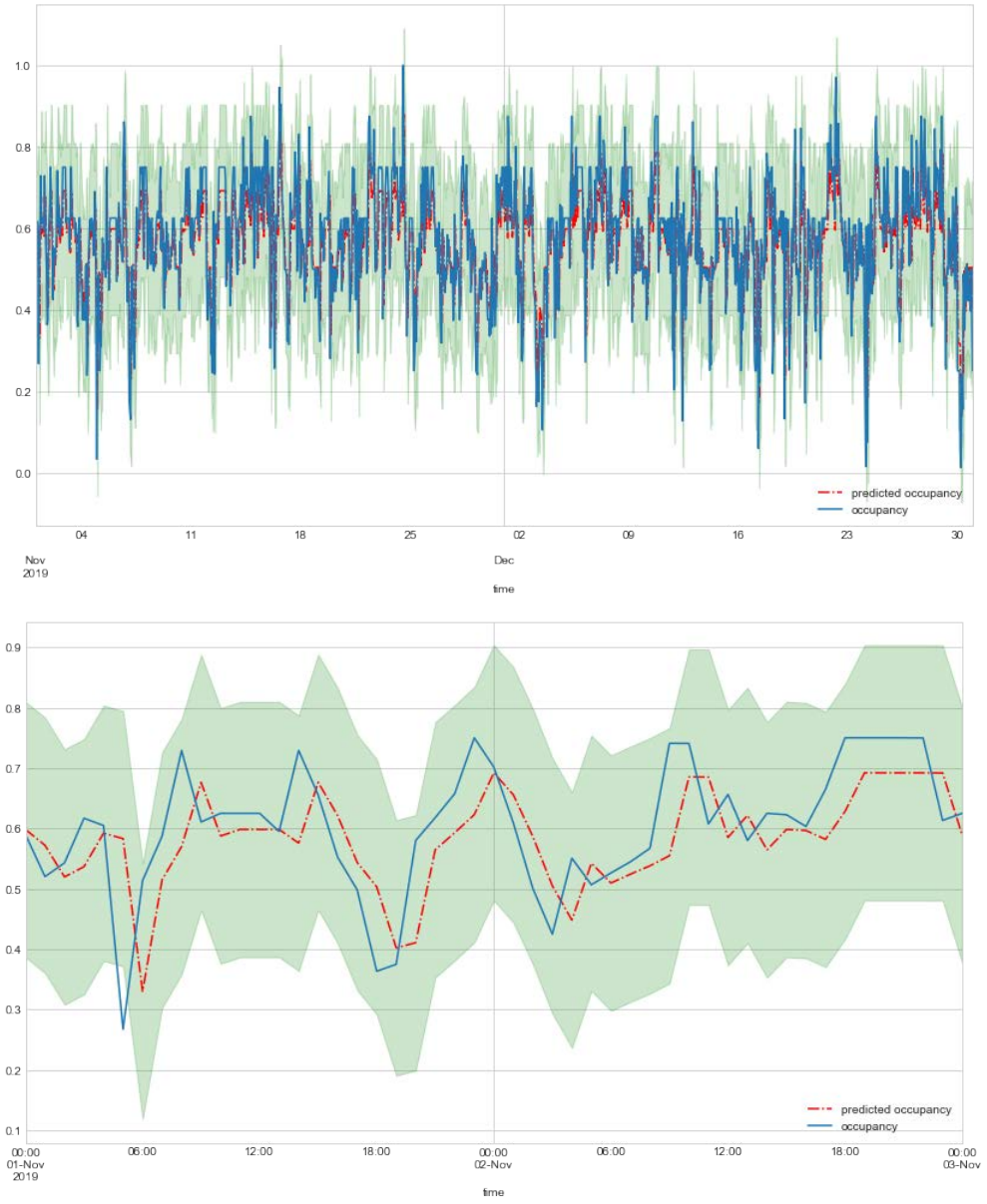


Figure 6.17: Antwerp Van Schoonhovestraat Zone 1 ARIMA forecasts. Blue denotes the observed occupancy, red the occupancy forecasted by the model, and green is the confidence interval. Top shows the entire out-of-sample interval of data while bottom shows out-of-sample predictions for two days.

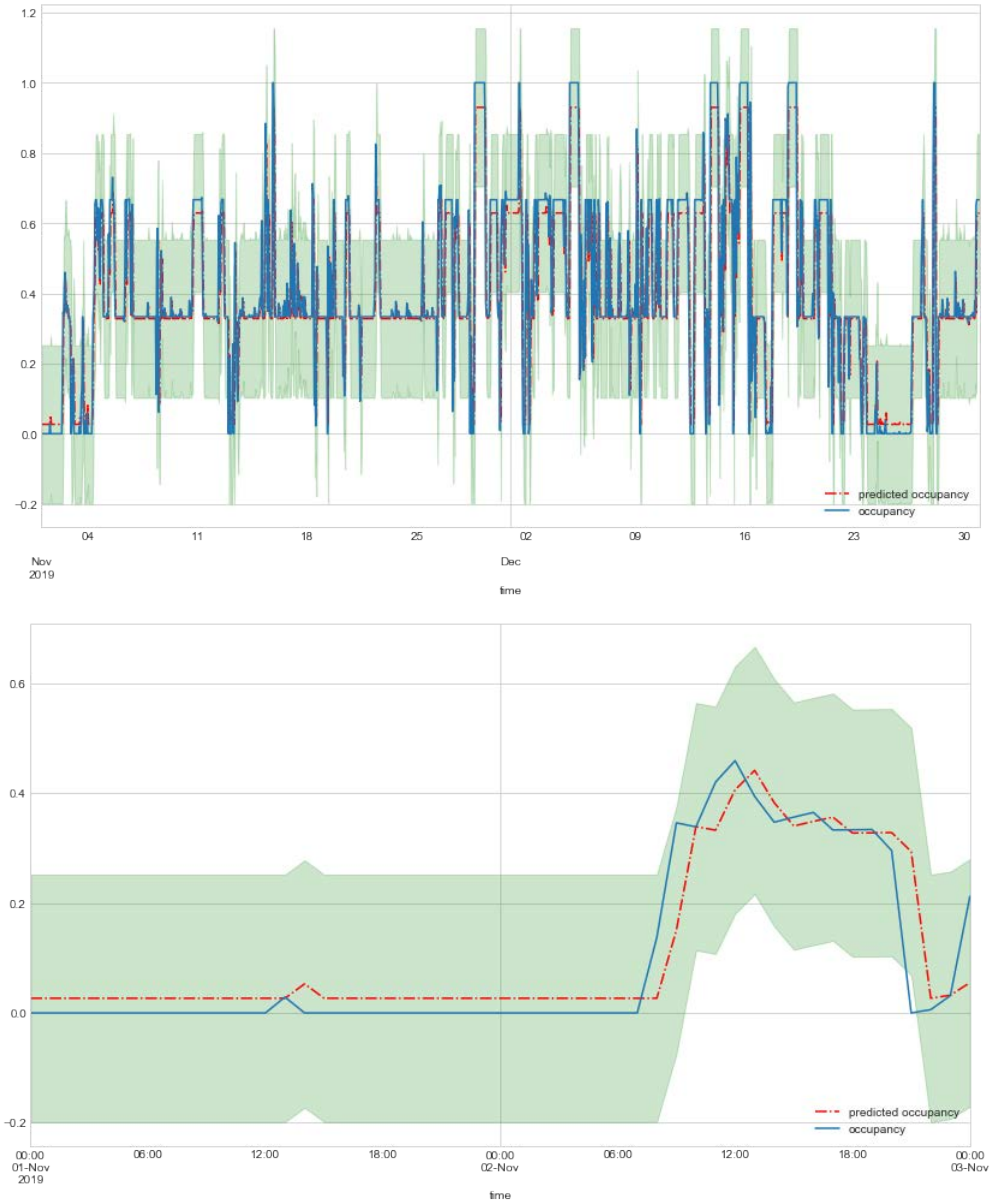


Figure 6.18: Antwerp Van Schoonhovestraat Zone 2 ARIMA forecasts. Blue denotes the observed occupancy, red the occupancy forecasted by the model, and green is the confidence interval. Top shows the entire out-of-sample interval of data while bottom shows out-of-sample predictions for two days.

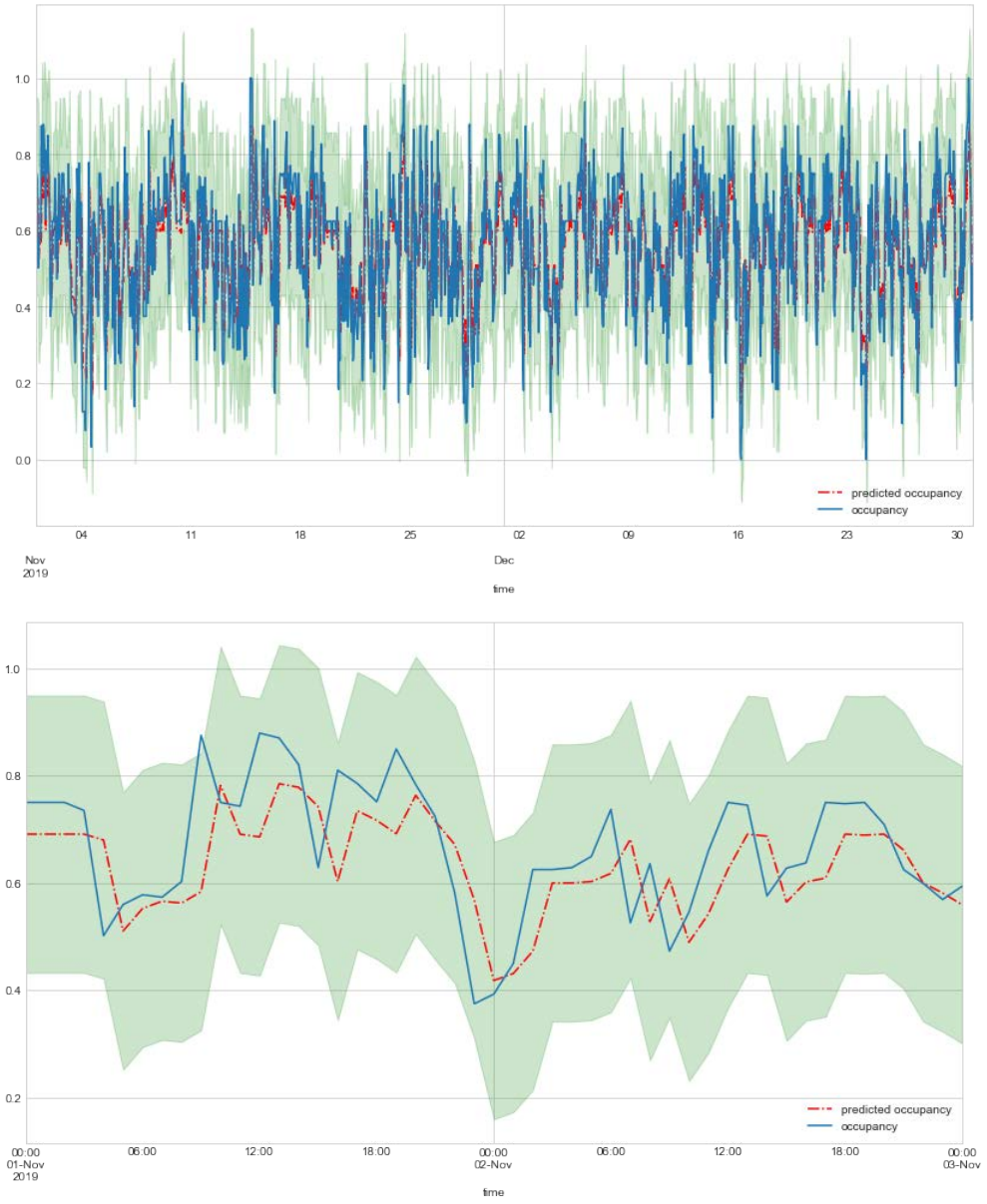


Figure 6.19: Antwerp Van Schoonhovestraat Zone 3 (bottom) ARIMA forecasts. Blue denotes the observed occupancy, red the occupancy forecasted by the model, and green is the confidence interval. Top shows the entire out-of-sample interval of data while bottom shows out-of-sample predictions for two days.

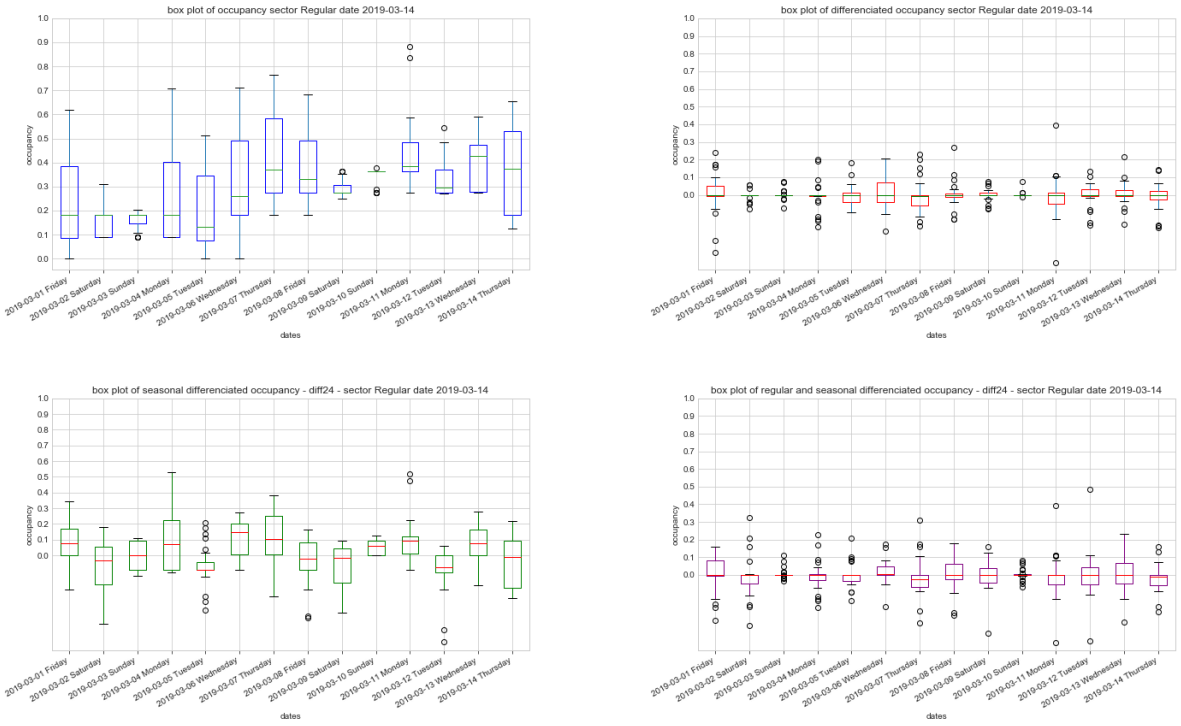


Figure 6.20: Box plots of the Barcelona Regular sector for the first two weeks of March 2019. Top left shows the non-differentiated series, top right the differentiated one, bottom left presents the seasonal differentiated (daily) and bottom right is the series with one regular differentiation plus one seasonal differentiation.

Daily box plots for the Barcelona sector Regular (see Figure 6.20) present the behavior of the mean and variance through the first two weeks of March 2019 and under various differences applied to the data: non-differentiated, one regular differentiation, one seasonal (daily) differentiation and one regular plus one seasonal differentiation. It can be observed that the mean has different levels for weekdays and weekends, meaning that perhaps the series are non-stationary. In the seasonal plots presented in Chapter 3.5.1, this behavior was already observed. When applying one regular differentiation to the series, the mean and variance seem to become stationary. With one seasonal differentiation, the results are similar to the non-differentiated series. For the one regular plus one seasonal differentiation, the results are similar to the one regular differentiated series. The non-differentiated series must also be considered, as the long-term plots in Section 3.5.1 indicate that the mean is constant over the years.

Table 6.12 shows the mean and variance of the Barcelona sector under different orders of differentiation. The minimum variance is presented when using one regular differentiation and, in the case of seasonal differentiation, when using one regular plus one seasonal.

The ACF and the PACF visualizations of the Barcelona sector presented in Figure

Difference order	Mean	Variance
No differentiation	0.2954	0.0430
One regular	0	0.0063
Two regular	0	0.0092
One seasonal	0.0006	0.0283
Two seasonal	-0.0002	0.0699
Regular plus seasonal	0	0.0078
Two regular, one seasonal	0	0.0150
One regular, two seasonal	0	0.0223
Two regular, two seasonal	0	0.0225

Table 6.12: Barcelona sector mean and variance when using different orders of differentiation.

Model	Log likelihood	AIC	BIC
$ARIMA(2,0,0)$	14376.726	-28745.452	-28715.979
$SARIMA(2,1,2)(0,1,1)_{24}$	15682.740	-31353.480	-31309.283

Table 6.13: Barcelona regular sector ARIMA models scores comparison.

6.21 are somehow similar to those of Antwerp. The candidate models to consider are the AR(2) for the non-differentiated series and AR(2), MA(2) and SMA(1) for the one regular plus one seasonal differentiated series. For the other series presented, they present a large amount of correlations that would require fitting a high number of parameters. Because of this, they are not considered. On the one hand, the series presents multiple seasonalities, as can be seen in the weekly seasonality at lag 168; on the other hand, lag 8 shows high value, meaning that occupancy levels change suddenly every eight hours.

Table 6.13 presents the scores of the candidate models. Of the models considered, $SARIMA(2,1,2)(0,1,1)_{24}$ is the one with the best AIC and BIC scores (the lower, the better) and higher likelihood.

From Equation 6.15, it can be seen that all parameters are significant.

$$\begin{aligned}\hat{x_t} = {} & 0.7258_{(0.049)}x_{t-1} + 0.1132_{(0.045)}x_{t-2} - 0.6973_{(0.048)}w_{t-1} \\ & - 0.2872_{(0.047)}w_{t-2} - 0.9523_{(0.002)}w_{t-24} + w_t,\end{aligned} \tag{6.15}$$

with $\sigma^2 = 0.0040$. Lower values in parenthesis correspond to the estimates of the standard errors for each parameter. w_{t-n} and x_{t-n} represent, respectively, the white noise values and the past series values for a number of n lags considered.

The visualizations in Figure 6.22 describe the residuals of the proposed model $SARIMA(2,1,2)(0,1,1)_{24}$. The residuals do not follow a normal distribution because some lags have significant correlations, as seen in the ACF and PACF plots. The fact that there are correlations in the residuals at specific lags indicates that the model is

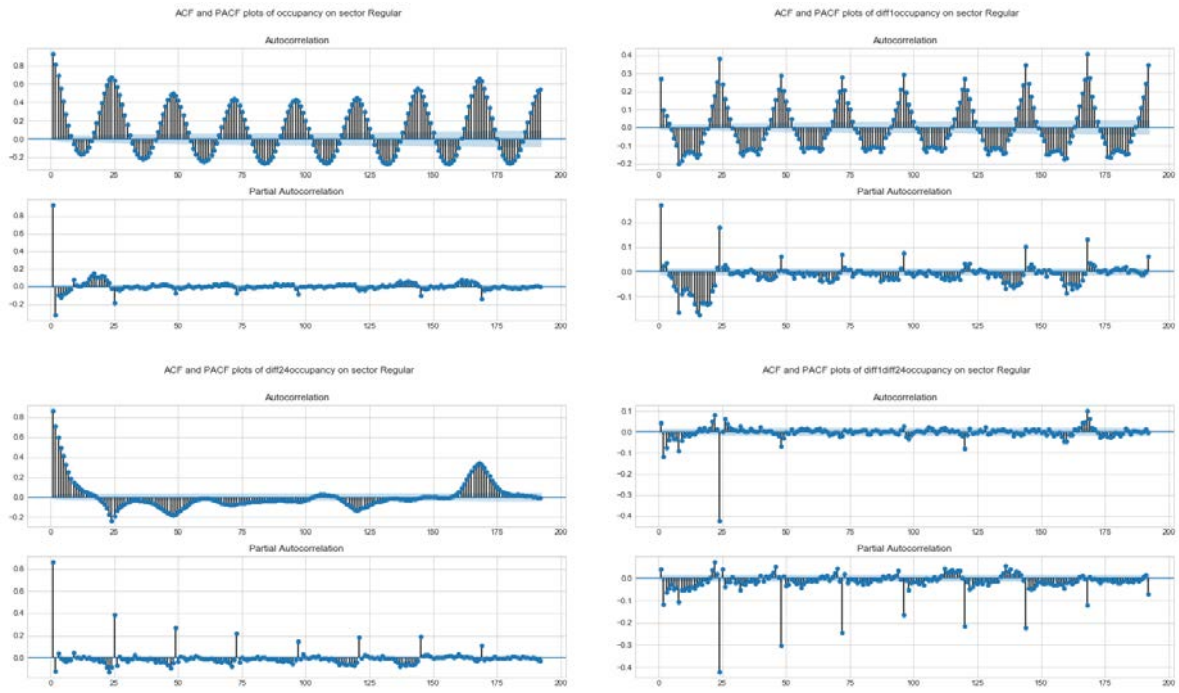


Figure 6.21: Barcelona ACF and PACF plots for the non-differentiated series, the one differentiated, the seasonal differentiated and the one regular plus one seasonal differentiated.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
0.0656	0.0929	0.1079	0.1157	0.1218	0.1277

Table 6.14: Barcelona sector regular ARIMA RMSE at different forecast steps.

unable to capture multiple seasonalities like those that arise every eight hours or the daily ones.

The time interval of the training data is from January 2018 until the end of October 2019. The time interval for the test data is from November 2019 to the end of December 2019. A multi-step strategy is used to study the effect of the forecasts at different time horizons. Table 6.14 presents the RMSE of the model at each of the considered forecasting hours, from one to six.

One-step forecasts of the model proposed for Barcelona are shown in Figure 6.23. The first visualization presents forecasts for the entire interval of the out-of-sample data. The second visualization shows the first 48 elements of the same date interval. The visualizations indicate that the model fails to capture the patterns on weekends, as it predicts higher occupancy levels. In the 48-hour plot, we can see more clearly that the forecast values are higher than the real occupancy.

6.6 Wattens ARIMA models

6.6.1 Swarco sectors

This section includes the experiments corresponding to the two sectors of Wattens: Inside and Outside. The corresponding experiments are: *ARIMA_ wattens_inside* and *ARIMA_ wattens_outside*.

Daily box plots for the Wattens sectors Inside and Outside are presented in Figure 6.24. The non-differentiated series show the behavior of the parking levels throughout the week. Here, we can distinguish normal day patterns (Monday to Thursday), Friday as a special day with fewer working hours (check Chapter 3.5.3 for more information on this) and, lastly, the days with minimum levels of occupancy that correspond to the weekends. Once a regular differentiation is made on the series, the mean and variance become lower and stationary.

Table 6.15 presents the evolution of the mean and variance under different orders of differentiation for the time series of the Inside and Outside sectors. Both sectors coincide in the sense that one differentiation yields lower mean and variance values for the regular part. When considering seasonal differentiation, the mean and variance are the lowest in the one regular plus one seasonal differentiation for both sectors.

The ACF and PACF plots of Wattens are presented in Figure 6.25. The non-differentiated series could be modeled with an AR(2) in both sectors. For the one regular plus one seasonal differentiation series, the AR(2), MA(2) and SMA(1) model is proposed for both

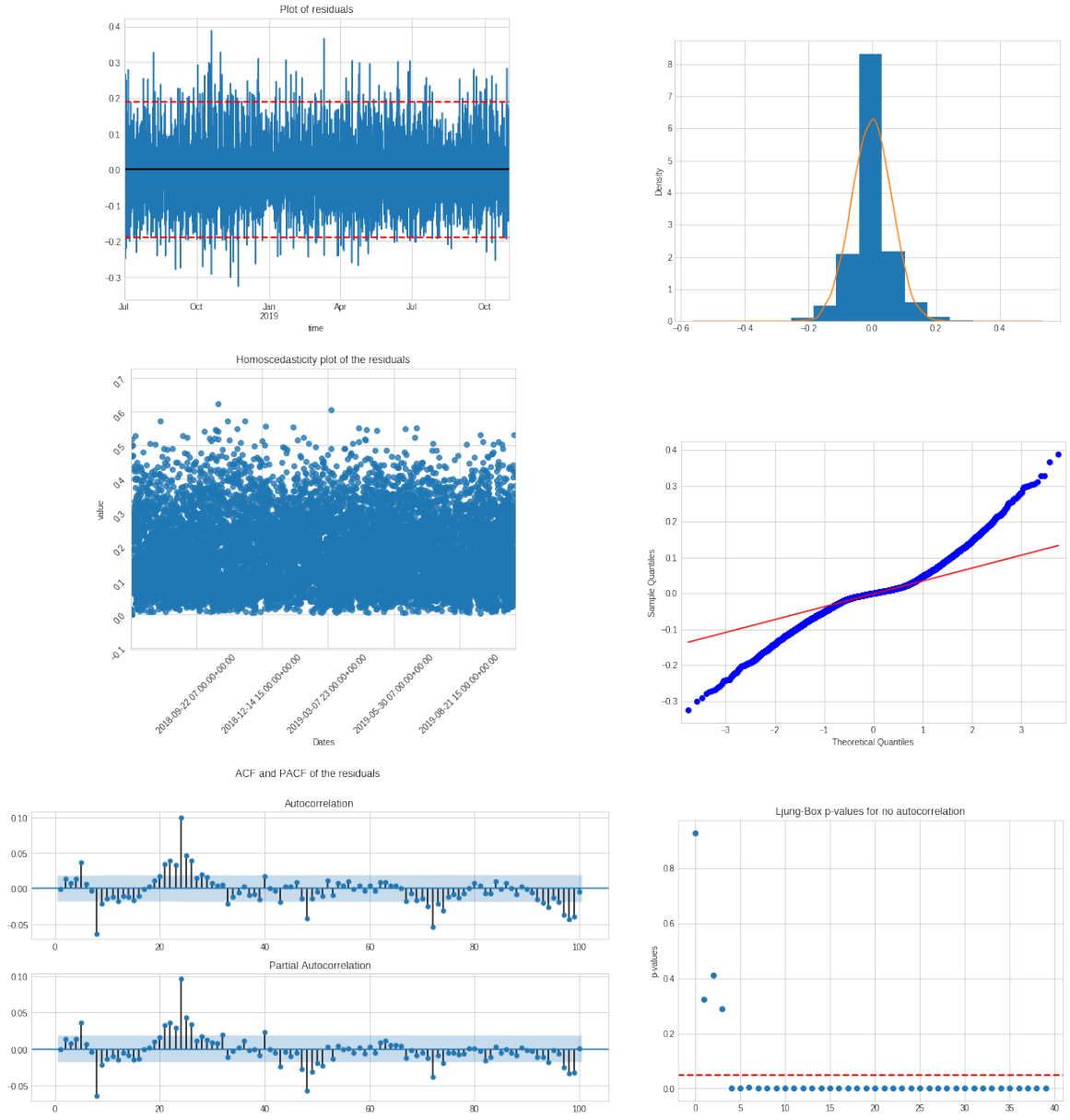


Figure 6.22: Barcelona regular sector visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

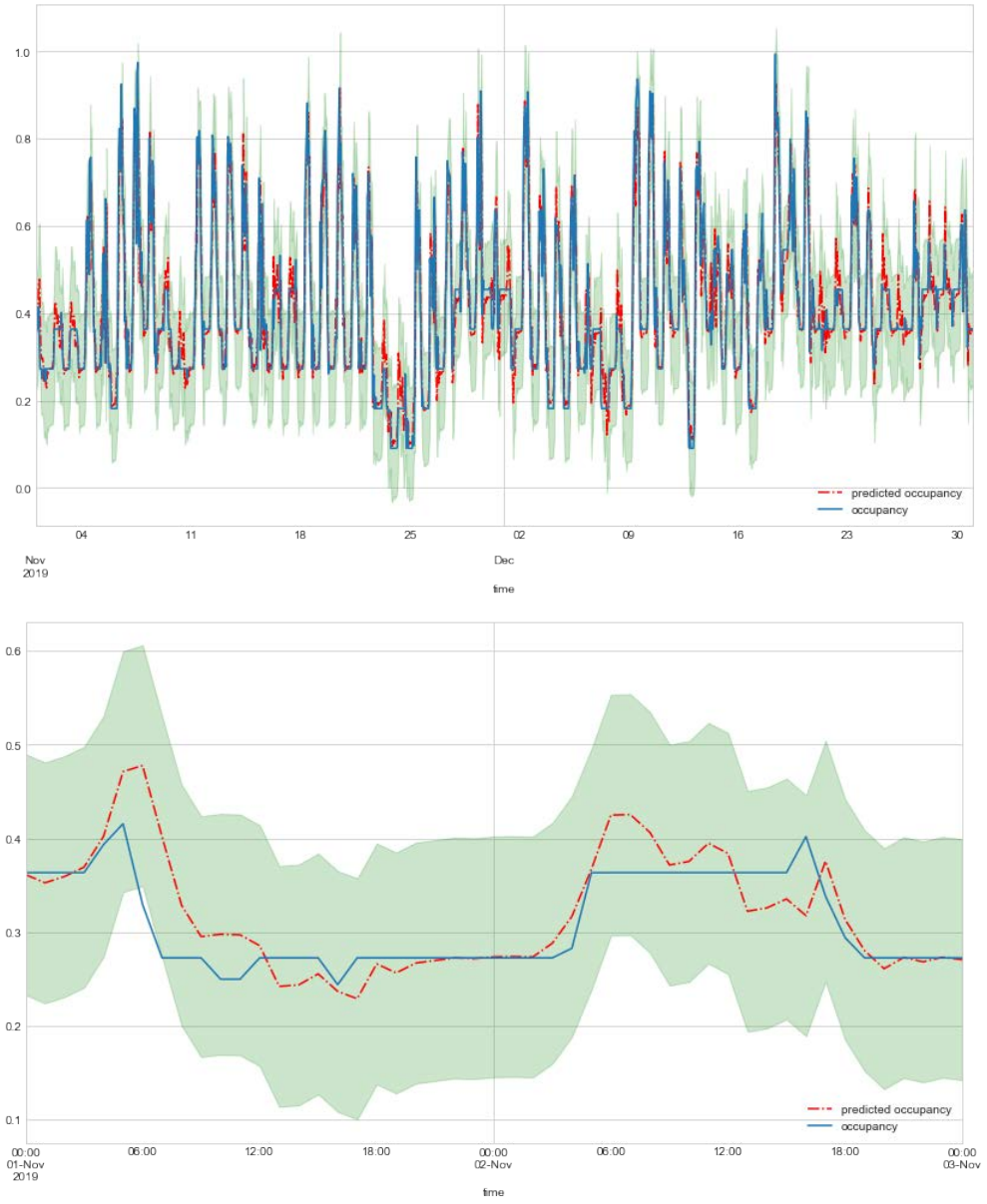


Figure 6.23: Barcelona ARIMA forecasts for the entire out-of-sample interval (top) and for the first two days of the same interval (bottom). Blue denotes the observed occupancy, red the occupancy forecasted by the model, and green is the confidence interval of the predictions.

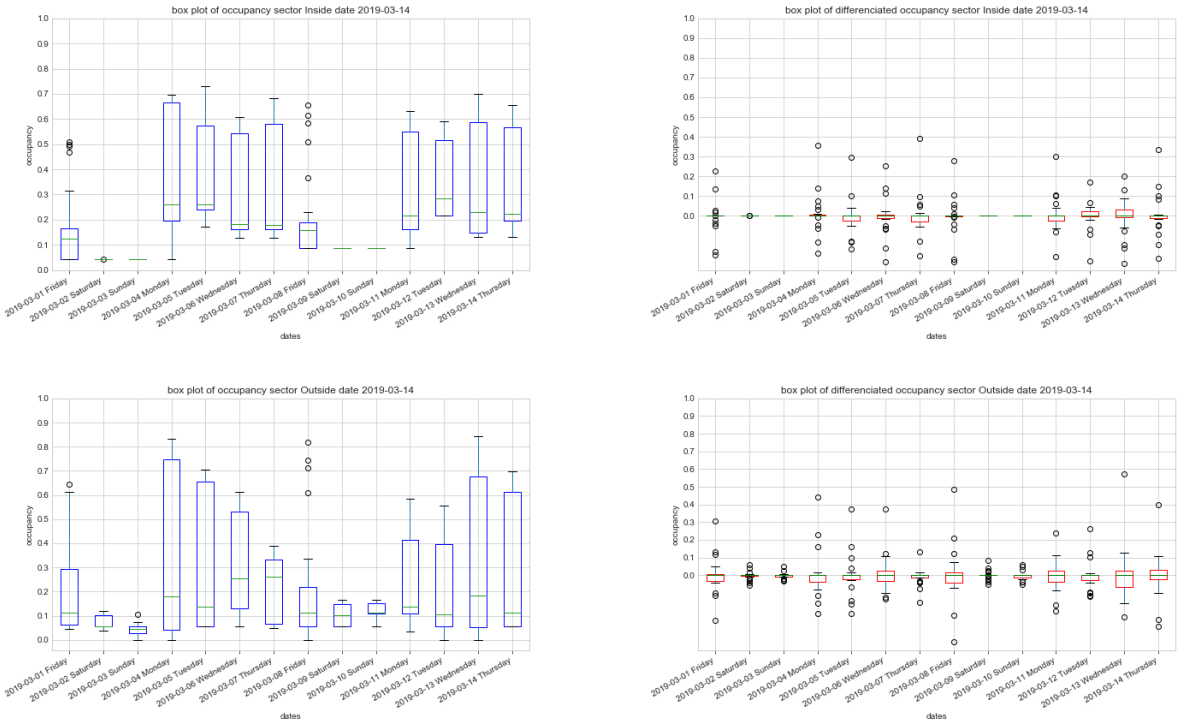


Figure 6.24: Daily box plots for sectors Inside (top) and Outside (bottom) of Wattens. The first column visualizations are for non-differentiated series and those in the second column are from the one regular differentiation

Difference order	Inside - Mean	Inside - Variance	Outside - Mean	Outside - Variance
No differentiation	0.2490	0.0427	0.2342	0.0677
One regular	0	0.0056	0	0.0089
Two regular	0	0.0056	0	0.0089
One seasonal	-0.0008	0.033	0.001	0.0467
Two seasonal	0.0005	0.0746	0.0002	0.1006
Regular plus seasonal	0	0.0049	0	0.0074
Two regular, one seasonal	0	0.0062	0	0.0099
One regular, two seasonal	0	0.0127	0	0.0192
Two regular, two seasonal	0	0.0127	0	0.0194

Table 6.15: Wattens sectors mean and variance values when using different orders of differentiation.

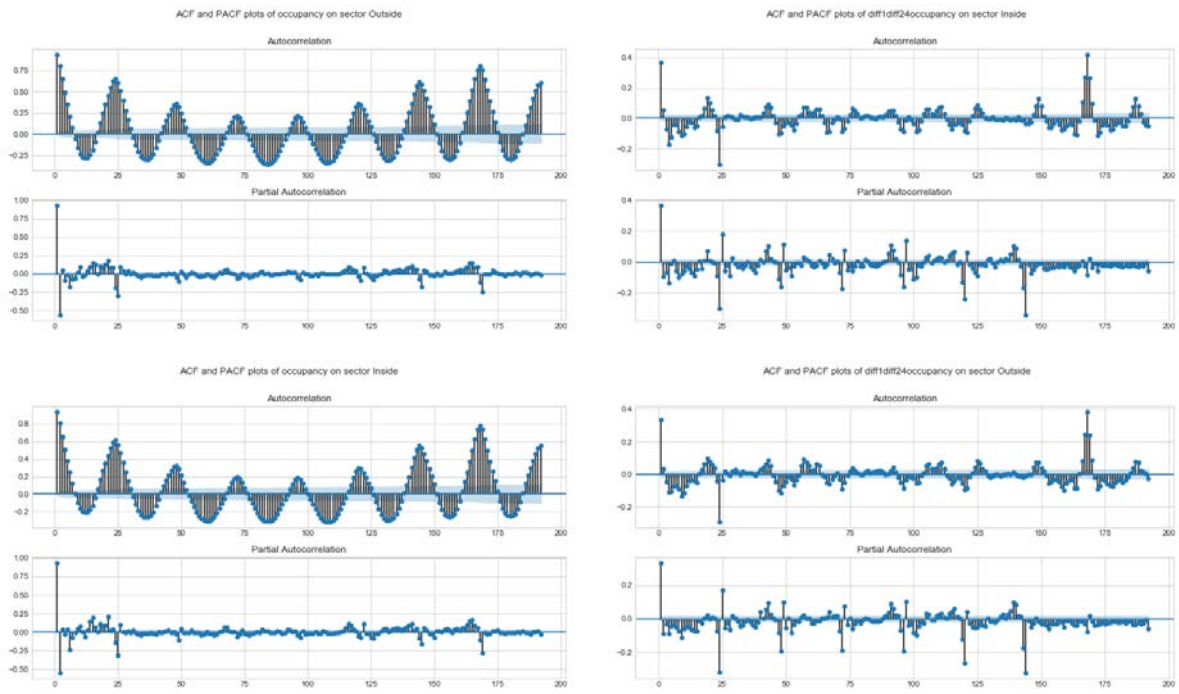


Figure 6.25: Wattens ACF and PACF plots for the sectors Inside (top) and Outside (bottom). Left is the non-differentiated series and right is the one regular plus one seasonal differentiated.

sectors. Similarly to what happened in other sectors, strong clues exist for multiple seasonality, at least daily and weekly.

Table 6.16 presents the scores of the two candidate models for both sectors. Clearly, the model $SARIMA(2,1,2)(0,1,1)_{24}$ is the one with the best scores. Because of this, it is the model proposed for the Wattens sectors.

Equation 6.16 and Equation 6.17 present the mathematical forms of the propose model $SARIMA(2,1,2)(0,1,1)_{24}$ for the Inside and Outside sectors, respectively. All the parameters are significant under the t-test. Notice that the first AR parameter of the models is greater than one, which forces the second AR parameter to become negative in order to preserve stationarity, as the sum of the AR parameters must be less than

Sector	Model	Log likelihood	AIC	BIC
Inside	$ARIMA(2,0,0)$	12895.923	-25783.847	-25755.219
Inside	$SARIMA(2,1,2)(0,1,1)_{24}$	14679.551	-29347.103	-29304.177
Outside	$ARIMA(2,0,0)$	10685.168	-21362.337	-21333.710
Outside	$SARIMA(2,1,2)(0,1,1)_{24}$	12435.992	-24859.985	-24817.060

Table 6.16: Wattens scores for the Inside and Outside sectors for the ARIMA models considered.

Sector	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
Inside	0.0501	0.0825	0.1060	0.1244	0.1368	0.1465
Outside	0.0587	0.0989	0.1248	0.1436	0.1561	0.1665

Table 6.17: Wattens Inside and Outside ARIMA RMSE.

one.

$$\hat{x_t} = 1.2662_{(0.024)}x_{t-1} - 0.3865_{(0.023)}x_{t-2} - 0.9264_{(0.025)}w_{t-1} - 0.0326_{(0.026)}w_{t-2} - 0.9402_{(0.002)}w_{t-24} + w_t, \tag{6.16}$$

with $\sigma^2 = 0.0027$. The lower values in parenthesis correspond to the estimates of the standard errors for each parameter. w_{t-n} and x_{t-n} represent, respectively, the white noise values and the past series values for a number of n lags considered.

$$\hat{x_t} = 1.2706_{(0.023)} - 0.3694_{(0.022)}x_{t-2} - 0.9020_{(0.024)}w_{t-1} - 0.0859_{(0.024)}w_{t-2} - 0.9214_{(0.002)}w_{t-24} + w_t, \tag{6.17}$$

with $\sigma^2 = 0.0042$

Figure 6.26 presents visualizations for the Inside sector and determines whether the residuals from the model have a normal distribution. The proposed model is able to capture most of the information, as seen in the residual plot, but it fails with extreme values. The density plot shows that the residuals do not follow a normal distribution and, in fact, the Q-Q plot indicates that there are heavy tails in their distribution. The ACF and the PACF show that significant correlations still exist in the data that the model is not able to capture. Finally, the visualization of the Ljung-Box test confirms that residuals do not follow a normal distribution, as none of the lags has a p-value over 0.05.

The plots of the residuals for the Outside sector are presented in Figure 6.27. Similarly to what happens in the Inside sector, the model fails to capture all the patterns and, thus, the residuals do not follow a normal distribution. Specifically, the model residuals show correlations at lags greater than two and at seasonal lags like 24, 48, etc.

The time interval of the training data is from March 2018 until the end of March 2019. The time interval for the test data is from April 2019 to the end of May 2019. A multi-step strategy is used to study the effect of the forecasts at different time horizons. Table 6.17 presents the RMSE of the model at each of the considered forecasting hours, from one to six.

Figures 6.28 and 6.29 show the one-step forecasts for the ARIMA models proposed for the Wattens sectors. It can be observed that the models fail especially on weekends, where no changes in the occupancy are expected but the model forecasts change. In the two-day plot, it is possible to see that the model gives forecasts slightly below zero.

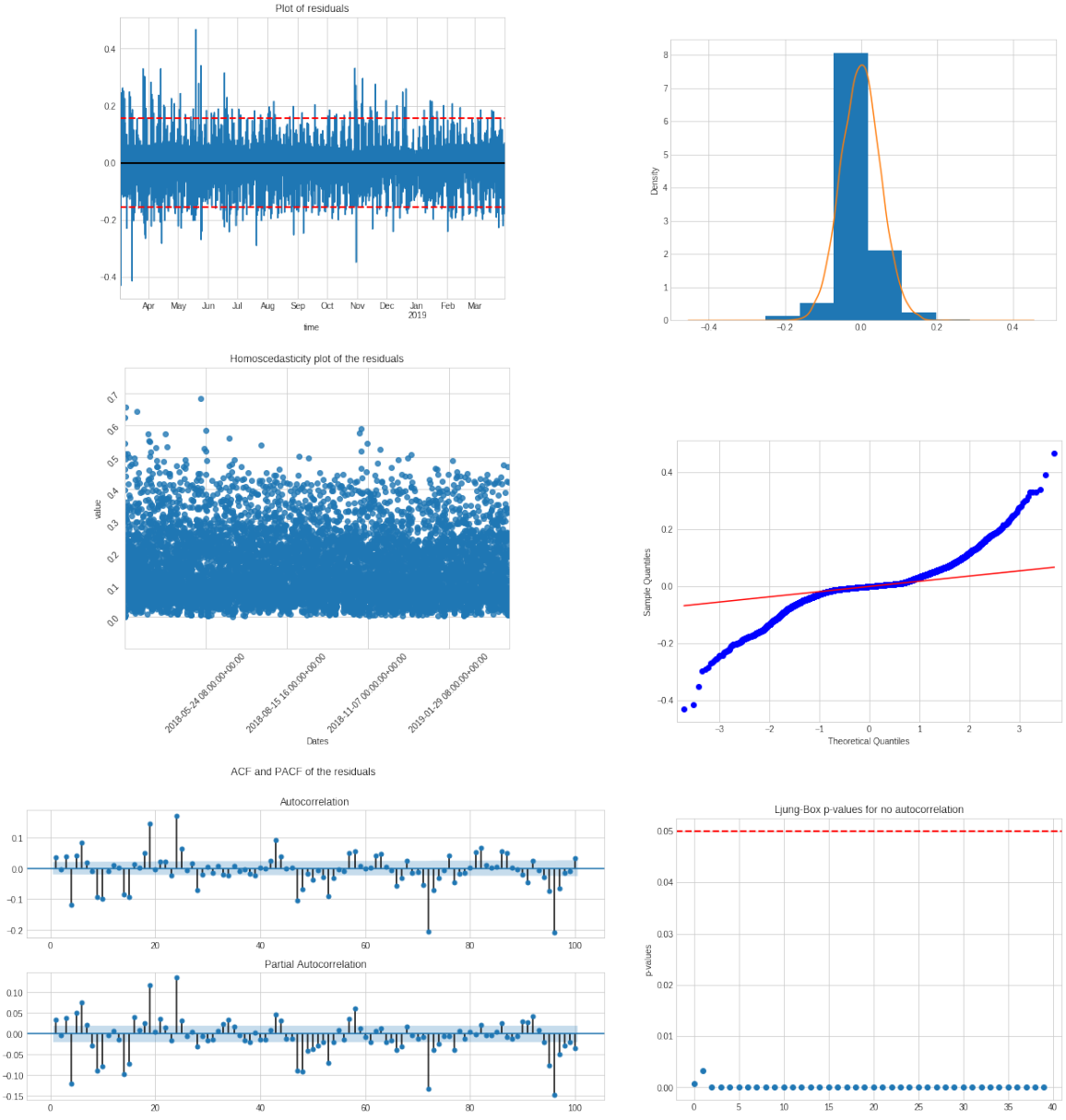


Figure 6.26: Wattens Inside sector visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

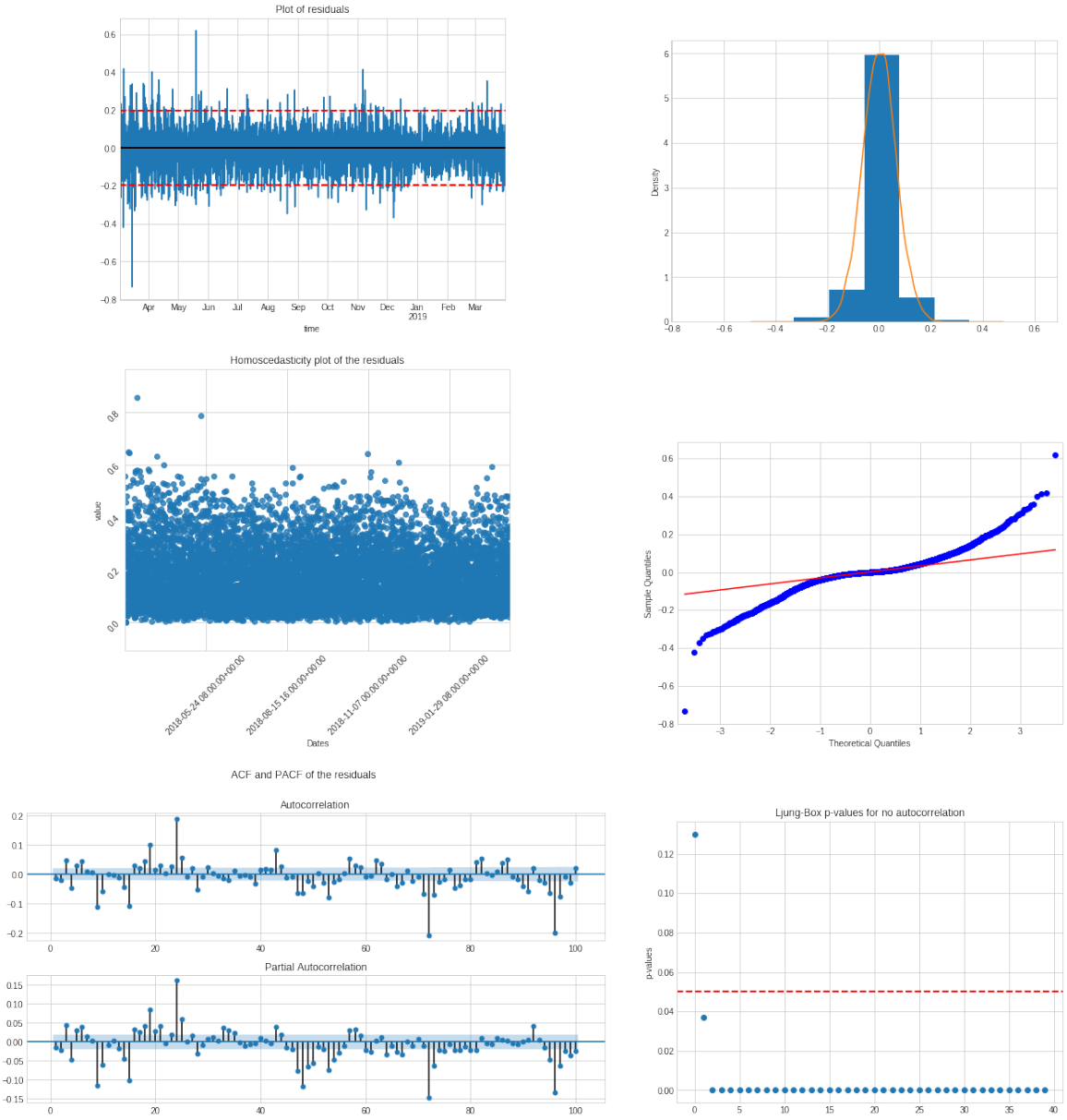


Figure 6.27: Wattens Outside sector visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

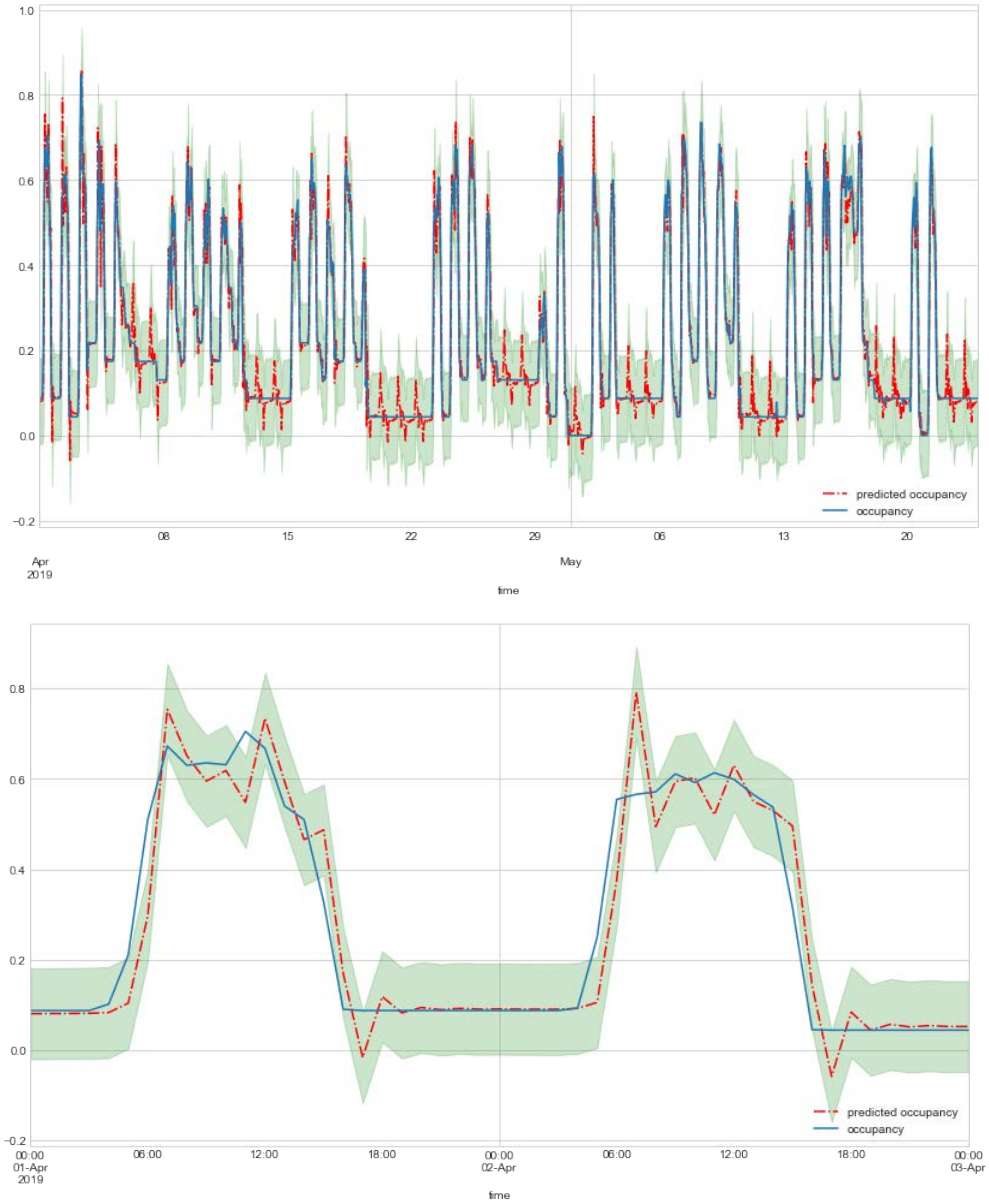


Figure 6.28: Wattens ARIMA forecasts for the Inside sectors. The top presents forecasts for the entire out-of-sample interval while the bottom shows the first two days of the same time interval. Blue denotes the observed occupancy, red the occupancy forecasted by the model, and green is the confidence interval.

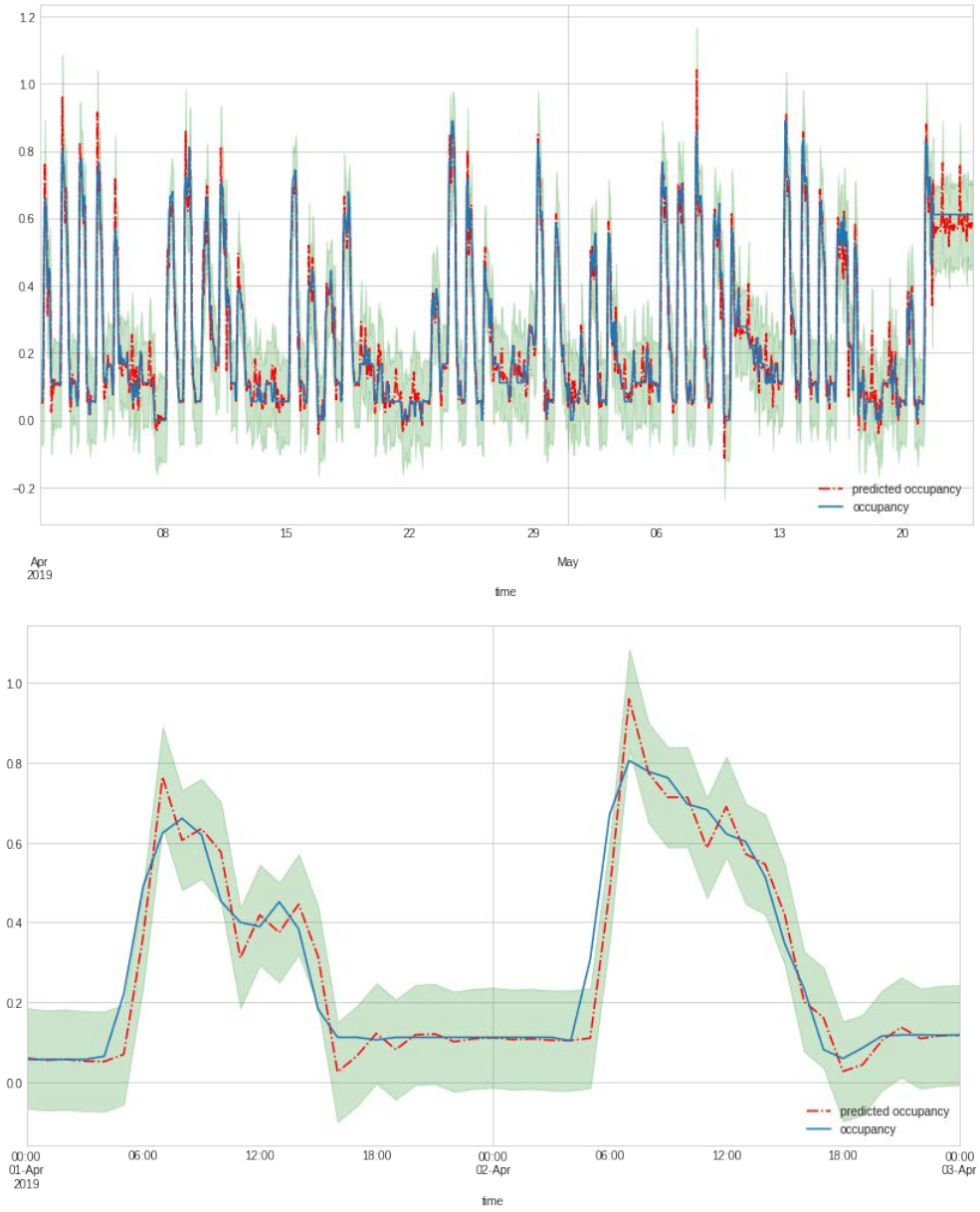


Figure 6.29: Wattens ARIMA forecasts for the Outside (bottom) sectors. The top presents forecasts for the entire out-of-sample interval while the bottom shows the first two days of the same time interval. Blue denotes the observed occupancy, red the occupancy forecasted by the model, and green is the confidence interval.

6.7 Los Angeles ARIMA models

6.7.1 Standard sector

This section includes the experiment corresponding to the Standard sector in Los Angeles, namely *ARIMA_ LA_standard*.

Chapter 3.5.4 shows that the Willow zone, where the sectors are located, underwent street works in early 2019. Because of this, this section provides a comparison of the situation prior to 2019, the situation in 2019 and the situation with all the data used (2018 and 2019) in order to detect how an external change to the occupancy process affects the modeling. This implies that, for some of the Los Angeles Willow sectors, the information is presented from three different intervals of dates.

Daily box plots for the Standard sector (Figure 6.30) for the first two weeks of November 2018 show that the untransformed time series has two different patterns in which it is easy to distinguish weekdays and weekends. Once the series is transformed using differentiation operations, the mean seems to have a value of 0 and the variance is reduced in the case of one regular differentiation and one regular plus one seasonal differentiation. In the one daily differentiation (bottom left), Mondays and Saturdays present very high variance than the other days.

Figure 6.31 presents the daily box plots for the first two weeks of July 2019. Because there were fewer parking spots available, the occupancy levels dropped to very low values. This has an effect on the variance, which has been greatly reduced.

Fortunately, as the works end in the zone, the levels of parking occupancy slowly return to what they were previously. Figure 6.32 shows how the occupancy patterns at the end of 2019 are similar to those for the year 2018. Because the levels of occupancy and patterns are slowly returning to those prior to the works in the zone, the proposed model is trained with all the available data; but we test the last two months of data with the expectation that the models become adapted to changes in occupancy patterns. We do the same for the other sectors in the zone, as some of them are affected by the works and others are not.

Table 6.18 presents the mean and variance for each of the years and the entire interval of data under different differentiation orders. As expected, the values of the mean and variance are lower in 2019 than those in 2018; and, when taking into account the entire interval of data available, the values are more or less in between. The lower levels of variance are achieved with two regular differentiations. When using seasonal differentiation, the lowest levels of variance are achieved with two regular plus one seasonal differentiation.

The column in Figure 6.33 shows the ACF and PACF plots for the Los Angeles Standard sector of the non-differentiated series for the years 2018, 2019 and when using all the available data. The visualizations show nearly identical patterns, with slightly lower values in 2019. This indicates that a model trained with all the available data will have

Figure 6.30: Daily box plots of the Los Angeles Standard sector from the first two weeks of November 2018 under different numbers of differentiations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

Difference order	2018 - Mean	2018 - Variance	2019 - Mean	2019 - Variance	All - Mean	All - Variance
No differentiation	0.2856	0.1145	0.1013	0.0265	0.1899	0.0773
One regular	0	0.0098	0	0.0029	0	0.0063
Two regular	0	0.0053	0	0.0027	0	0.0039
One seasonal	0.0003	0.0648	0.0003	0.0136	0.0002	0.0382
Two seasonal	-0.0008	0.1276	-0.001	0.0271	-0.0005	0.0756
Regular plus seasonal	0	0.0046	0	0.0018	0	0.0032
Two regular, one seasonal	0	0.0038	0	0.0024	0	0.0031
One regular, two seasonal	0	0.0101	0	0.0044	0	0.0072
Two regular, two seasonal	0	0.0095	0	0.0064	0	0.0079

Table 6.18: Los Angeles Willow Standard sector mean and variance values when using different orders of differentiation for the years 2018, 2019 and the whole period of data.

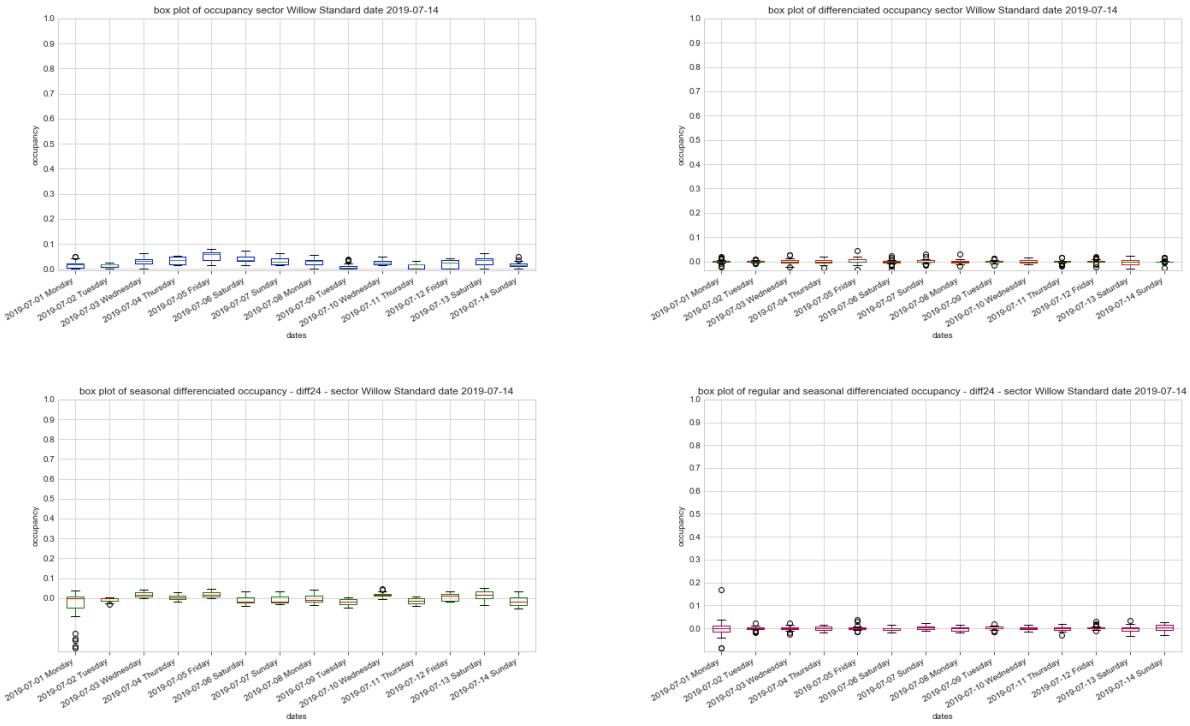


Figure 6.31: Daily box plots of the Los Angeles Standard sector from the first two weeks of July 2019 under different numbers of differentiations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

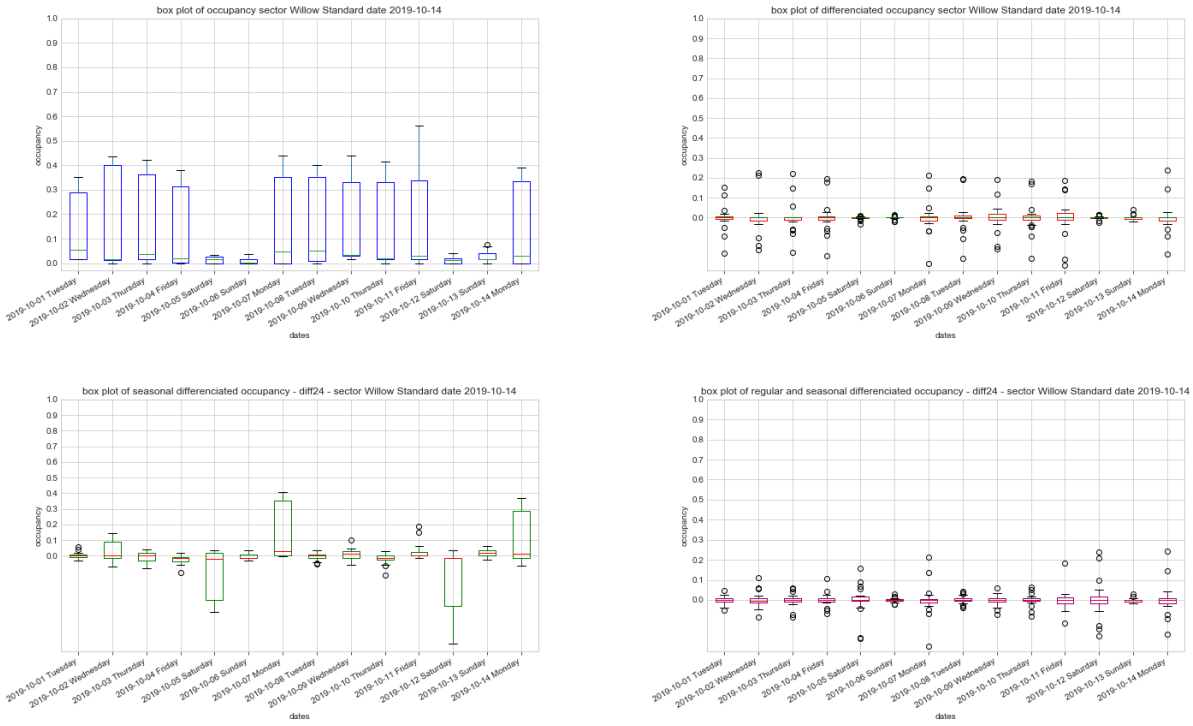


Figure 6.32: Daily box plots of the Los Angeles Standard sector from the first two weeks of October 2019 under different numbers of differentiations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

the same order as the one trained with only the 2018 data or only the 2019 data. The second column shows the same information for the one regular differentiation series. The visualization for the year 2019 presents discrepancies with the one for 2018, but the visualization of both years together presents patterns similar to those of 2018. From the non-differentiated series, it is possible to consider an AR(2) or an AR(2) with a seasonal AR(1). However, for simplicity, the AR(2) alone is considered. Similar conclusions can be extracted from the plot of the one regular differentiation. Notice that the one regular differentiation presents high correlations at higher lags, which can be understood as multiple seasonalities in the data. There is a seasonality around lag eleven. The reason for this is given in Chapter 3.5.4 and it is namely because, on the daily basis (weekdays), the occupancy of this sector rises at 6 am from minimum levels to maximum levels; and then, at 18 pm, the occupancy drops from very high levels to very low levels. It is also possible to see daily and weekly seasonality. For this series, an AR(10) can be considered for the regular part and SAR(1) for the seasonal part.

Visualizations of the one seasonal and one regular plus one seasonal time series are presented in Figure 6.34 for the three date intervals considered. The patterns presented in each column are more or less shared across the different time intervals. The visualizations of the one seasonal differentiation suggests using an AR(2), MA(10), but it is difficult to specify the order for the seasonal part, as multiple seasonalities arise. The one regular plus one seasonal differentiated series suggests an AR(1) and MA(2) for the regular part, and an SMA(2) for the seasonal part.

The minimum values for the variance were achieved at two regular differentiations. Figure 6.35 presents the ACF and PACF for such series. Again, the patterns are similar for the different time intervals considered. The series for two regular differentiations present very high correlations in a huge number of lags, perhaps because of over-differentiation. The two regular plus one seasonal differentiation suggests MA(2) and SMA(2) models.

After analyzing the different ACF and PACF plots, it can be concluded that using both 2018 and 2019 data to estimate the candidate models is adequate, as the patterns persist throughout the changes in the sector during 2019. This is also assumed for the other sectors in Los Angeles. More specifically, the data used for training the models are from April 2018 to August 2019, and the out-of-sample data are from September 2019 until the end of October 2019.

Table 6.19 presents the scores of the candidate models for the Los Angeles Standard sector. The proposed model is $SARIMA(1,1,2)(0,1,2)_{24}$ because it achieves the highest scores, although the $SARIMA(0,2,2)(0,1,2)_{24}$ is also a good candidate.

Equation 6.18 present the mathematical form of the proposed model $SARIMA(1,1,2)(0,1,2)_{24}$. Using the t-test, the second MA parameter is nearly not significant, as its t-value is 2.159; thus, it is possible to simplify the model by removing it.

$$\begin{aligned}\hat{x_t} = {} & 0.4487_{(0.034)}x_{t-1} + 0.1936_{(0.034)}w_{t-1} - 0.0482_{(0.022)}w_{t-2} \\ & - 0.6259_{(0.005)}w_{t-24} - 0.3135_{(0.005)}w_{t-48} + w_t,\end{aligned} \tag{6.18}$$

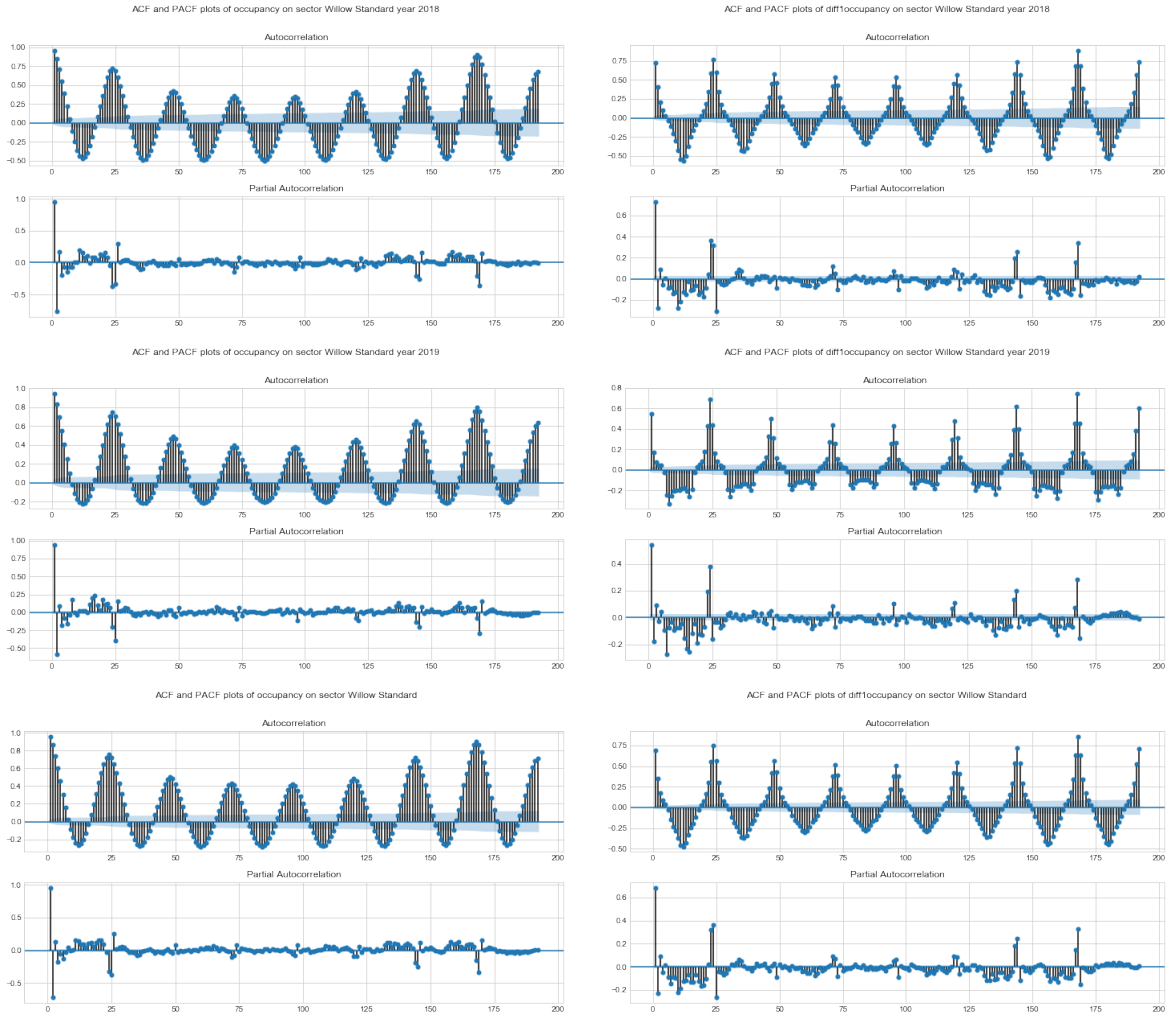


Figure 6.33: Los Angeles Willow Standard sector ACF and PACF plots for the non-differentiated (left) and one regular differentiated series (right). Top is for year 2018, middle for year 2019 and bottom is computed using all the available data.

Model	Log likelihood	AIC	BIC
$ARIMA(2,0,0)$	20495.470	-40982.941	-40952.832
$SARIMA(10,1,0)(1,0,0)_{24}$	21735.692	-43447.384	-43358.249
$SARIMA(2,1,10)(0,1,0)_{24}$	20689.075	-41352.151	-41255.613
$SARIMA(1,1,2)(0,1,2)_{24}$	22758.626	-45505.251	-45460.695
$SARIMA(0,2,2)(0,1,2)_{24}$	22473.901	-44937.803	-44900.673

Table 6.19: Scores from the candidate ARIMA models for the Standard sector. The higher the log likelihood and lower the AIC/BIC, the better.

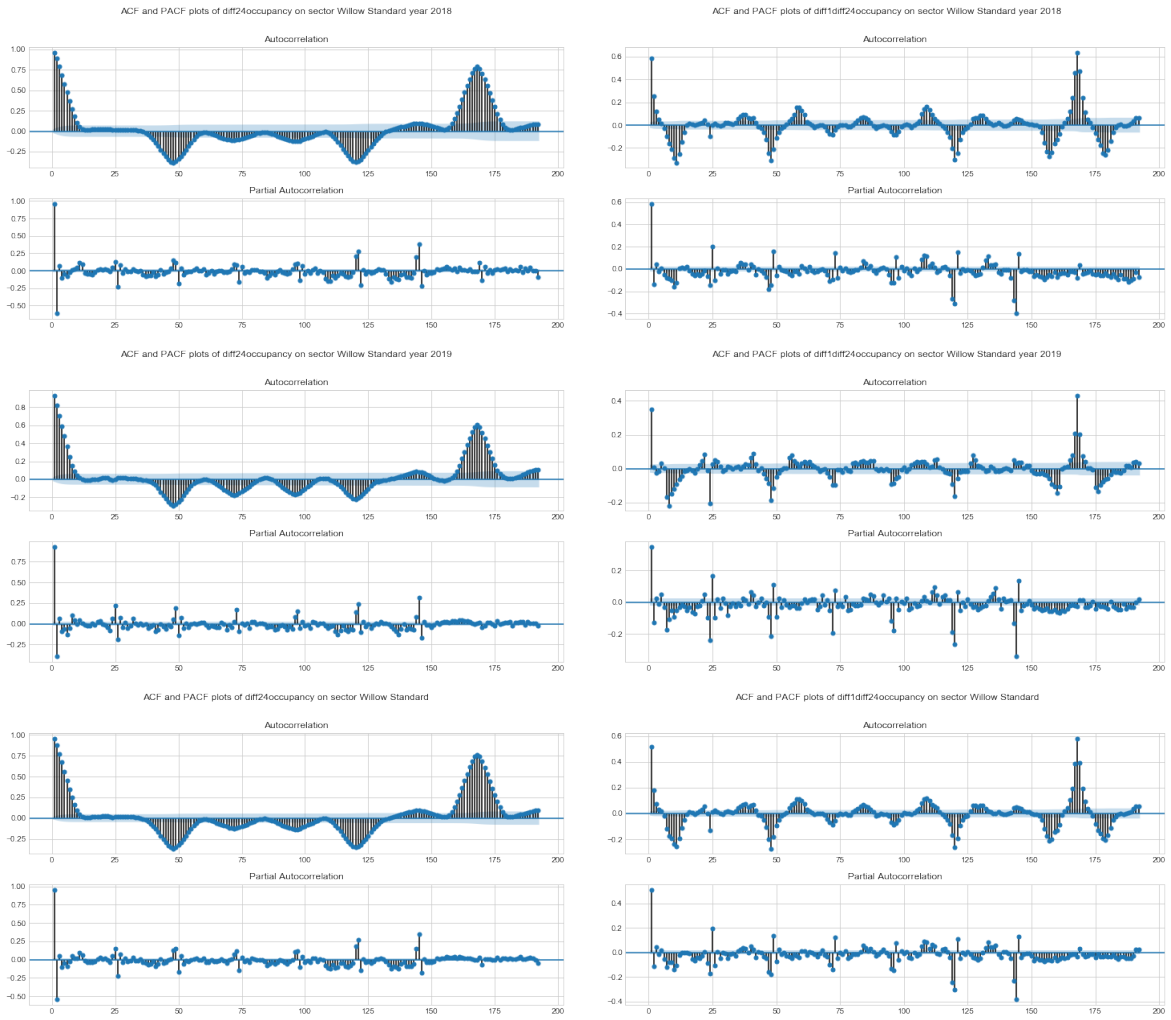


Figure 6.34: Los Angeles Willow Standard sector ACF and PACF plots for the seasonal differentiated (left) and one regular plus one seasonal differentiated series (right). Top is for year 2018, middle for year 2019 and bottom is computed using all the available data.

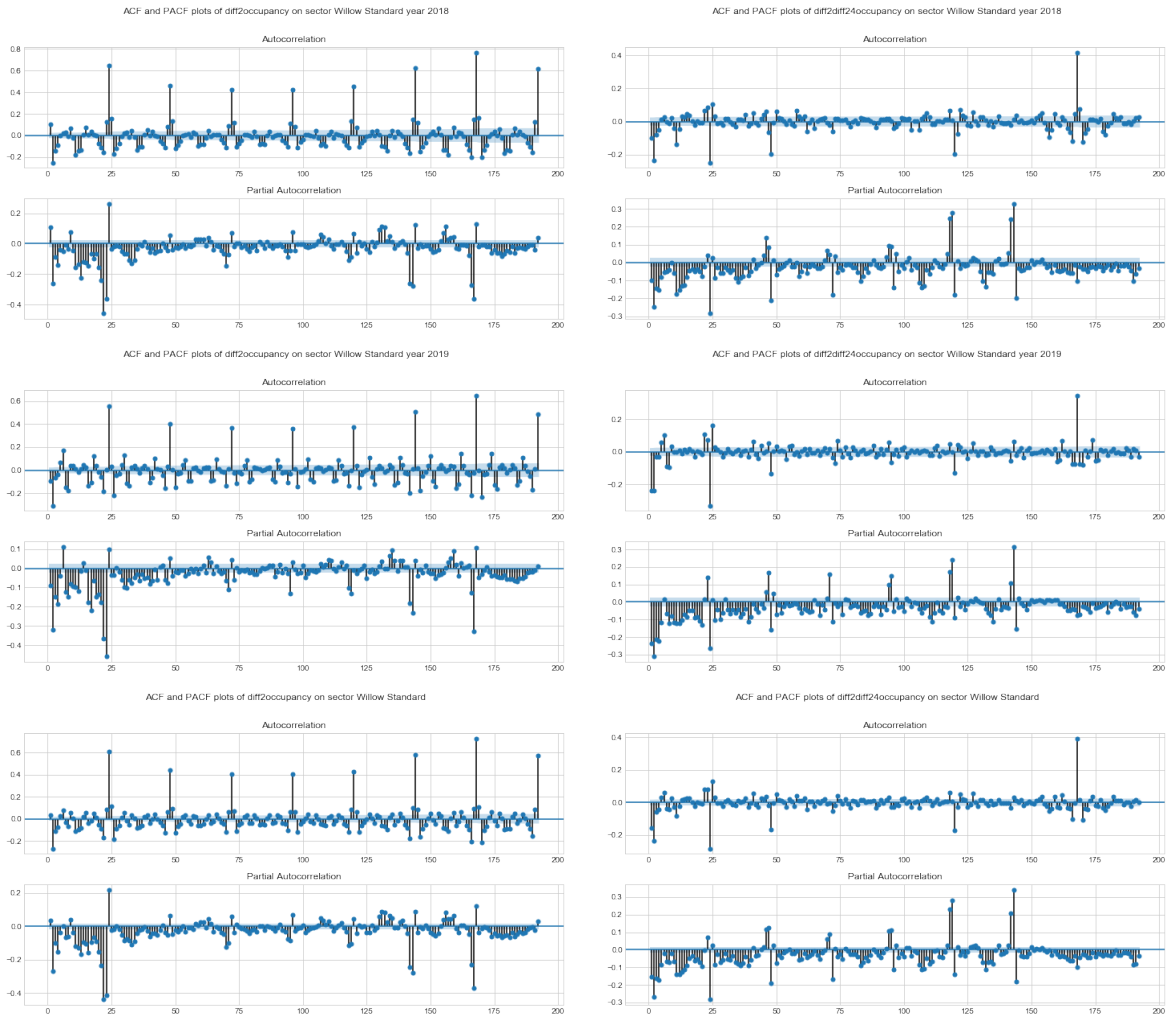


Figure 6.35: ACF and PACF plots of the Los Angeles Willow Standard sector occupancy time series under two regular differentiations (left) and two regular plus one seasonal differentiation (right). From top to bottom, the visualizations use data from different time intervals: 2018, 2019 and both.

with $\sigma^2 = 0.0015$. Lower values in parenthesis correspond to the estimates of the standard errors for each parameter. w_{t-n} and x_{t-n} represent, respectively, the white noise values and the past series values for a number of n lags considered.

The simplified version of the model is presented in Equation 6.19 and achieves -45502.861 AIC and -45465.731 BIC, slightly better than the previous model.

$$\hat{x_t} = 0.3812_{(0.008)}x_{t-1} + 0.2634_{(0.009)}w_{t-1} - 0.6271_{(0.005)}w_{t-24} - 0.3124_{(0.005)}w_{t-48} + w_t, \tag{6.19}$$

with $\sigma^2 = 0.0015$.

Figure 6.36 presents visualizations of the residuals for the proposed model. Similarly to previous cities and sectors, the visualizations show that the model is not able to capture all the patterns in the data due to complex seasonalities that violate the assumptions of the ARIMA models. The density plot of the residuals presents a skewed and leptokurtic distribution with a large amount of values in the left tail. This fact, which is also noticeable in the Q-Q plot, is caused by the change in the occupancy levels throughout the year 2019, when street works started in the zone. However, the occupancy patterns behaved the same after the street works ended, so the same ARIMA model is still useful. The Ljung-Box test and the ACF-PACF plot show that the model captures patterns for the first lags but is not capable of dealing with the multiple seasonalities of the data.

In the ARIMA literature, there exist techniques to correct temporal changes in the levels of a time series, namely outlier detection and intervention analysis. This area of research is focused on detecting, classifying and modeling different outliers with ARIMA, which can have a negative effect on the model. Outliers are classified as:

- Additive outlier: Represents an isolated spike (event).
- Level shift: Represents an abrupt change in the mean level of the series, which can be seasonal (seasonal level shift).
- Transient change: Represents a spike (event) that fades slowly in time.

In the case of the Los Angeles sectors, it can be considered that the effects of the street works on the occupancy levels of the sectors comes from a level shift outlier, as the patterns remain at lower values. Another way to overcome this problem is through the use of exogenous variables (SARIMAX model). A dummy variable that indicates street works could mitigate their effects on the model. This is out of the scope of this study, and the treatment of outliers in ARIMA modeling should be further investigated.

The model has been tested under three different scenarios characterized by different training intervals.

1. The proposed model is trained with data from April 2018 until October 2018 and tested on November and December 2018 data.

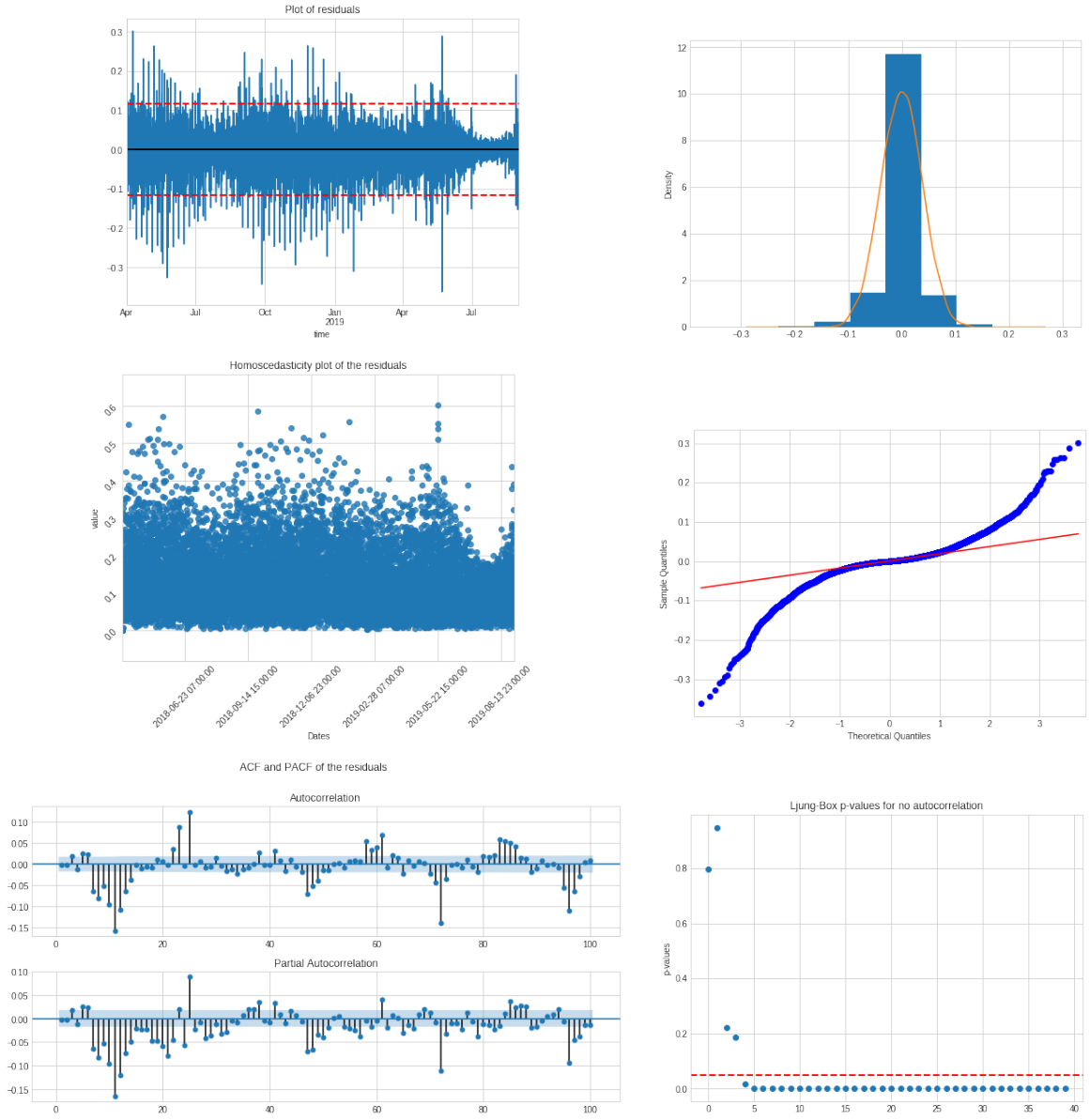


Figure 6.36: Los Angeles Willow Standard sector visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

Train date interval	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
2018	0.0447	0.0878	0.1252	0.1581	0.1873	0.2131
2019	0.0382	0.0642	0.0808	0.0928	0.1020	0.1099
All	0.0396	0.0672	0.0849	0.0971	0.1064	0.1140

Table 6.20: Los Angeles Standard sector ARIMA RMSE.

2. The proposed model is trained with data from January 2019 until August 2019 and tested on data from September 2019 until the end of October 2019.
3. The proposed model is trained with data that comprises the time interval from April 2018 to August 2019. The time interval for the test data is from September 2019 to the end of October 2019.

A multi-step strategy is used to study the effect of the forecasts at different time horizons. Table 6.20 presents the RMSE of the model at each of the considered forecasting hours for each scenario, from one to six.

Figure 6.37 shows the one-step forecasts of the ARIMA models proposed for the Standard sector at the different time intervals considered. The visualizations for the 2018 scenario indicate that the one-step forecasts are accurate and coincide with the values of the real occupancy, but they fail at the weekends because the model is not able to adapt to the change in pattern. In the 2019 scenario, the model achieves lower RMSE than when trained and tested with 2018 data, and it is slightly lower than that of the model trained with all the data, although there is not a huge difference. In the all-data scenario, the effects of the occupancy level changes between 2018 and 2019 are visible, as the forecasts of the models are way higher or lower than the true occupancy. This is because the parameters are estimated by taking into account the 2018 and 2019 occupancy levels. Despite using data with changes in the mean and variance, the model provides low RMSE values. This indicates that, if the changes in the mean and variance are expected to be temporary, training the model with changing patterns in the mean and variance levels is a viable strategy when compared to training the model with only the most recent data.

6.7.2 Permit sector

This section includes the experiment corresponding to the Permit sector in Los Angeles, namely *ARIMA_ LA_permit*.

Daily box plots for the Permit sector in Los Angeles for the first two weeks of 2018 are presented in Figure 6.38. The non-differentiated series presents two distinct behaviours, one for the weekdays and another for the weekends. With one regular differentiation, the series becomes more uniform with less variance and a mean of zero, similarly to other studied sectors. One seasonal differentiation does not achieve constant mean or constant variance. The series after one regular plus one seasonal differentiation performs similarly to the series with only one regular differentiation, but with some more variance.

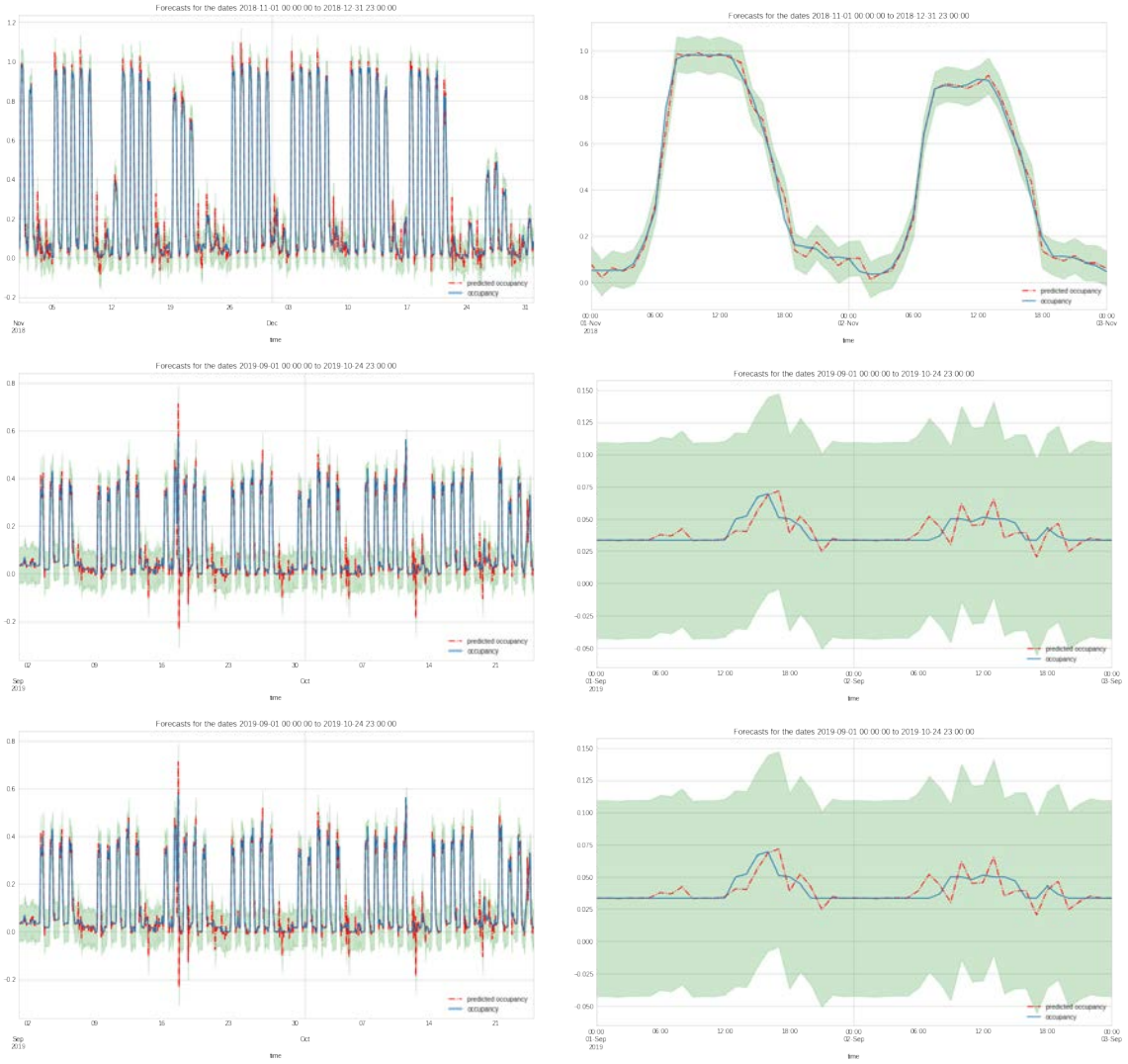


Figure 6.37: ARIMA forecasts from the model proposed for the Standard sector in Los Angeles. Top visualizations belong to the 2018 scenario, middle visualizations to the 2019 scenario and the bottom visualizations to the all-data scenario. Left presents forecasts for the entire out-of-sample interval while right is the first two days of the same interval of time. Blue denotes the observed occupancy, red that forecasted by the model and green is the confidence interval.

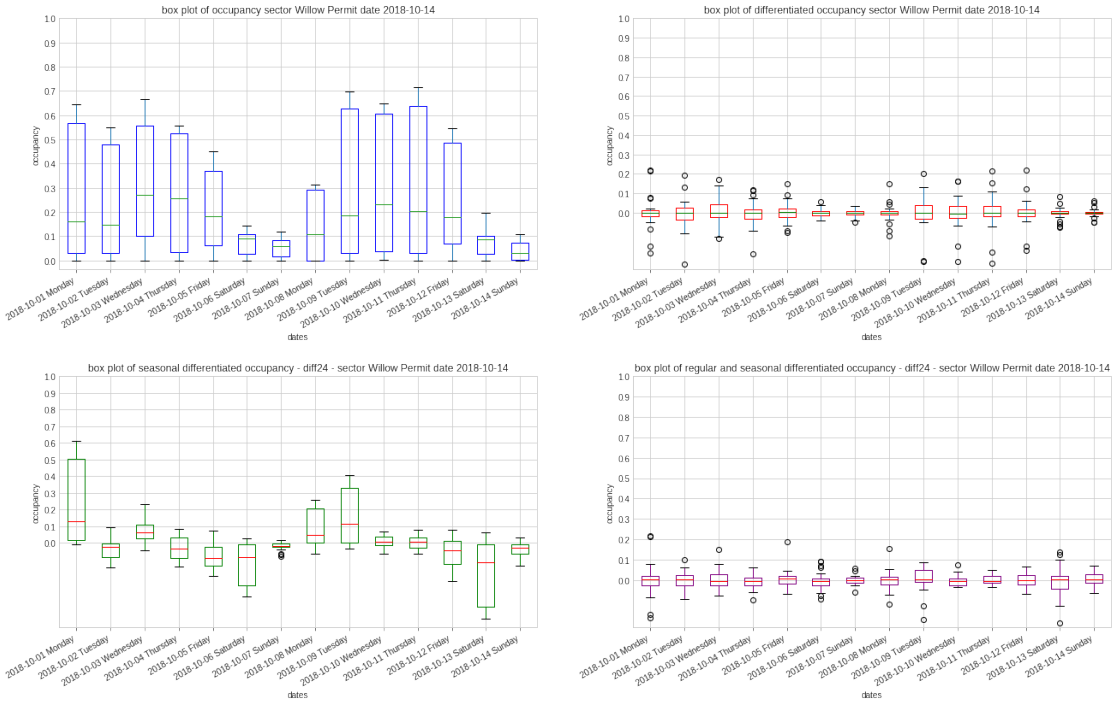


Figure 6.38: Daily box plots of the Los Angeles Permit sector from the first two weeks of October 2018 under different numbers of differentiations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

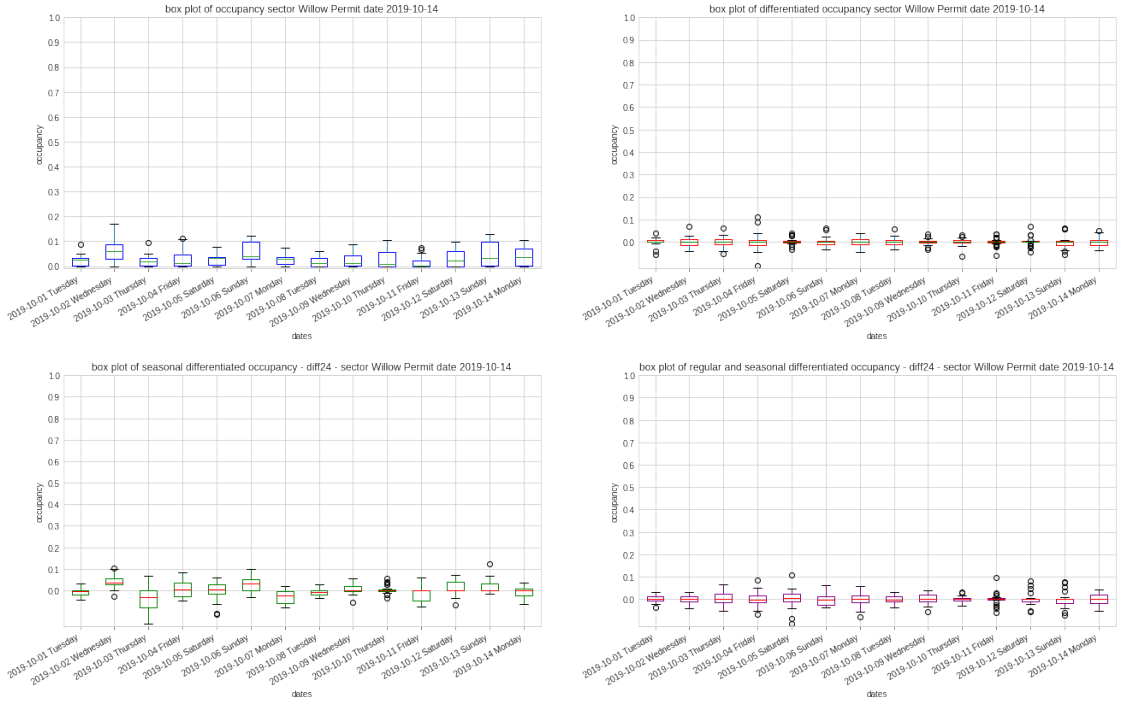


Figure 6.39: Daily box plots of the Los Angeles Permit sector from the first two weeks of October 2019 under different numbers of differentiations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

Similarly to what happened in the Standard sector in the year 2019 (see Figure 6.31), street works in the zone affected the levels of occupancy of the sector. Daily box plots for the first two weeks of October 2019 are presented in Figure 6.39. In the non-differentiated series, the levels of occupancy are pushed down, thus presenting changes in their values. It is not clear at all if the patterns from 2018 are still present.The one regular differentiated series and the one regular plus one seasonal seem to have constant mean and low levels of variance. The seasonal differentiated series shows no constant mean.

From the conclusions extracted in the Standard sector, all the available data are taken into account in order to establish the orders of the ARIMA models, with the expectation that the patterns for the year 2018 continue to be valid when the street works end.

Table 6.21 presents the mean and variance for the entire interval of data available for the Permit sector. Similarly to the Standard sector, the lowest levels of variance are accomplished when using one regular plus one seasonal differentiation and with two regular differentiations.

Assuming, after the street works of 2019 are completed, that the parking occupancy levels are the same as those of the year 2018, Figure 6.40 presents the ACF and PACF plots when using the entire interval of data available. The non-differentiated series suggests AR(2) and SAR(1) models. The one regular differentiation series presents a large number

Difference order	Mean	Variance
No differentiation	0.1271	0.0364
One regular	0	0.0028
Two regular	0	0.0021
One seasonal	0	0.0163
Two seasonal	-0.0005	0.0322
Regular plus seasonal	0	0.0018
Two regular, one seasonal	0	0.0025
One regular, two seasonal	0	0.0045
Two regular, two seasonal	0	0.0072

Table 6.21: Los Angeles Willow Permit sector mean and variance values when using different orders of differentiation.

Model	Log likelihood	AIC	BIC
$SARIMA(2,0,0)(1,0,0)_{24}$	23726.514	-47443.028	-47405.888
$SARIMA(1,1,2)(1,0,0)_{24}$	23341.401	-46672.802	-46635.662
$SARIMA(1,1,1)(0,1,1)_{24}$	24570.411	-49132.822	-49103.118

Table 6.22: Scores from the candidate ARIMA models for the Permit sector. The higher the log likelihood and lower the AIC/BIC, the better.

of significant correlations, which are perhaps caused by complex seasonalities; but AR(1), MA(2) and SAR(1) can be considered. In the one seasonal differentiation series, it is harder to suggest the order of an ARIMA model, as there are a large number of very significant correlations. The one regular plus one seasonal differentiation suggests AR(1), MA(1) and SMA(1). Finally, the two regular differentiation series present a large number of significant correlations for multiple seasonalities.

Table 6.22 presents the scores of the candidate models for the Los Angeles Permit sector. The proposed model is the $SARIMA(1,1,1)(0,1,1)_{24}$, because it achieves the highest scores.

Equation 6.20 presents the mathematical form of the model proposed $SARIMA(1,1,1)(0,1,1)_{24}$. Using the t-test, the MA parameter is not significant, so it is possible to simplify the model by removing it.

$$\hat{x_t} = 0.4559_{(0.018)}x_{t-1} - 0.0336_{(0.020)}w_{t-1} - 0.9482_{0.002}w_{t-24} + w_t, \tag{6.20}$$

with $\sigma^2 = 0.0011$. The lower values in parenthesis correspond to the estimates of the standard errors for each parameter. w_{t-n} and x_{t-n} represent, respectively, the white noise values and the past series values for a number of n lags considered.

The model without the MA parameter in the regular part is shown in Equation 6.21. The model achieves an AIC of -49132.498 and a BIC value of -49110.220, which is similar

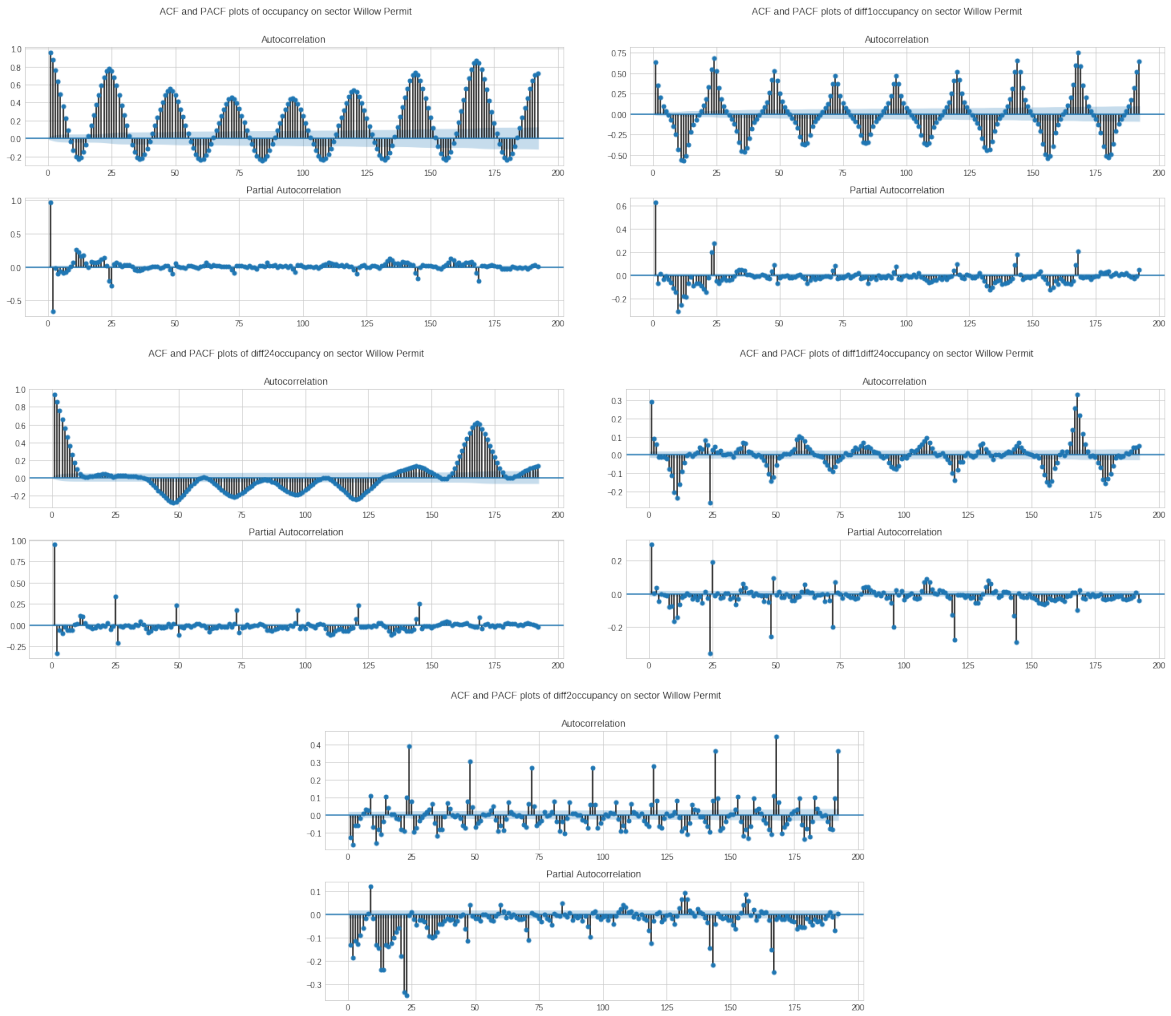


Figure 6.40: Los Angeles Willow Permit sector ACF and PACF plots for all the available data under different differentiation orders. From top left to bottom: non-differentiated, one regular differentiation, one seasonal differentiation, one regular plus one seasonal differentiation and two regular differentiations.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
0.0179	0.0259	0.0294	0.0322	0.0334	0.0344

Table 6.23: Los Angeles Permit sector ARIMA RMSE

to the previous model.

$$\hat{x_t} = 0.4285_{(0.005)}x_{t-1} - 0.9481_{(0.002)}w_{t-24} + w_t, \tag{6.21}$$

with $\sigma^2 = 0.0011$.

Figure 6.41 presents visualizations of the residuals from the model proposed for the Permit sector of Los Angeles. The model is not able to capture extreme values of the series, especially for 2018, but it is more capable of doing so with more recent data. Similarly to previous models, the model cannot handle multiple seasonalities, as can be seen clearly in the ACF/PACF plot, in which the residuals present correlations around lags eleven and twenty-four.

The training data comprises the time interval from April 2018 to August 2019. The time interval for the test data is from September 2019 to the end of October 2019. A multi-step strategy is used to study the effect of the forecasts at different time horizons. Table 6.23 presents the RMSE of the model at each of the considered forecasting hours, from one up to six. Although the models present very low error rate , it must be taken into account that the out-of-sample data has very low variability due to the effects of the street works on the parking occupancy.

Figure 6.42 shows the one-step forecasts for the proposed ARIMA model. In all the out-of-sample visualizations, forecast values are slightly over and under the true values of occupancy. In the two-day plot, it is possible to see that the model over-estimates the peaks in the series by giving higher or lower values of occupancy in regard to the changes. Lastly, note that the model forecasts are falling behind, as their values are a replication of the most recent input.

6.7.3 Disabled sector

This section includes the experiment corresponding to the Disabled sector in Los Angeles, namely *ARIMA_ LA_disabled*.

In Chapter 3.5.4, Section 3.65 showed that the Disabled sector has similar patterns to the previous sectors in 2018, but in 2019 it was affected by the street works in the zone, which greatly reduced the occupancy values. Figure 6.43 presents the daily box plots of the Disabled sector in Los Angeles for the first two weeks of October 2018 under different numbers of differentiations. The first thing to notice is how the occupancy levels are different between weekdays and the weekend. The one regular differentiated

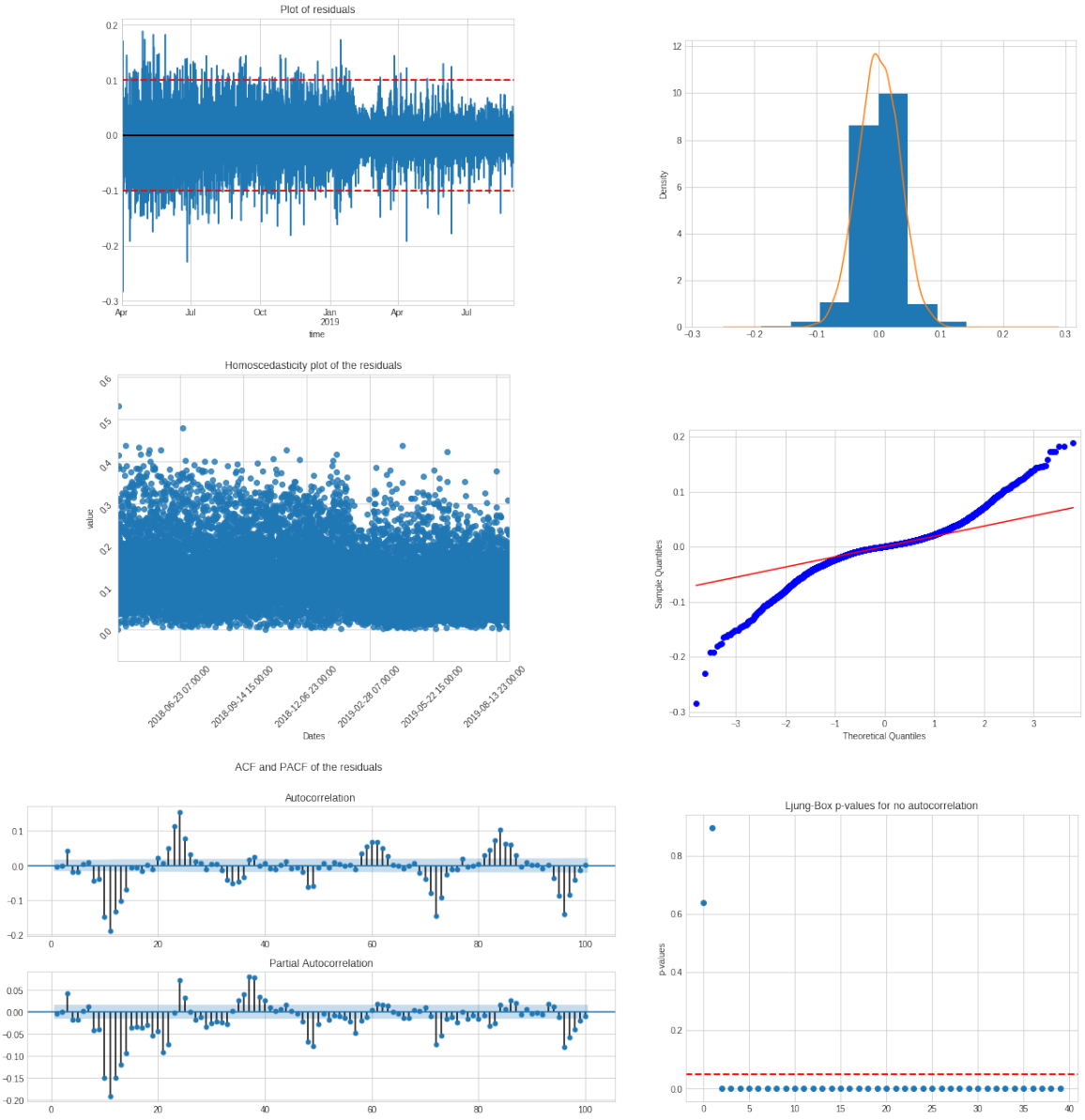


Figure 6.41: Los Angeles Willow Permit sector visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

Figure 6.42: ARIMA forecasts from the proposed model for sector Permit from Los Angeles. Left presents forecasts for all the out-of-sample interval while right the first two days of the same interval of time. Blue denotes the observed occupancy, red the forecasted by the model and green is the confidence interval.

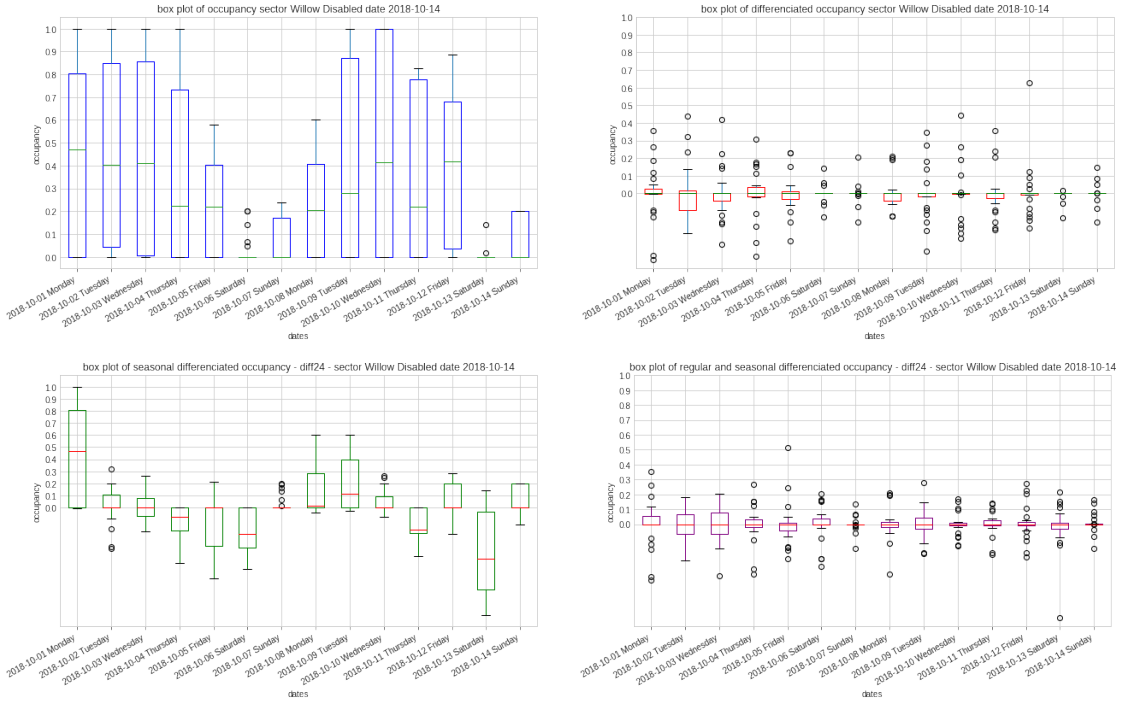


Figure 6.43: Daily box plots of the Los Angeles Disabled sector from the first two weeks of October 2018 under different numbers of differentiations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

series presents a constant mean around zero and a reduced variance. The series with one seasonal differentiation has no constant variance and no constant mean. Lastly, the series with one regular plus one seasonal differentiation presents constant mean and a more or less constant variance.

Figure 6.44 presents daily box plots for the occupancy levels of the Disabled sector in October 2019. The patterns presented are similar to those of the year 2018 but the levels of occupancy have been reduced drastically. It can be assumed that, similarly to the Standard sector (Figure 6.32), the levels of occupancy are similar to those of 2018, as can be seen in the plot for the entire time interval in Figure 3.58.

Table 6.24 presents the mean and variance for the years 2018, 2019 and the entire interval of data available for the Disabled sector. As shown in the daily box plots, the levels of the mean and variance are lower in 2019 compared to 2018. When considering the entire time interval, the mean and variance levels are between 2018 and 2019 values. When considering the entire interval of data, the series with lower variance is the one with one regular differentiation and the one with one regular plus one seasonal differentiation.

From the visualizations on the Standard sector (see Figure 6.33) and the time plot with all the data (Figure 3.58), it is assumed that the patterns of occupancy are similar to those of 2018 once the street works end. Because of this, the ACF and PACF are

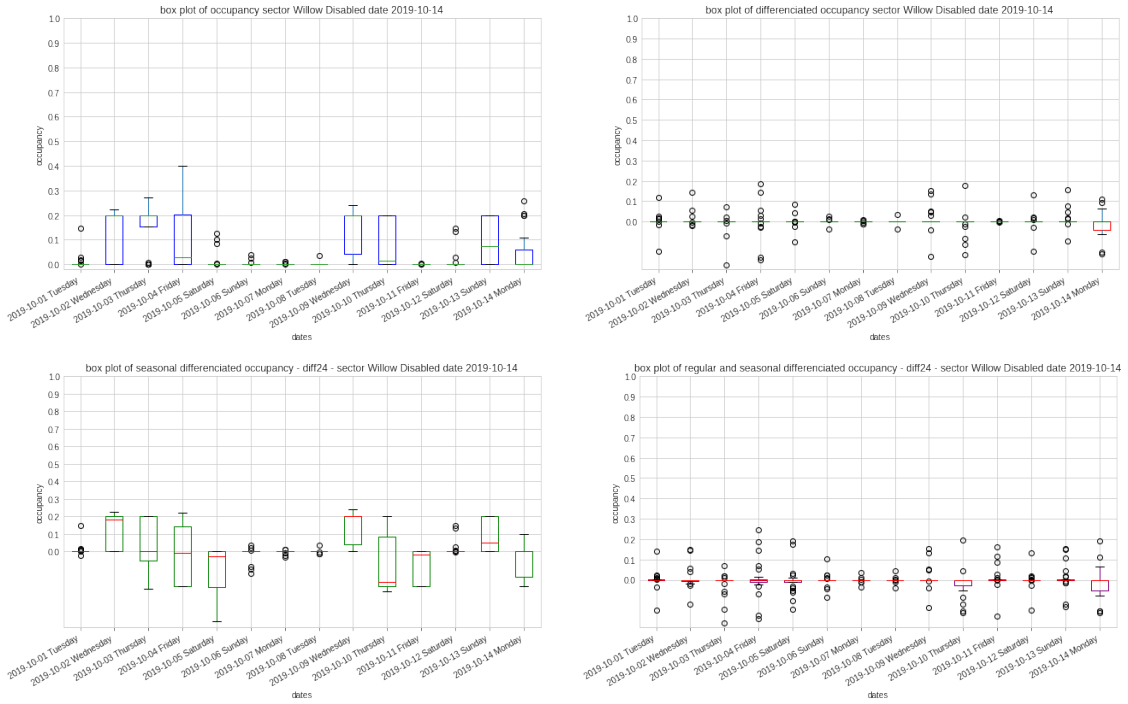


Figure 6.44: Daily box plots of the Los Angeles Disabled sector from the first two weeks of October 2019 under different numbers of differentiation operations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

Difference order	2018-Mean	2018-Variance	2019-Mean	2019-Variance	All-Mean	All-Variance
No differentiation	0.2804	0.1229	0.0473	0.0196	0.1594	0.0828
One regular	0	0.0149	0	0.0023	0	0.0083
Two regular	0	0.0167	0	0.0033	0	0.0097
One seasonal	0.0003	0.0868	-0.0001	0.013	0.0001	0.0484
Two seasonal	-0.001	0.2065	0.0002	0.0312	-0.0005	0.1152
Regular plus seasonal	0	0.0149	0	0.0029	0	0.0086
Two regular, one seasonal	0	0.0246	0	0.0052	0	0.0145
One regular, two seasonal	0	0.0412	0	0.0081	0	0.024
Two regular, two seasonal	0	0.0712	0	0.015	0	0.042

Table 6.24: Los Angeles Willow Disabled sector mean and variance values when using different orders of differentiation for years 2018, 2019 and for the entire interval of data available.

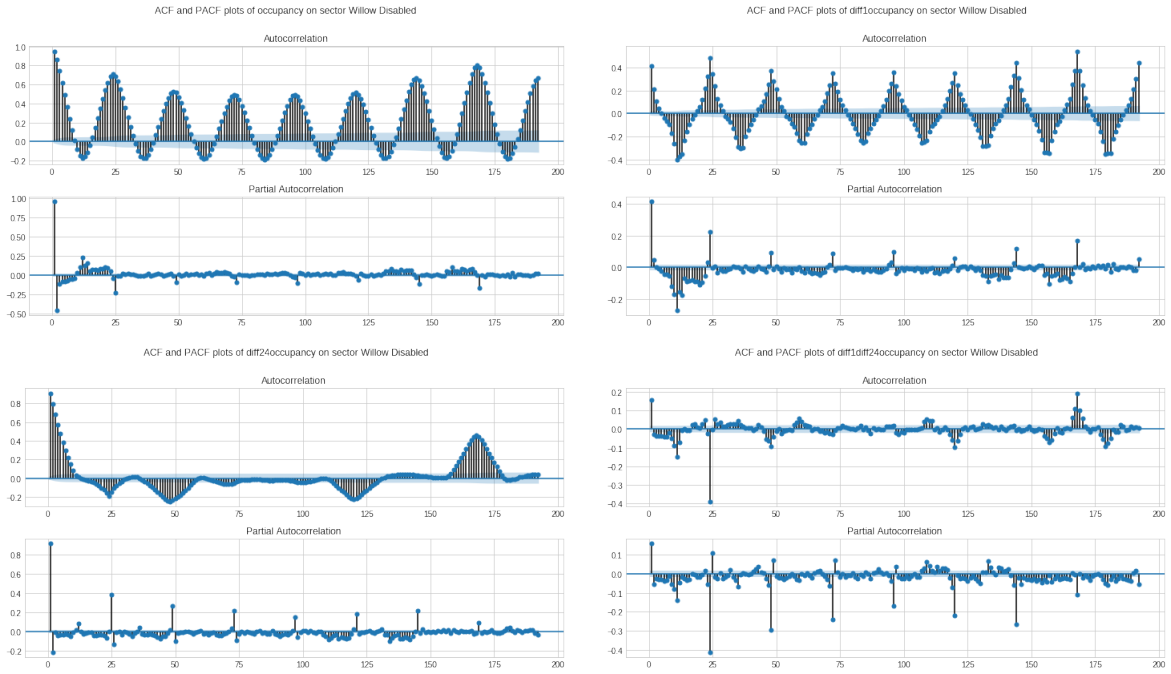


Figure 6.45: Los Angeles Willow Disabled sector ACF and PACF plots for all the available data under different differentiation orders. From top left to bottom right: non-differentiated, one regular differentiation, one seasonal differentiation and one regular plus one seasonal differentiation.

Model	Log likelihood	AIC	BIC
$ARIMA(2,0,0)$	13225.682	-26443.365	-26413.652
$SARIMA(0,1,1)(0,1,1)_{24}$	14877.417	-29748.834	-29726.556

Table 6.25: Scores from the candidate ARIMA models for the Disabled sector in Los Angeles. The higher the log likelihood and lower the AIC/BIC, the better.

computed using the entire interval of data available in order to suggests ARIMA models for the time series.

From the ACF and PACF plots in Figure 6.45, it is possible to suggest the following ARIMA models. The non-differentiated series suggests an AR(2). The one regular differentiation and the one seasonal differentiation series present a large number of correlated lags, so it is difficult to suggest any model. Finally, the one regular plus one seasonal differentiation suggests MA(1) and SMA(1).

Table 6.25 presents the scores of the candidate models for the Los Angeles Disabled sector. Clearly $SARIMA(0,1,1)(0,1,1)_{24}$ has higher values and is the model chosen to be proposed for the series.

Equation 6.22 presents the mathematical form of the proposed model

Model	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
$SARIMA(0,1,1)(0,1,1)_{24}$	0.0387	0.0594	0.0734	0.0827	0.0893	0.0929

Table 6.26: Los Angeles Disabled sector ARIMA RMSE

$SARIMA(0,1,1)(0,1,1)_{24}$. All parameters are significant under the t-test hypothesis.

$$\hat{x_t} = 0.2173_{(0.006)} w_{t-1} - 0.9518_{(0.002)} w_{t-24} + w_t, \tag{6.22}$$

with $\sigma^2 = 0.0053$. Lower values in parenthesis correspond to the estimates of the standard errors for each parameter. w_{t-n} represents the white noise values for a number of n lags considered.

From different visualizations of the residuals (Figure 6.46), it can be seen clearly that the residuals do not follow a normal distribution. Furthermore, the distribution has a leptokurtic shape (heavy tails), caused in part by the change in levels of occupancy due to external factors. Another fact that affects the model is that it is not capable of handling multiple seasonalities, as can be seen in the ACF and PACF plots of residuals.

After the residual analysis, it is tempting to try a high order model like $SARIMA(12,1,12)(3,1,1)_{24}$. However, such a model provides an AIC of -30980.603 and a BIC of -30765.248, which is little improvement at the cost of adding a high number of parameters.

Figure 6.47presents the residuals for the experimental model $SARIMA(12,1,12)(3,1,1)_{24}$. Despite the AIC and BIC scores, the model is able to capture patterns at the first lags that the previous model could not. This could indicate that methods which naturally involve a high number of parameters (like Neural Networks) could easily manage multiple seasonalities.

Table 6.26 presents the RMSE of the proposed model $SARIMA(0,1,1)(0,1,1)_{24}$. As in the other Los Angeles sectors, the training data comprises the time interval from April 2018 to August 2019. The time interval for the out-of-sample data is from September 2019 until the end of October 2019. A multi-step strategy is used to study the effect of the forecasts at different time horizons.

Figure 6.48 shows the one-step forecasts for the ARIMA models proposed for the Disabled sector in the Los Angeles Willow sector. Although the model provides acceptable one-step-ahead forecasts, the plots show that the forecasts appear to replicate the most recent input. Also notice that sometimes the plot provides forecasts with negative values.

6.7.4 ZipCar sector

This section includes the experiment corresponding to the ZipCar sector in Los Angeles, namely *ARIMA_ LA_zipcar*.

In Chapter 3.5.4, Section 3.66 showed that the ZipCar sector was not affected by the street works of 2019. Figure 6.49 presents the daily box plots of the ZipCar sector in

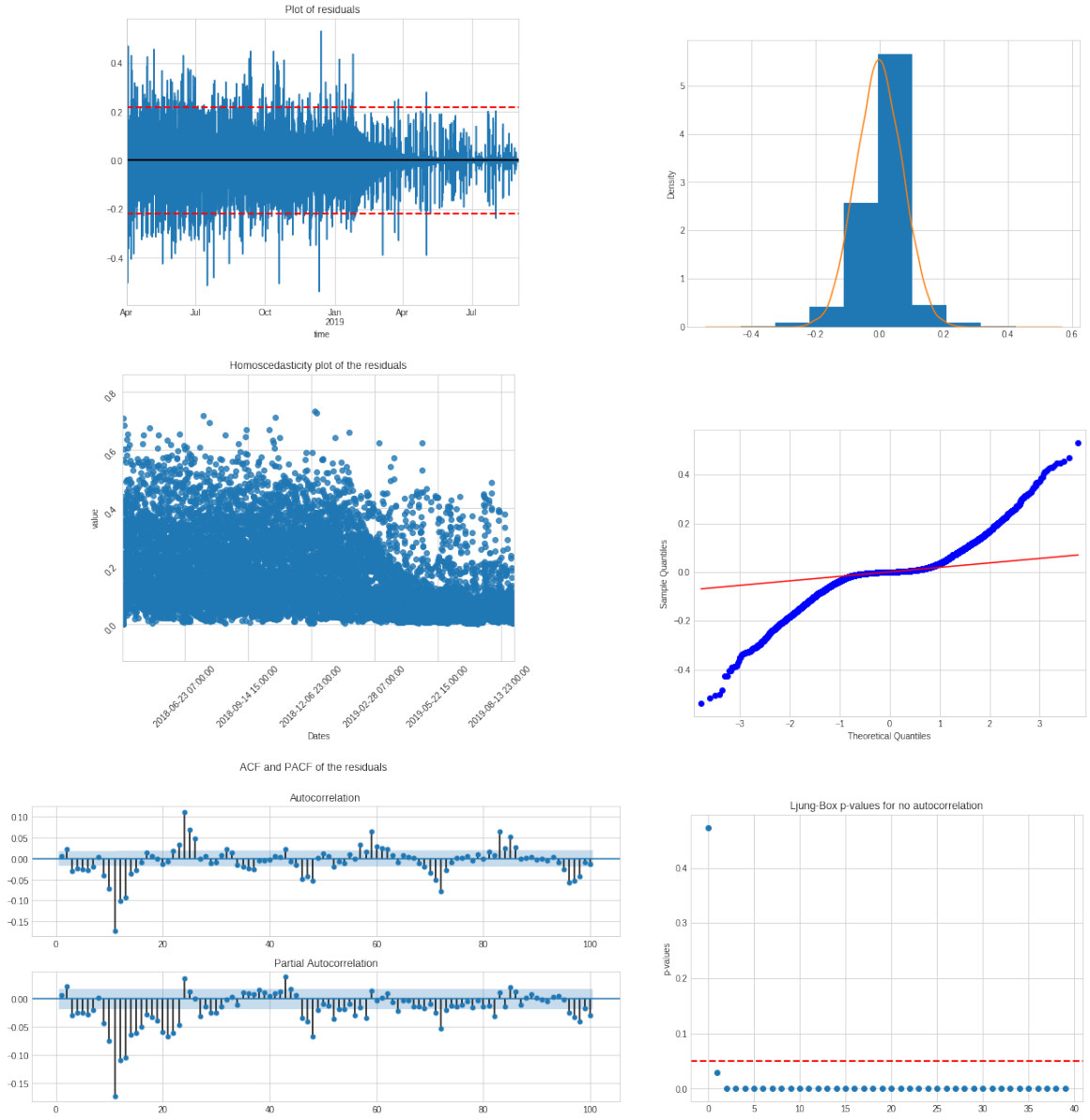


Figure 6.46: Los Angeles Willow Disabled sector visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

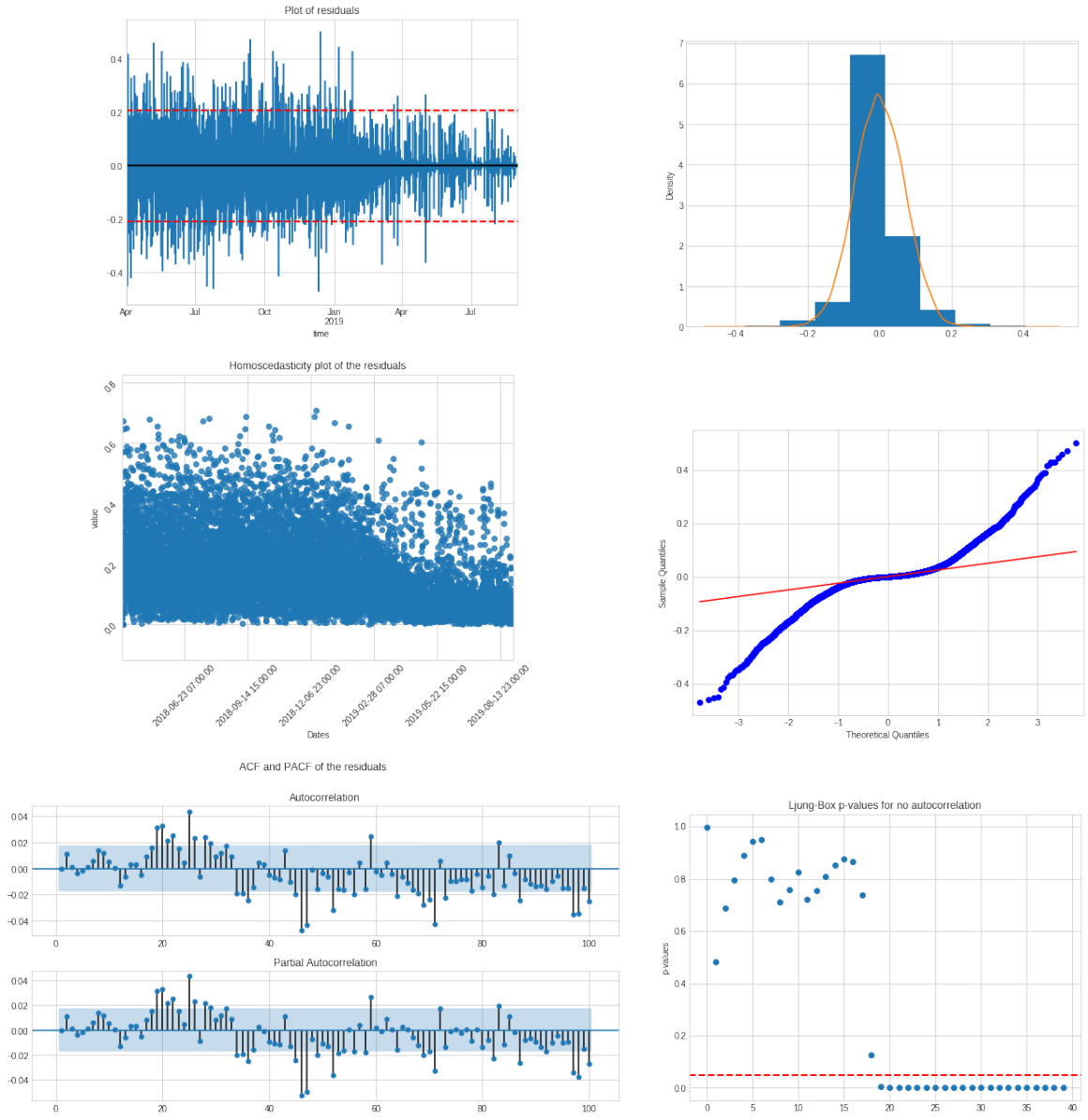


Figure 6.47: Los Angeles Willow Disabled sector visualizations and residual tests of the experimental model with a large number of parameters. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

Figure 6.48: ARIMA forecasts from the model proposed for the Disabled sector in Los Angeles. Top presents forecasts for the entire out-of-sample interval while bottom is the first two days of the same interval of time. Blue denotes the observed occupancy, red the occupancy forecasted by the model and green is the confidence interval.

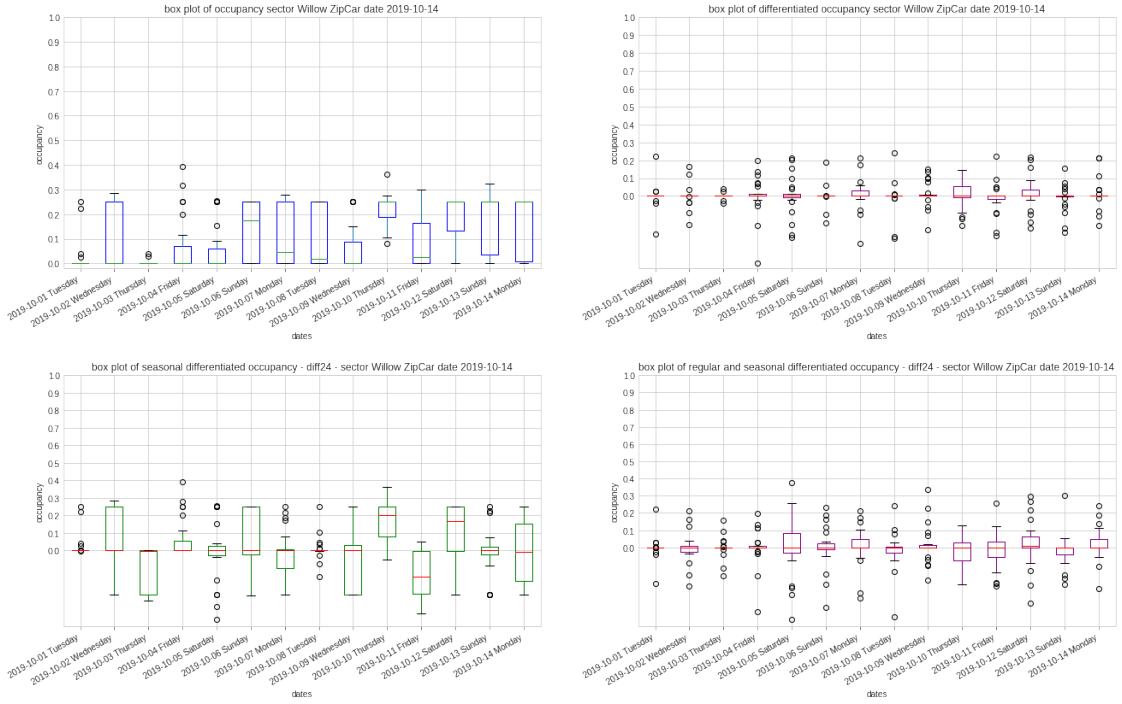


Figure 6.49: Daily box plots of the Los Angeles ZipCar sector from the first two weeks of October 2019 under different numbers of differentiations: no differentiation (top left), one regular differentiation (top right), one seasonal differentiation (bottom left) and one regular plus one seasonal differentiation (bottom right).

Los Angeles under different numbers of differentiations. Similarly to other sectors with a low number of sensors, the non-differentiated series shows chaotic behaviour. When the series are differentiated, the mean becomes zero and constant, and the variance is reduced and seemingly constant. When only applying a seasonal differentiation, the mean does not become constant and the same happens with the variance. In the time series with a regular plus seasonal differentiation, the mean seems to be constant with a value of zero, but the variance is not.

Table 6.27 presents the mean and variance for the entire interval of data available for the ZipCar sector. The lowest levels of variance are achieved with one regular differentiation and one regular plus one seasonal differentiation.

Figure 6.50 presents the ACF and PACF plots when using all the available data. The non-differentiated series suggests and AR(1). The one regular differentiation series presents a large number of correlated lags, so, it is difficult to suggest any model. In the case of one seasonal differentiation, similar to the one regular differentiation series, it is difficult to suggest any model. Finally, the one regular plus one seasonal differentiation series suggest an SMA(1).

Table 6.28 presents the scores of the candidate models for the Los Angeles ZipCar sector. Both candidate models have similar scores, but $SARIMA(0,1,0)(0,1,1)_{24}$ has slightly

Difference order	Mean	Variance
No differentiation	0.1627	0.0295
One regular	0	0.005
Two regular	0	0.0091
One seasonal	0.0004	0.026
Two seasonal	0	0.0729
Regular plus seasonal	0	0.0088
Two regular, one seasonal	0	0.017
One regular, two seasonal	0	0.0265
Two regular, two seasonal	0	0.0517

Table 6.27: Los Angeles Willow ZipCar sector mean and variance values when using different orders of differentiation.

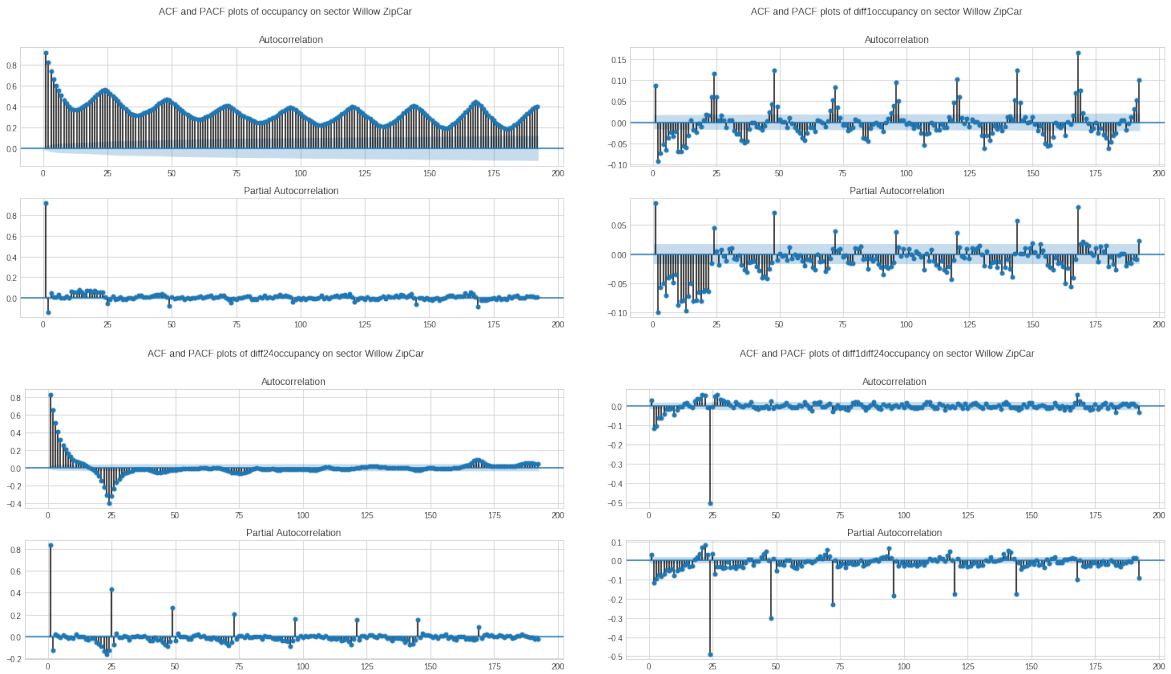


Figure 6.50: Los Angeles Willow ZipCar sector ACF and PACF plots for all the available data under different differentiation orders. From top left to bottom: non-differentiated, one regular differentiation, one seasonal differentiation and one regular plus one seasonal differentiation.

Model	Log likelihood	AIC	BIC
$ARIMA(1,0,0)$	15485.150	-30964.301	-30942.016
$SARIMA(0,1,0)(0,1,1)_{24}$	15626.366	-31248.731	-31233.879

Table 6.28: Scores from the candidate ARIMA models for the ZipCar sector in Los Angeles. The higher the log likelihood and lower the AIC/BIC, the better.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
0.0643	0.0913	0.1056	0.1139	0.1212	0.1261

Table 6.29: Los Angeles ZipCar sector ARIMA RMSE

higher values and is the model chosen to be proposed.

Equation 6.23 is the mathematical form of the proposed model $SARIMA(0,1,0)(0,1,1)_{24}$. All parameters are significant under the t-test hypothesis.

$$\hat{x_t} = -0.9494_{(0.002)} w_{t-1} + w_t, \tag{6.23}$$

with $\sigma^2 = 0.0047$. Lower values in parenthesis correspond to the estimates of the standard errors for each parameter. w_{t-n} represents the white noise values for a number of n lags considered.

From different visualizations of the residuals (Figure 6.51), it can be seen clearly how residuals inherit the chaotic behaviour of the time series. This is represented by heavy tails in the distribution, no homoscedasticity in time and a high number of significant correlations in the ACF and PACF plots.

As in the other Los Angeles sectors, the training data comprises the time interval from April 2018 to August 2019. The time interval for the out-of-sample data is from September 2019 to the end of October 2019. A multi-step strategy is used to study the effect of the forecasts at different time horizons. Table 6.29 presents the RMSE of the model at each of the considered forecasting hours, from one up to six. Observe how the RMSE increases faster in the first forecasting steps than in the later ones.

Figure 6.52 shows the one-step forecasts for the ARIMA model proposed for the ZipCar sector in Los Angeles Willow. The model fails clearly when there are no changes in the occupancy for an interval of time, and the predictions give the sensation of displacement or that they are replications of the most recent input.

6.8 Riyadh ARIMA models

From the analysis of data in Section 3.5.5, it was concluded that sectors in the Riyadh city fulfill the following statements:

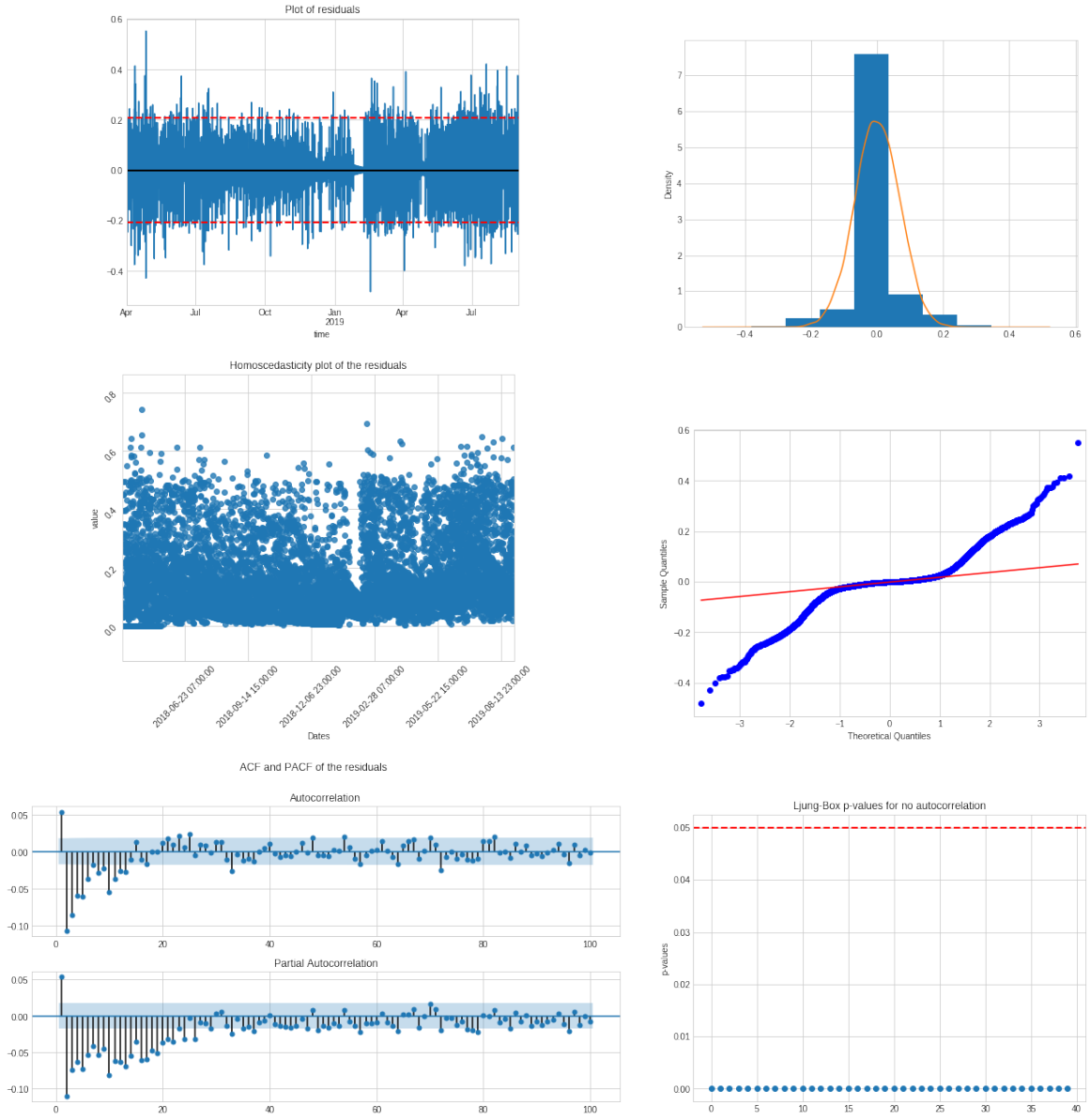


Figure 6.51: Los Angeles Willow ZipCar sector visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

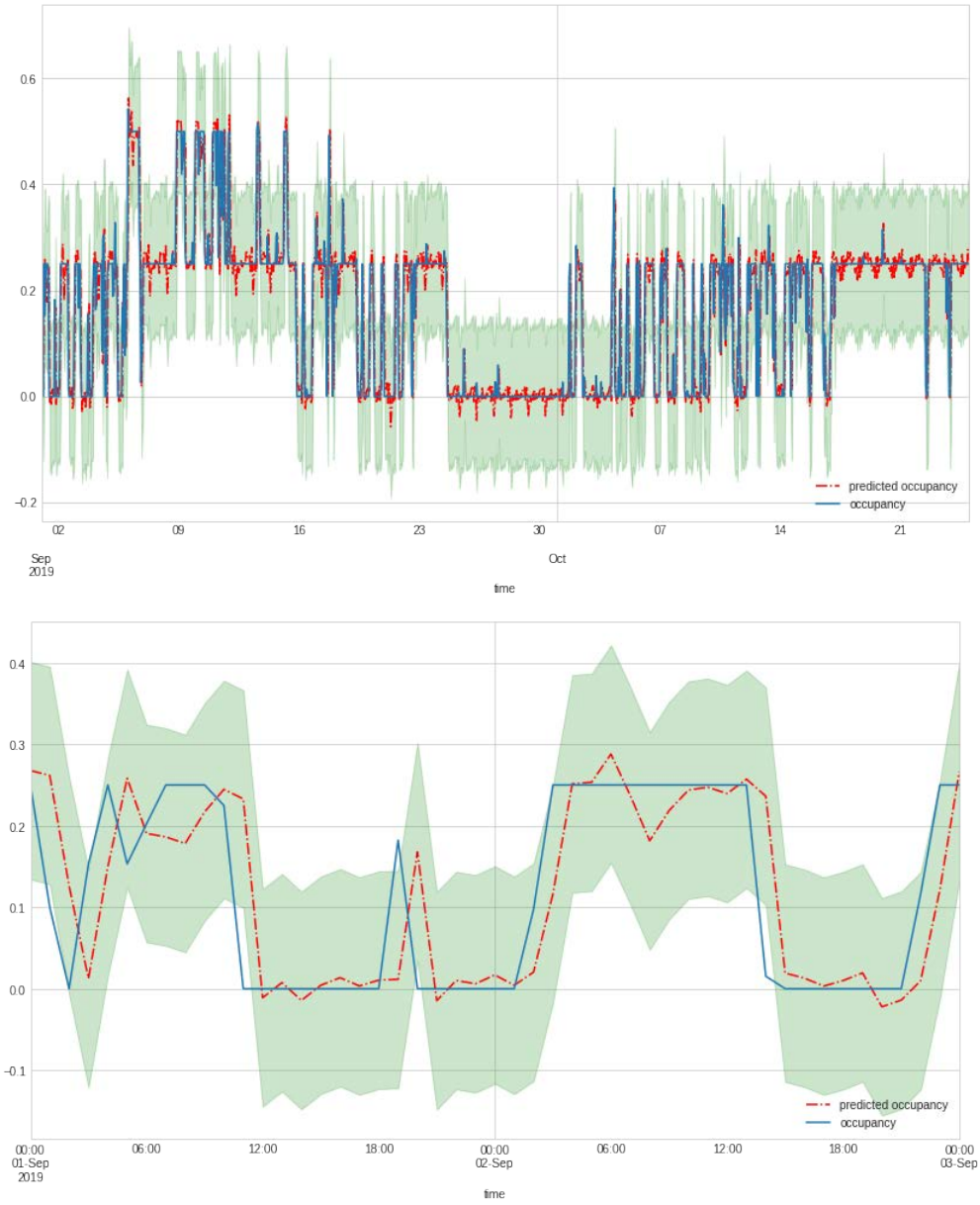


Figure 6.52: ARIMA forecasts from the model proposed for the ZipCar sector in Los Angeles. Top presents forecasts for the entire out-of-sample interval while bottom is the first two days of the same interval of time. Blue denotes the observed occupancy, red the occupancy forecasted by the model and green is the confidence interval.

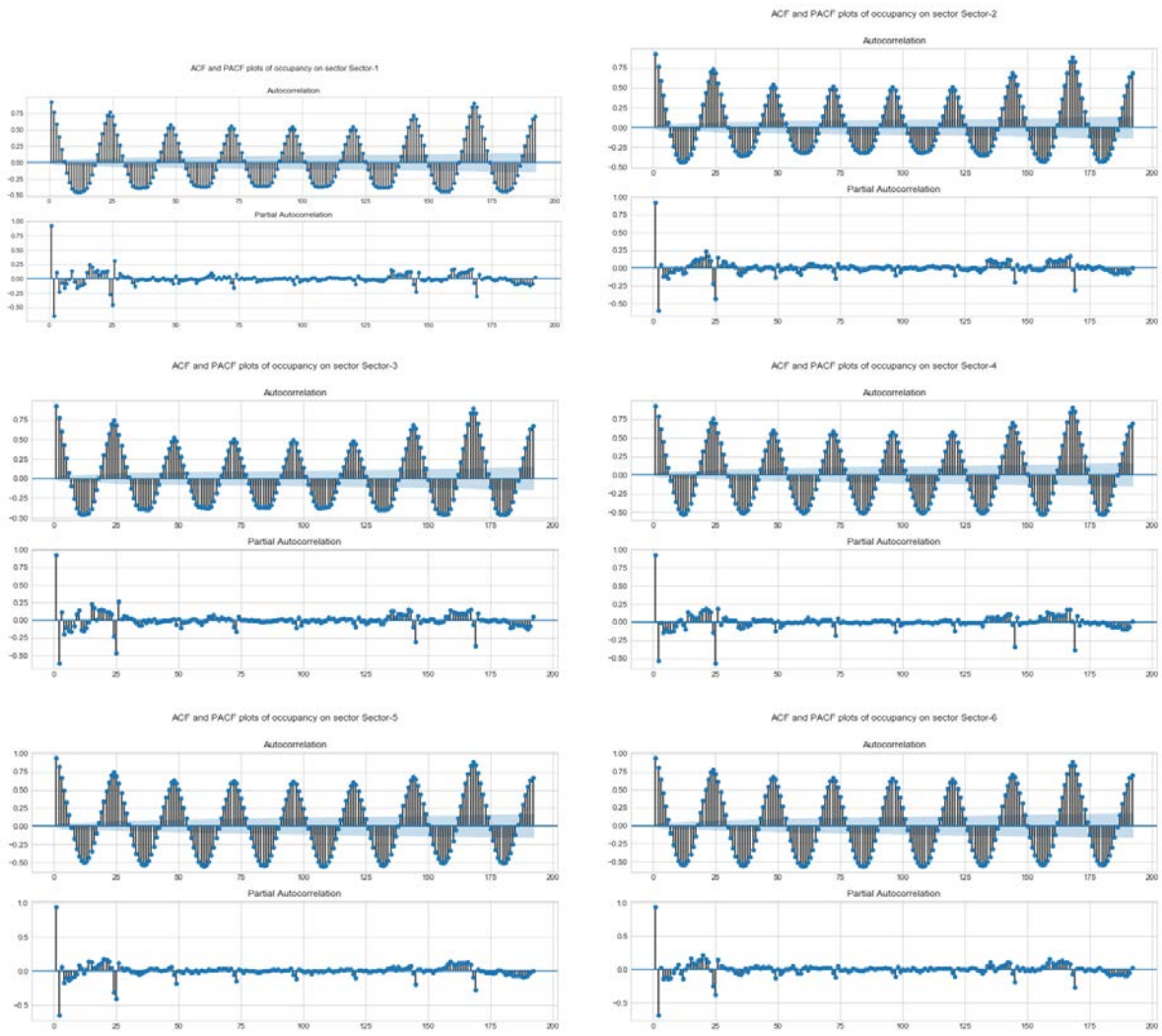


Figure 6.53: ACF and PACF plots of the non-differentiated series from Riyadh. From top left to bottom right: Sector-1 to Sector-6.

- In the nearly two years of data, there have not been any major changes in the occupancy patterns or levels.
- Nearby sectors present highly correlated data.

Moreover, close sectors behave nearly identically in terms of patterns and occupancy level. In order to avoid repetition of explanations, this section will focus on those sectors that differ in their ACF and PACF plots of the non-differentiated series. Figures 6.53 and 6.54 present visualizations of the ACF and PACF for all Riyadh sectors. To the naked eye, they are very similar.

In order to evaluate the similarity between the time series realizations of the sectors, the following methods have been used:

- Clustering of the ACF and PACF values using different algorithms: k-means and

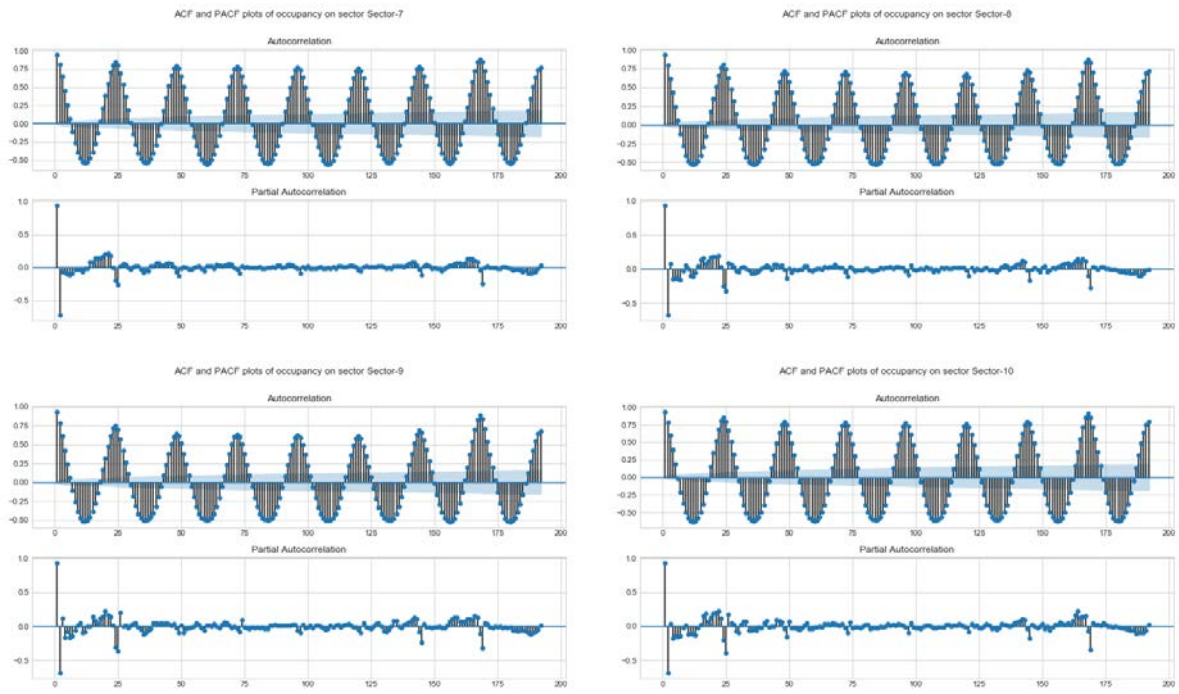


Figure 6.54: ACF and PACF plots for the non-differentiated series from Riyadh. From top left to bottom right: Sector-7 to Sector-10.

hierarchical clustering (single linkage). The optimal number of clusters is measured by the Calinski-Harabasz score.

- Clustering of the time series by computing their distance through Dynamic Time Warping and then using different cluster algorithms: k-means and hierarchical clustering (single linkage).

In-depth information about the clustering algorithms can be found in Bishop [2006]. Dynamic Time Warping is a dynamic programming algorithm that provides the distance between sequences and time series, and this distance is invariant to the size and alignment of the compared elements. In-depth information can be found in Keogh et al. [2005], Warren Liao [2005].

Tables 6.30 show the results of the different cluster techniques when considering the different distance/similarity matrices. The majority of the results indicate that there are 2 different clusters that can be interpreted as being the north and the south sectors. It is for this reason that this section focuses on presenting the ARIMA modeling for Sector-1 and Sector-10, as those are the northernmost and southernmost sectors.

6.8.1 Northernmost and southernmost sectors

This section includes the experiment corresponding to Sector-1 and Sector-10 in Riyadh, namely *ARIMA_ riyadh_sector1* and *ARIMA_ riyadh_sector10.*

Algorithm	Sector-1	Sector-2	Sector-3	Sector-4	Sector-5	Sector-6	Sector-7	Sector-8	Sector-9	Sector-10	Calinski-Harabasz score
K-means clusters	1	1	1	0	0	0	0	0	0	0	13.811
Hierarchical clustering	1	1	1	0	0	0	0	0	0	0	13.811
Algorithm	Sector-1	Sector-2	Sector-3	Sector-4	Sector-5	Sector-6	Sector-7	Sector-8	Sector-9	Sector-10	Calinski-Harabasz score
K-means clusters	1	1	1	0	0	0	0	0	0	0	2.135
Hierarchical clustering	1	0	1	1	0	0	0	0	0	0	2.442
Algorithm	Sector-1	Sector-2	Sector-3	Sector-4	Sector-5	Sector-6	Sector-7	Sector-8	Sector-9	Sector-10	Calinski-Harabasz score
K-means clusters	1	1	1	1	1	1	0	0	0	0	2.135
Hierarchical clustering	0	0	0	0	0	1	0	0	0	0	2.442

Table 6.30: Clustering results: label assignment and scores. From top to bottom: ACF, PACF and by distance from DTW.

Difference order	Sector-1 Mean	Sector-1 Variance	Sector-10 Mean	Sector-10 Variance
No differentiation	0.5306	0.1005	0.5727	0.085
One regular	0	0.0144	0	0.0111
Two regular	0	0.0118	0	0.0082
One seasonal	0.0005	0.0463	0.0002	0.0244
Two seasonal	-0.0004	0.0996	-0.0001	0.0641
Regular plus seasonal	0	0.0063	0	0.005
Two regular, one seasonal	0	0.0071	0	0.0068
One regular, two seasonal	0	0.0149	0	0.0135
Two regular, two seasonal	0	0.0177	0	0.0191

Table 6.31: Means and variances of Riyadh sectors under different numbers of differentiation operations.

Daily box plots of the first and second weeks of March 2019 for Riyadh Sector-1 are presented in Figure 6.55 under different numbers of differentiation operations. It can be seen that the one non-differentiated series presents two behaviours during the week, separating weekdays and weekends. Using regular differentiation, the mean becomes constant with value zero and the same happens for the one regular plus one seasonal differentiation. The one seasonal differentiation suggests that the series has different means and different variances, which depend on the days of the week.

Daily box plots of the first and second weeks of March 2019 for Riyadh Sector-10 are presented in Figure 6.56 under different numbers of differentiation operations. It can be seen that the one non-differentiated series presents two behaviours during the week, separating weekdays and weekends, although the levels of occupancy are higher than those of the previously presented Sector-1. Using regular differentiation, the mean becomes constant with value zero and the same happens for the one regular plus one seasonal differentiation. The one seasonal differentiation suggests that the series has different means and different variances, which depend on the days of the week.

Table 6.31 presents the variance and the mean of Sector-1 and Sector-10. For both sectors, the variance hits minimum at one regular plus one seasonal differentiation.

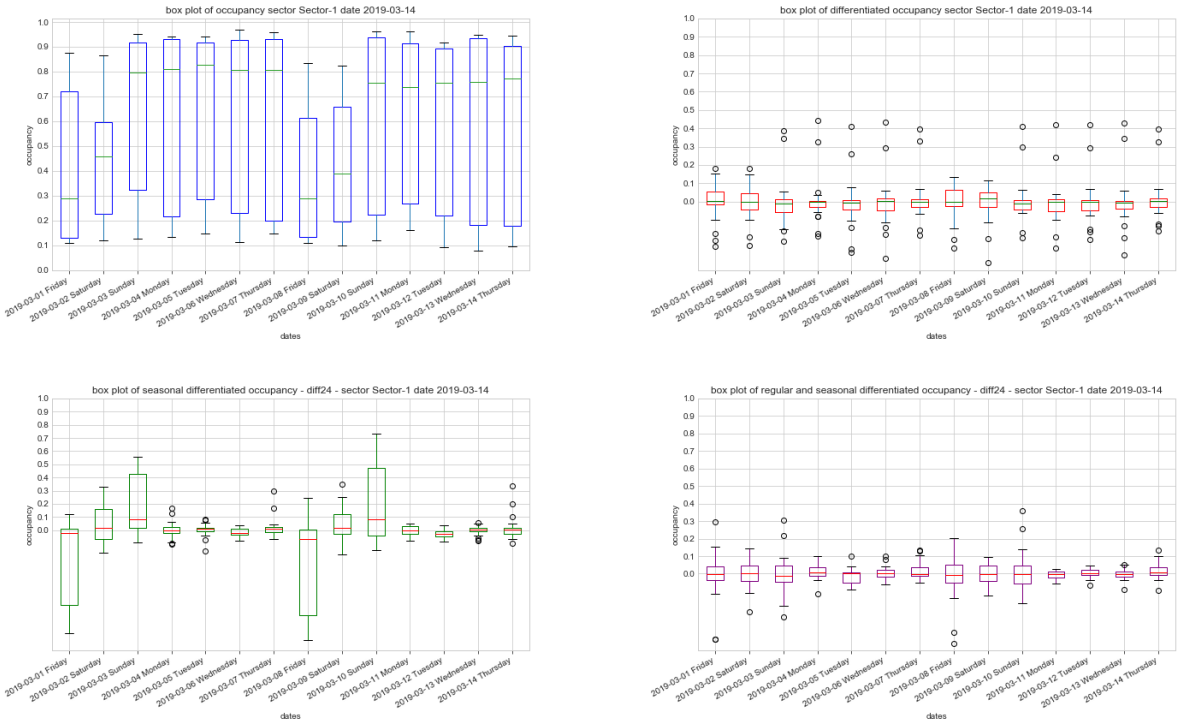


Figure 6.55: Daily box plots for Sector-1 in Riyadh. From top left to bottom right: non-differentiation, one regular differentiation, one seasonal differentiation and one regular plus one seasonal differentiation.

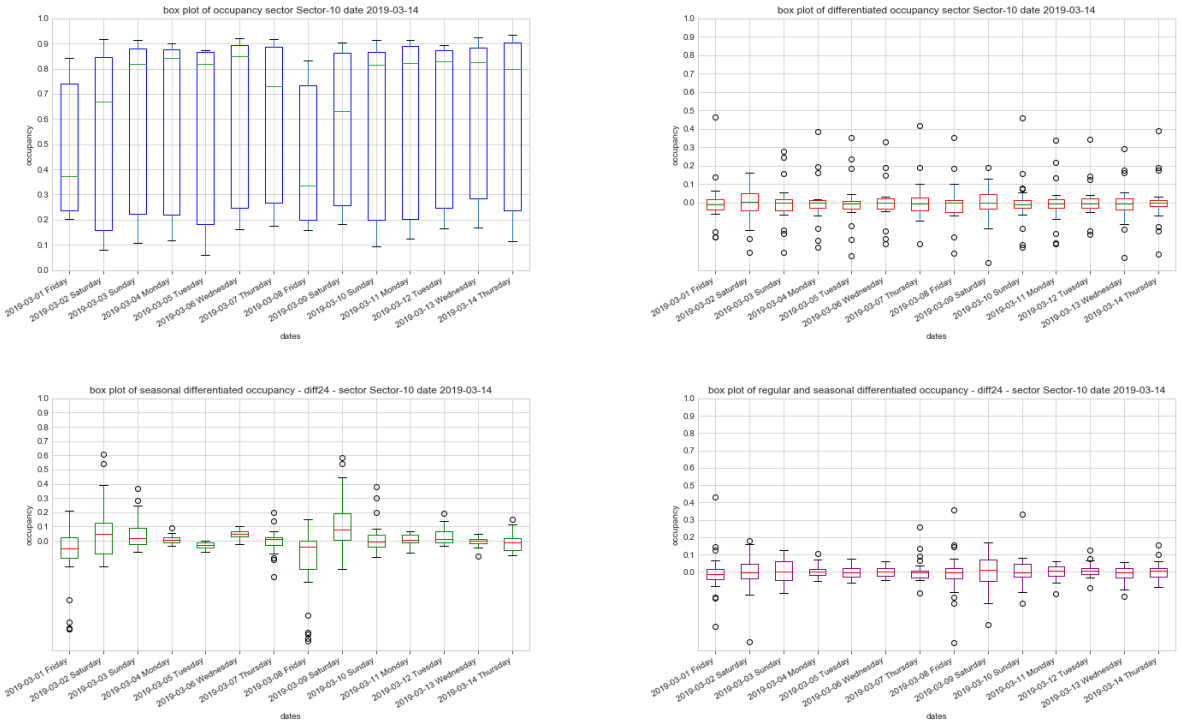


Figure 6.56: Daily box plots for Sector-10 in Riyadh. From top left to bottom right: non-differentiation, one regular differentiation, one seasonal differentiation and one regular plus one seasonal differentiation.

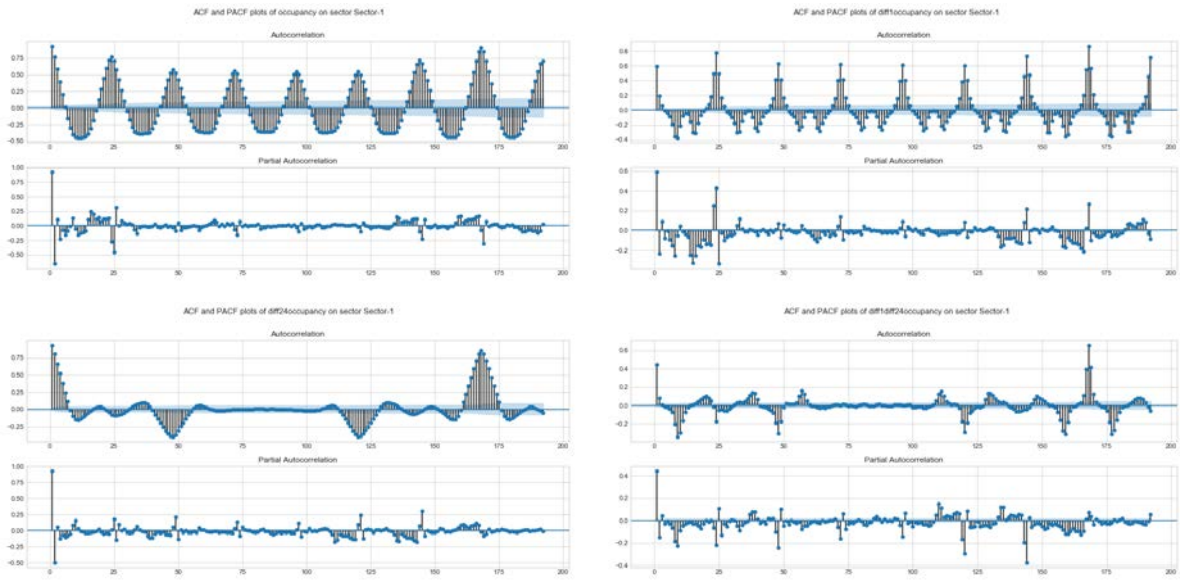


Figure 6.57: ACF and PACF visualizations for Riyadh Sector-1 under different numbers of differentiations.

From the information gathered in the above visualizations and tables, together with the ACF and PACF plots presented in Figures 6.57 and 6.58, the following ARIMA models can be suggested:

- $SARIMA(2,0,0)(1,0,0)_{24}$ for both non-differentiated series.
- $SARIMA(1,1,1)(0,1,2)_{24}$ for Sector-1 with one regular plus one seasonal differentiated series.
- $SARIMA(1,1,1)(1,1,1)_{8}$ for Sector-1 with the one regular plus one seasonal differentiation series. The idea of this model is to capture multiple seasonalities that arise around eight and, if doing this, to also check whether it captures the daily seasonality, as twenty-four is multiple of eight.
- $SARIMA(1,1,1)(1,1,1)_{24}$ for Sector-10 with one regular plus one seasonal differentiated series.
- $SARIMA(12,1,12)(0,1,1)_{24}$ for Sector-10 with the one regular plus one seasonal differentiated series. The idea behind this proposal is to capture all those first lags that are significant in the correlations and compare them with a more simple model.

Table 6.32 presents the results from the candidate models for both sectors. For Sector-1, the best model is $SARIMA(1,1,1)(0,1,2)_{24}$. The score for the candidate model with eight seasonalities falls far behind the other two candidate models; thus, it is not a good candidate. For Sector-10, the model with the highest score is $SARIMA(12,1,12)(0,1,1)_{24}$, but it has a high number of parameters that are non-significant under the t-test. Moreover, its score is not greater than the more simple model $SARIMA(1,1,1)(1,1,1)_{24}$.

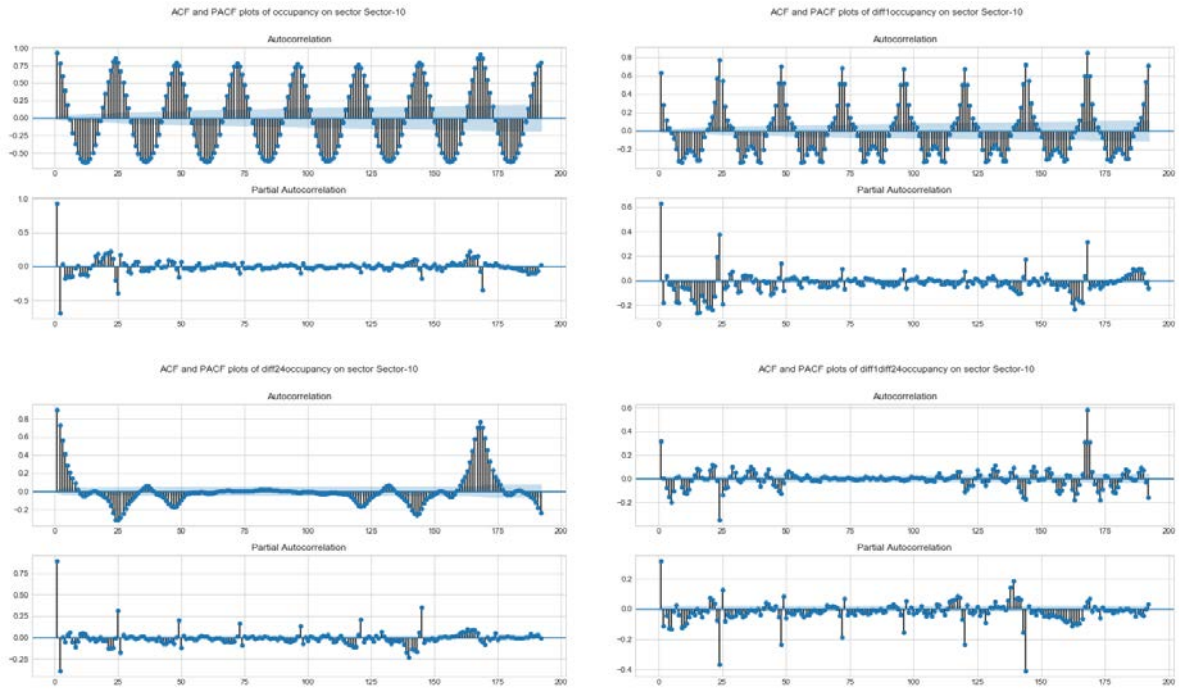


Figure 6.58: ACF and PACF visualizations for Riyadh Sector-10 under different numbers of differentiations.

Given these facts, and following the principle of parsimony, the simpler model is chosen to be proposed.

Equation 6.24 is for the model $SARIMA(1,1,1)(0,1,2)_{24}$ and Equation 6.25 is for the model $SARIMA(1,1,1)(1,1,1)_{24}$.

$$\hat{x_t} = 0.1422_{(0.019)}x_{t-1} - 0.3747_{(0.020)}w_{t-1} - 0.5278_{(0.007)}w_{t-24} - 0.3379_{(0.008)}w_{t-48} + w_t, \tag{6.24}$$

with $\sigma^2 = 0.0036$. Lower values in parenthesis correspond to the estimates of the standard errors for each parameter. w_{t-n} and x_{t-n} represent, respectively, the white noise values and the past series values for a number of n lags considered.

$$\hat{x_t} = 0.0564_{(0.029)}x_{t-1} - 0.3337_{(0.029)}w_{t-1} - 0.1792_{(0.009)}x_{t-24} - 0.9022_{(0.004)}w_{t-24} + w_t, \tag{6.25}$$

with $\sigma^2 = 0.0029$.

Because the first lag of the proposed Sector-10 model is non-significant under the t-test, the proposed model becomes $SARIMA(0,1,1)(1,1,1)_{24}$, with the equation presented in Equation 6.26. This model has an AIC of -26128.366 and a BIC score of -26100.066,

Model	Log likelihood	AIC	BIC
$SARIMA(2,0,0)(1,0,0)_{24}$	11654.363	-23298.725	-23263.336
$SARIMA(1,1,1)(0,1,2)_{24}$	12221.376	-24432.752	-24397.376
$SARIMA(1,1,1)(1,1,1)_{8}$	8747.727	-17485.454	-17450.070

Model	Log likelihood	AIC	BIC
$SARIMA(2,0,0)(1,0,0)_{24}$	12382.249	-24754.497	-24719.108
$SARIMA(1,1,1)(1,1,1)_{24}$	13070.469	-26130.938	-26095.563
$SARIMA(12,1,12)(0,1,1)_{24}$	13530.561	-27009.121	-26825.169

Table 6.32: ARIMA candidate models for Sector-1 and Sector-10 Riyadh. Top table is Sector-1 and bottom is Sector-10.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
0.05216	0.0955	0.1289	0.1564	0.1793	0.1999

Table 6.33: Riyadh Sector-1 ARIMA RMSE

which are slightly better than the previous model.

$$\hat{x_t} = 0.3808_{(0.009)} w_{t-1} + 0.1801_{(0.009)} x_{t-24} - 0.9027_{(0.004)} w_{t-24} + w_t, \tag{6.26}$$

with $\sigma^2 = 0.0029$.

In order to test the hypothesis of the ARIMA models, visualizations of the residuals for the model proposed for Sector-1 are presented in Figure 6.59. Similarly to models from other sectors, the model captures part of the patterns in the data but not all of them. Specifically, it cannot deal with those patterns linked to seasonalities that have not been explicitly modeled, such as those that occur around lag nine.

In order to compare the model proposed for Sector-1 with the discarded model $SARIMA(1,1,1)(1,1,1)_8$, Figure 6.60 presents the residuals of the model. The most noticeable difference is that, although this model more or less handles the first lags, it fails with the daily seasonalities.

The training data for creating the proposed model $SARIMA(1,1,1)(0,1,2)_{24}$ comprises the interval from September 2018 until August 2019. Out-of-sample data for testing purposes comprises the interval from September 2019 until the end of October 2019. Table 6.33 presents the RMSE of the model at each of the considered forecasting hours, from one up to six.

Figure 6.61 shows the one-step forecasts for the ARIMA model proposed for Sector-1 in Riyadh. The model accurately follows the changes in occupancy throughout the days but sometimes over-estimates and under-estimates the peaks.

Visualizations of the residuals for the candidate model $SARIMA(0,1,1)(1,1,1)_{24}$ for Sector-10 are presented in Figure 6.62. The model captures the patterns for the first

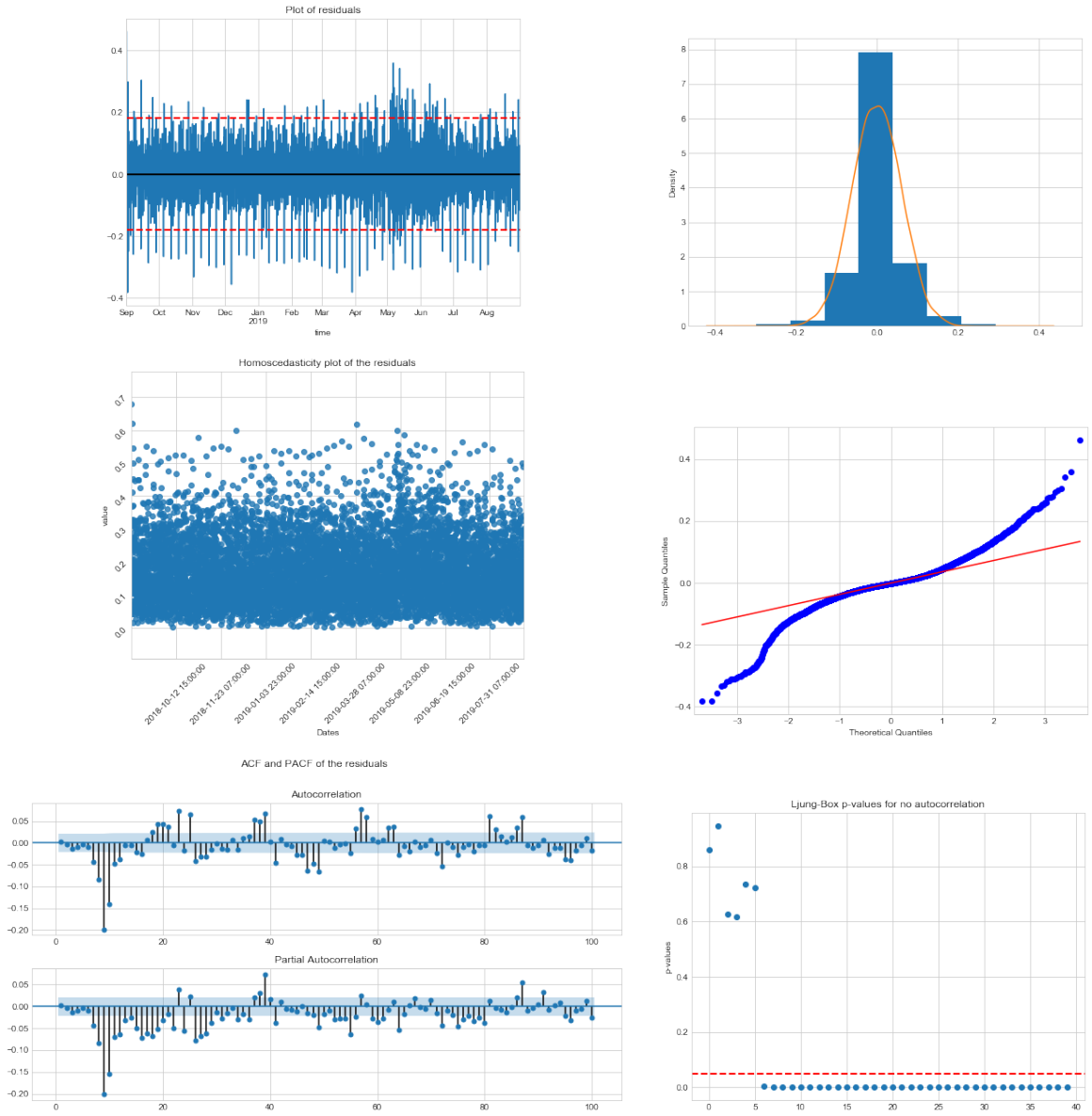


Figure 6.59: Riyadh Sector-1 visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

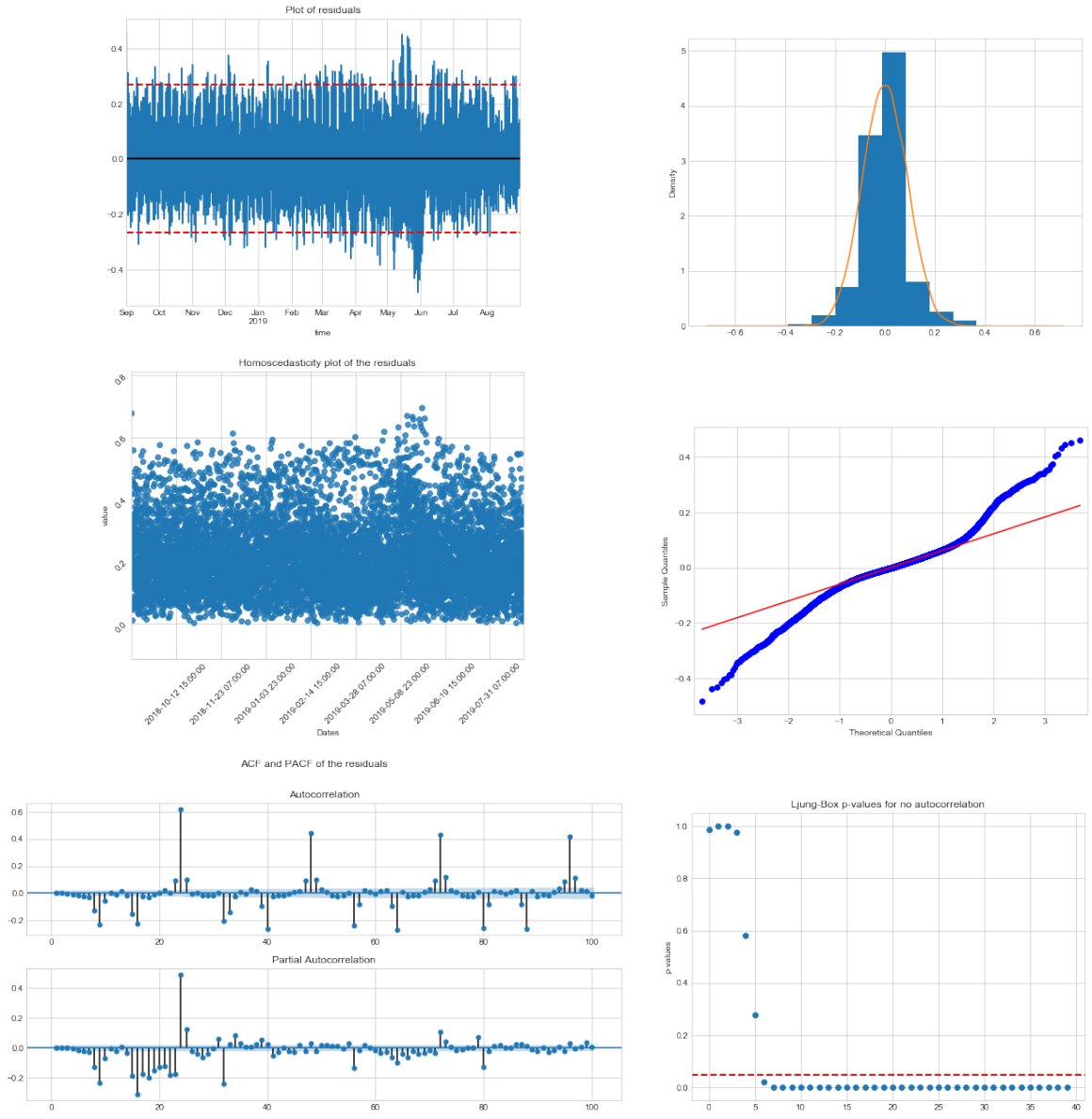


Figure 6.60: Riyadh Sector-1 visualizations and residual tests of the discarded model with seasonality eight. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

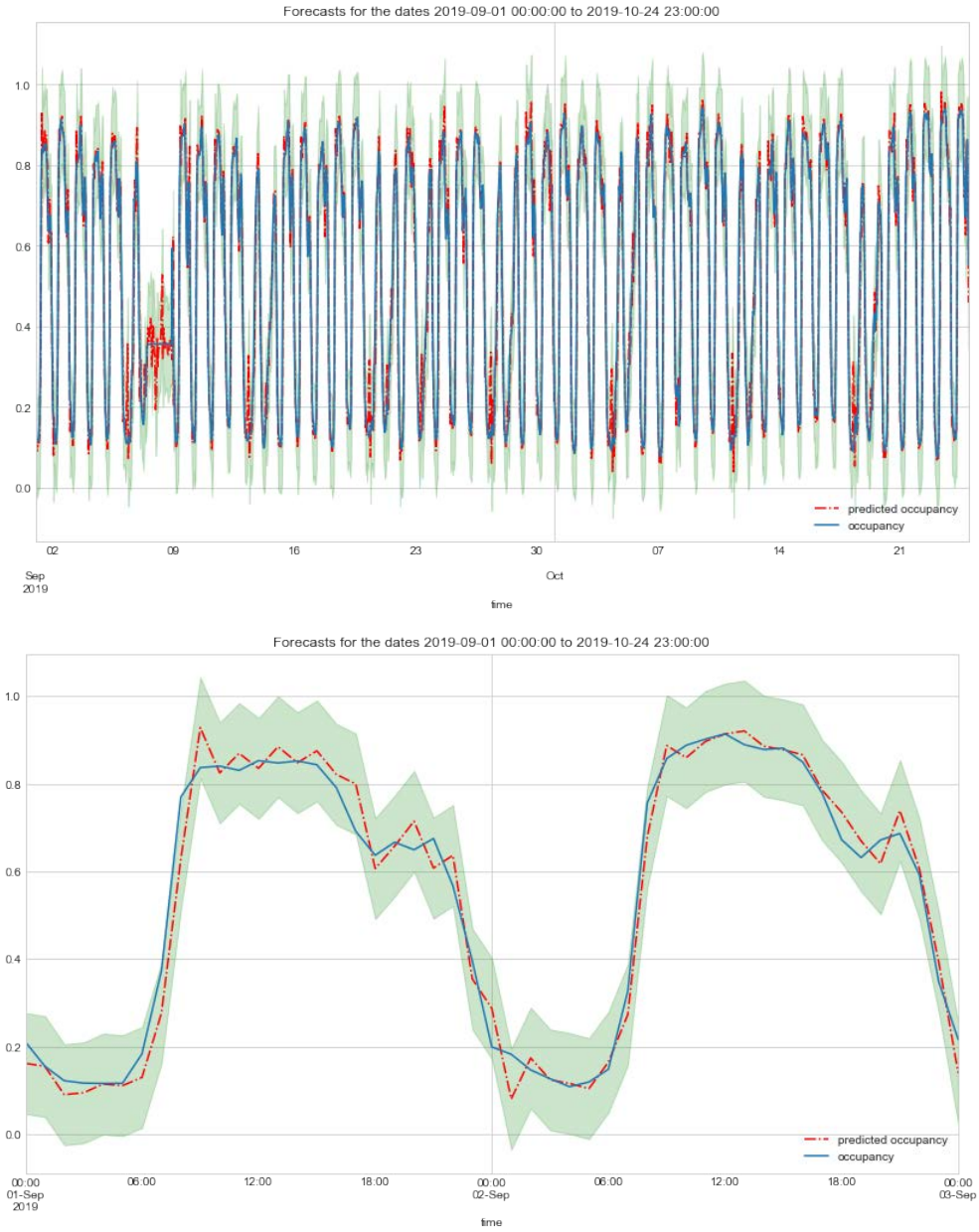


Figure 6.61: ARIMA forecasts from the model proposed for Sector-1 from Riyadh. Top presents forecasts for the entire out-of-sample interval while bottom is the first two days of the same interval of time. Blue denotes the observed occupancy, red the occupancy forecasted by the model and green is the confidence interval.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
0.05	0.0851	0.1117	0.1318	0.1458	0.1553

Table 6.34: Riyadh Sector-10 ARIMA RMSE

lags and the daily seasonality, but it is not capable of dealing with those seasonalities that have not been taken into account explicitly, such as the one that arises every five lags. In order to deal with such issues, it is possible to use intervention analysis methods and exogenous variables. Another way could be to use models that have less constraints on using a high number of parameters.

The training data for creating the models comprises the interval from September 2018 until August 2019. Out-of-sample data for testing purposes comprises the interval from September 2019 until the end of October 2019. Table 6.34 presents the RMSE of the model at each of the considered forecasting hours, from one up to six.

Visualizations in Figure 6.63 present the forecasts of the model for Sector-10 in Riyadh. Similarly to the forecast visualizations of Sector-1, the forecast values are as expected, although they fall at the peaks due to their overestimating or underestimating the expected values.

6.9 Summary of the results and conclusions

By means of the different experiments presented throughout this chapter, the following has been observed:

- All sectors present a general pattern in which the most influential past observations are the recent ones and the one that coincides with daily seasonality.
- One seasonal plus one regular differentiation is the transformation that most sectors need in order to reduce the constant mean to zero and obtain the minimal variance value.
- The majority of the sectors presents a common modeling pattern for the seasonality, that is, $SARIMA(\cdot, 1, \cdot)(0, 1, 1)_{24}$. One common regular pattern is $ARIMA(2, 1, 2)$, but it depends completely on the sector characteristics.
- The model $ARIMA(1, 0, 0)$ with trend offers good results in most of the sectors.
- When sectors are close in terms of physical distance and have high cross-correlation values, as in the cases of *ARIMA_ antwerp_vanwesenbekestraat* and *ARIMA_ antwerp_vanwesenbekestraat2*, a single model could be used because the patterns are nearly identically, as well as the occupation values.
- Most of the sectors present multiple seasonalities that cannot be modeled using ARIMA, as only one seasonal component can be considered.

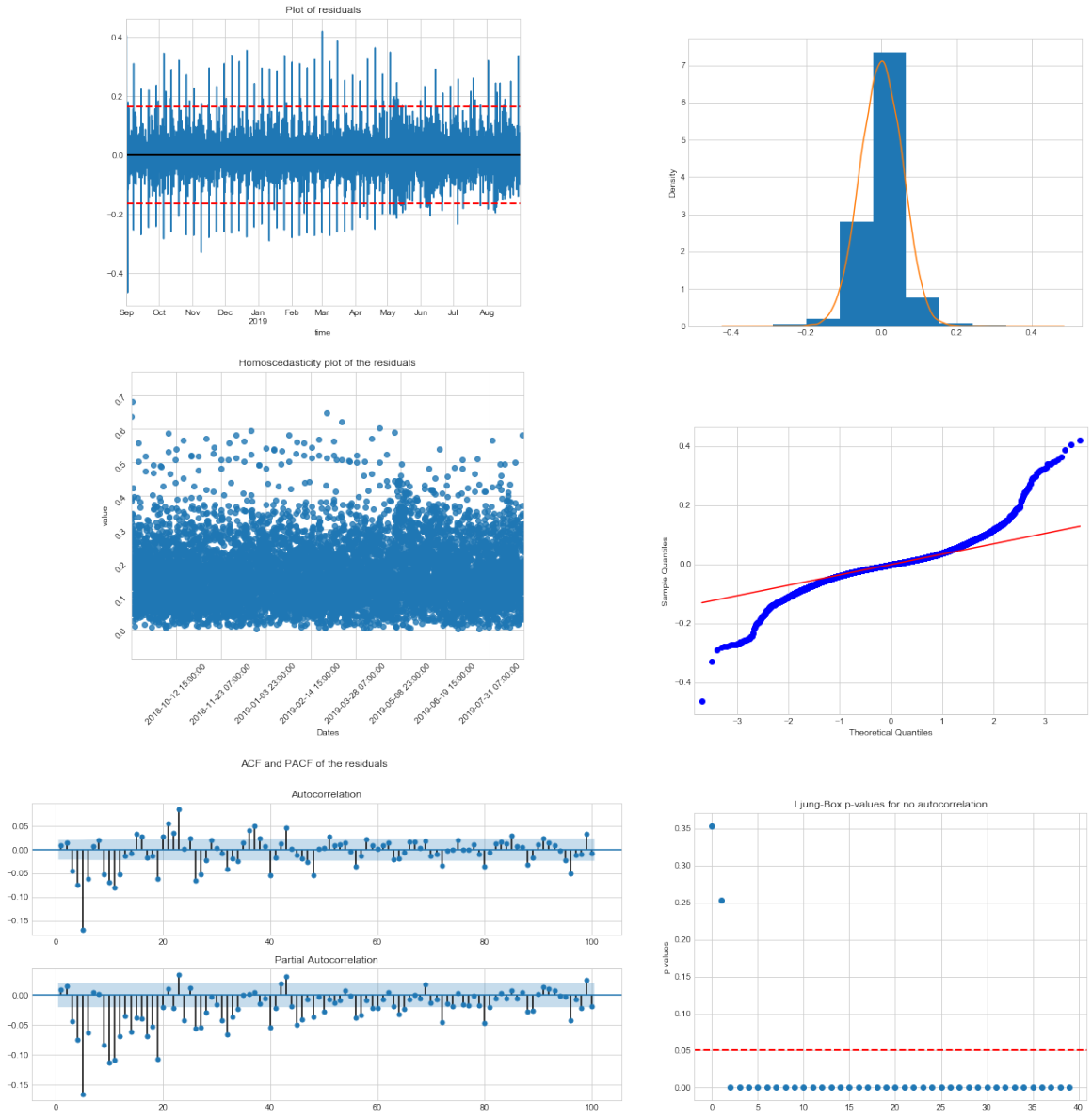


Figure 6.62: Riyadh Sector-10 visualizations and residual tests of the proposed model. From top left to bottom right: residual plot, density plot, homoscedasticity plot, Q-Q plot, ACF and PACF plots and Q-statistic p-values plot.

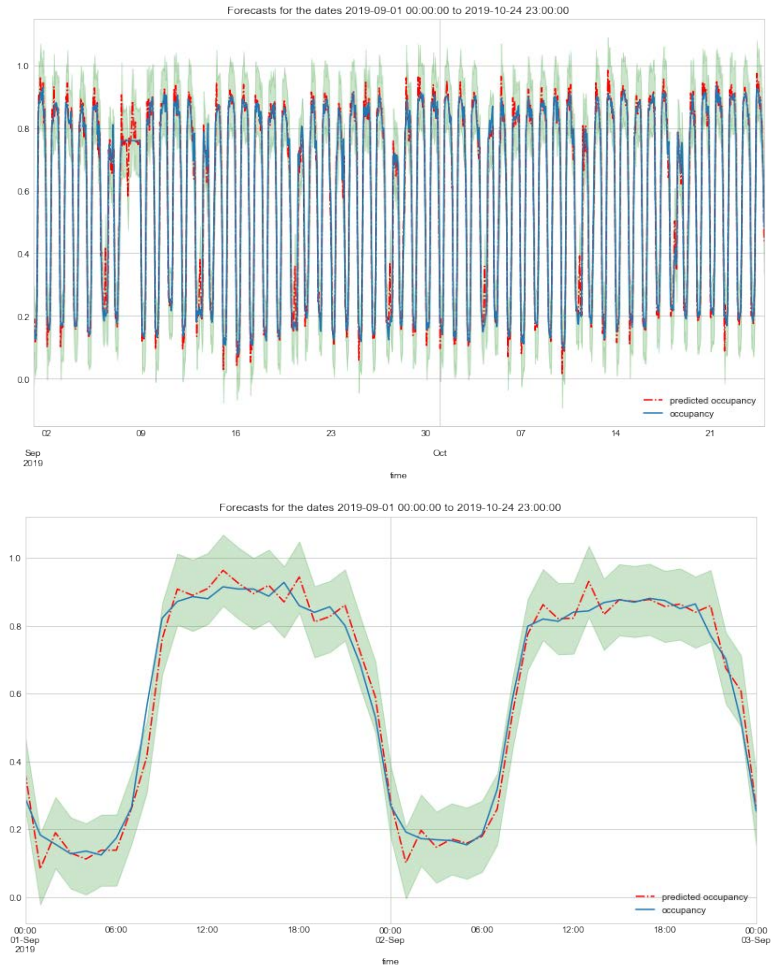


Figure 6.63: ARIMA forecasts from the model proposed for Sector-10 from Riyadh. Top presents forecasts for the entire out-of-sample interval while bottom is the first two days of the same interval of time. Blue denotes the observed occupancy, red the occupancy forecasted by the model and green is the confidence interval.

- High order models suggested by ACF and PACF plots do not have higher scores than simpler models.
- Any of the proposed models do not achieve a normal distribution in their residuals. The reasons for this are specific to each sector, but the most common one is that multiple-seasonalities are present in the data.
- Although the normality conditions are not satisfied, the models provide accurate one-step forecasts when using out-of-sample data.
- Temporary changes in the mean and variance have an important negative effect on the forecasts and the models need to be retrained with new data. If the patterns continue to be the same, fitting the models with all the available data has the effect of making it adaptable to the old and new occupancy values. This has the benefit of not needing to retrain the models when the temporary changes have stopped.
- Sectors with a high number of sensors that are interconnected in the same area present very similar patterns and values, as in the case of the Riyadh scenario. For scenarios such as this, it is useful to cluster the sectors and propose a model for each cluster, and this will be used for all the sectors belonging to that cluster.

Moreover, specific observations for some of the experiments are:

- *ARIMA_ antwerp_vanwesenbekestraat* and *ARIMA_ antwerp_vanwesenbekestraat2* presented models with very similar values in their estimated parameters. This indicates that it is possible to use a single model. The same happened with the models from the experiments *ARIMA_ antwerp_vanschoonhovestraat1* and *ARIMA_ antwerp_vanschoonhovestraat3* and *ARIMA_ wattens_inside* and *ARIMA_ wattens_outside*.
- Random walks are identified in those sectors with low numbers of parking spots, as observed in the experiments *ARIMA_ antwerp_vanschoonhovestraat2* and *ARIMA_ LA_zipcar.*
- The experiment *ARIMA_ barcelona_regular* highlighted that SARIMA models cannot handle multiple seasonalities like those that arise in parking sectors.
- Temporal changes in the mean and variance can be overcome if the model is trained with all the available data, as in the case of the experiments *ARIMA_ LA_standard, ARIMA_ LA_permit and ARIMA_ LA_disabled*.
- When sectors are interconnected with each other and have a high number of parking spots, it is possible to group them in clusters and use models for one of the sectors in each cluster, as has been observed in the experiment *ARIMA_ riyadh_sector1*.

Table 6.35 presents a summary of the results obtained from each of the experiments. The order is provided for the proposed ARIMA model, as well as the RMSE obtained in the test data with different forecasting horizons.

Conclusions that can be extracted from the modeling process with ARIMA are:

Experiment name	ARIMA model	RMSE (1 to 6 hours)					
ARIMA_antwerp_pelikaanstraat	$SARIMA(0,1,2)(0,1,1)_{24}$	0.1033	0.1360	0.1562	0.1737	0.1881	0.1994
ARIMA_antwerp_vanwesenbekestraat	$SARIMA(1,0,0)(0,1,1)_{24}$	0.0778	0.0997	0.1073	0.1110	0.1130	0.1146
ARIMA_antwerp_vanwesenbekestraat2	$SARIMA(1,0,0)(0,1,1)_{24}$	0.0824	0.1039	0.1130	0.1190	0.1221	0.1233
ARIMA_antwerp_vanschoonhovestraat1	$ARIMA(1,0,0)$	0.1042	0.1259	0.1367	0.1423	0.1459	0.1489
ARIMA_antwerp_vanschoonhovestraat2	$ARIMA(1,0,0)$	0.1310	0.1727	0.1975	0.2109	0.2207	0.2271
ARIMA_antwerp_vanschoonhovestraat3	$ARIMA(1,0,0)$	0.1307	0.1540	0.1642	0.1710	0.1731	0.1738
ARIMA_barcelona_regular	$SARIMA(2,1,2)(0,1,1)_{24}$	0.0656	0.0929	0.1079	0.1157	0.1218	0.1277
ARIMA_wattens_inside	$SARIMA(2,1,2)(0,1,1)_{24}$	0.0501	0.0825	0.1060	0.1244	0.1368	0.1465
ARIMA_wattents_outside	$SARIMA(2,1,2)(0,1,1)_{24}$	0.0587	0.0989	0.1248	0.1436	0.1561	0.1665
ARIMA_LA_standard	$SARIMA(1,1,2)(0,1,2)_{24}$	0.0396	0.0672	0.0849	0.0971	0.1064	0.1140
ARIMA_LA_permit	$SARIMA(1,1,1)(0,1,1)_{24}$	0.0179	0.0259	0.0294	0.0322	0.0334	0.0344
ARIMA_LA_disabled	$SARIMA(0,1,1)(0,1,1)_{24}$	0.0387	0.0594	0.0734	0.0827	0.0893	0.0929
ARIMA_LA_zipcar	$SARIMA(0,1,0)(0,1,1)_{24}$	0.0643	0.0913	0.1056	0.1139	0.1212	0.1261
ARIMA_riyadh_sector1	$SARIMA(1,1,1)(0,1,2)_{24}$	0.05216	0.0955	0.1289	0.1564	0.1793	0.1999
ARIMA_riyadh_sector10	$SARIMA(0,1,1)(1,1,1)_{24}$	0.05	0.0851	0.1117	0.1318	0.1458	0.1553

Table 6.35: Experiments results for ARIMA.

- A daily weighted moving average is present in nearly all the sectors, so parking occupancy values are in part affected by the most recent occupation realizations and the daily mean occupancy. This has a negative effect when the time series realizations have two very different patterns, such as weekdays and weekends, because the daily seasonality stops having an effect on the forecasts for the weekends.
- Sectors that are close in physical distance and with high cross-correlation values can be modeled with a single model because the patterns are nearly identically, as well as the occupation values. It would be of interest to further investigate the conditions under which this observation is valid, as sector distance and place (type and quantity) clearly play a role in this.
- Techniques like outlier detection and intervention analysis can further enhance the capabilities of the models. What is more, the use of exogenous variables could be of great interest, especially in the case of the Los Angeles scenario.
- Multiple seasonalities and a high number of correlations are not treated well with the ARIMA approach, but other methods like NN can provide natural ways to deal with these issues when facing these types of characteristics in the data.
- It is preferable to train models with all the available data rather than with only the most recent data.
- Although their residual distributions are not Gaussian, one-step forecasts are accurate.
- It is possible that the large sample size (8064 observations for a year of data) has a negative effect on the models, which is reflected in their residuals. This could be

investigated by using adaptive models.

- Sectors with low numbers of parking spots present a random walk signature. It would be of interest to investigate the conditions that cause the sectors to change from random walk to a more complex process.

7 Computational experiments - Neural Networks

This chapter presents the characterization of the sectors presented in Chapter 3, on which we used Neural Network (NN) methods. Similarly to Chapter 6, the goal is to provide hourly forecasts of the parking occupancy of the different sectors using past occupancy values. Similarly also to the previous chapter, a study of the accuracy at different forecasting horizons is made between the obtained results from each of the NN methods considered.

The organization of the chapter is as follows. First, in order to provide structure and context, we present the experimental design used and the characteristics of each of the scenarios. Next, we introduce the validation framework used to obtain and compare the models. Following this, we present the structure applied to the data for making it appropriate for the NN methodologies. After, we present a set of experiments performed with the goal of finding the level of signifance of the hyperparameters, quantify its effects when training different NN models with the data of our problem and reduce the computational load when performing the hyperparameters tuning. Then, the characterization of each of the sectors using different NN methods is presented together with their results at different forecasting horizons. Finally, a summary of the important facts and observations that emerged during the experiments are commented as the concluding remarks for the chapter.

7.1 Design of experiments

An experiment is considered to be the modeling process for each sector in each city using a specific NN method. The NN methods considered are:

- Multilayer perceptron (MLP).
- The recurrent neural networks LSTM and GRU.

For each experiment, the outcome will be the best model from each methodology within the validation framework, with comments about facts that emerged during the execution and the RMSE values of the best models throughout the different forecasting steps. Table 7.1 shows the experiments with their names and comments about the scenarios (cities and sectors).

Questions that arise from the comments on the experiments are:

City	Sector	Experiment name	Comments
Antwerp	Pelikaanstraat	MLP_endogenous_antwerp_pelikaanstraat	Sectors present different sizes, usage and locations.
		LSTM_endogenous_antwerp_pelikaanstraat	
		GRU_endogenous_antwerp_pelikaanstraat	
	Van Wesenbekestraat	MLP_endogenous_antwerp_vanwesenbekestraat	
		LSTM_endogenous_antwerp_vanwesenbekestraat	
		GRU_endogenous_antwerp_vanwesenbekestraat	
	Van Wesenbekestraat Zone 2	MLP_endogenous_antwerp_vanwesenbekestraat2	
		LSTM_endogenous_antwerp_vanwesenbekestraat2	
		GRU_endogenous_antwerp_vanwesenbekestraat2	
	Van Schoonhovestraat Zone 1	MLP_endogenous_antwerp_vanchoonhovestraat1	
		LSTM_endogenous_antwerp_vanchoonhovestraat1	
		GRU_endogenous_antwerp_vanchoonhovestraat1	
	Van Schoonhovestraat Zone 2	MLP_endogenous_antwerp_vanchoonhovestraat2	
		LSTM_endogenous_antwerp_vanchoonhovestraat2	
		GRU_endogenous_antwerp_vanchoonhovestraat3	
	Van Schoonhovestraat Zone 3	MLP_endogenous_antwerp_vanchoonhovestraat3	
		LSTM_endogenous_antwerp_vanchoonhovestraat3	
		GRU_endogenous_antwerp_vanchoonhovestraat3	
Barcelona	Regular	MLP_endogenous_barcelona_regular	At least two seasonalities: eight hours and daily.
		LSTM_endogenous_barcelona_regular	
		GRU_endogenous_barcelona_regular	
Wattens	Inside	MLP_endogenous_wattens_inside	Both sectors belong to the same zone and have the same usage.
		LSTM_endogenous_wattens_inside	
		GRU_endogenous_wattens_inside	
	Outside	MLP_endogenous_wattens_outside	
		LSTM_endogenous_wattens_outside	
		GRU_endogenous_wattens_outside	
Los Angeles	Standard	MLP_endogenous_LA_standard	2018 and 2019 have different occupancy values because of street works.
		LSTM_endogenous_LA_standard	
		GRU_endogenous_LA_standard	
	Permit	MLP_endogenous_LA_permit	
		LSTM_endogenous_LA_permit	
		GRU_endogenous_LA_permit	
	Disabled	MLP_endogenous_LA_disabled	
		LSTM_endogenous_LA_disabled	
		GRU_endogenous_LA_disabled	
	ZipCar	MLP_endogenous_LA_zipcar	
		LSTM_endogenous_LA_zipcar	
		GRU_endogenous_LA_zipcar	
Riyadh	Sector-1	MLP_endogenous_riyadh_sector1	All sectors are interconnected and present high cross correlation values.
		LSTM_endogenous_riyadh_sector1	
		GRU_endogenous_riyadh_sector1	
	Sector-10	MLP_endogenous_riyadh_sector10	
		LSTM_endogenous_riyadh_sector10	
		GRU_endogenous_riyadh_sector10	

Table 7.1: Table of experiments for NN.

- How different are the sector models based on their idiosyncrasies?
- How different are the sector models based on the NN methods' idiosyncrasies?
- Is there a configuration of hyperparameters that provides better results for the parking occupancy time series?
- Is it possible for an NN model to handle multiple seasonalities?
- Do nearby sectors share the same model?
- Can NN models overcome temporal changes?
- Is a model really necessary for each sector?
- Which of the NN methods achieved better results overall?

7.2 Validation framework

7.2.1 Data subsets

In order to validate the models, data is split into three subsets called the training, validation and test sets. A common procedure is to use 80% of the data for training, the next 10% of the data for validation and the rest for testing, as presented in Figure 7.1. Because the data consists of time series while the data indices give the context of time to the order of the data, a more natural split has been used, namely by making sure that the end and start of a subset coincides with the end and start of consecutive days. So, if the training set ends at 11/09/2019 23:00, then the first data point of the validation set is the one corresponding to 12/09/2019 00:00. The objective of this partition is, first, to use the training and validation subsets by employing the training fold for computing the parameters of the model and, second, to validate the computed model with the validation fold. Because a large number of models will be created with each one having different hyperparameter values, it is possible that the values of the hyperparameters are influenced by the validation dataset. In order to detect if the candidate model has been specialized towards a good score in the validation set, it is then used with the test set, where it is expected to provide a good score – assuming that the data distributions are the same for the three subsets. The score of the candidate model in the test set is the one used to evaluate and compare it with other models.

7.2.2 Hyperparameter values

For the creation of those models where a high number of hyperparameters are involved (such as with NN methods), one common practice in the literature is to use a random search strategy, as proposed in the work by Bergstra and Bengio [2012].

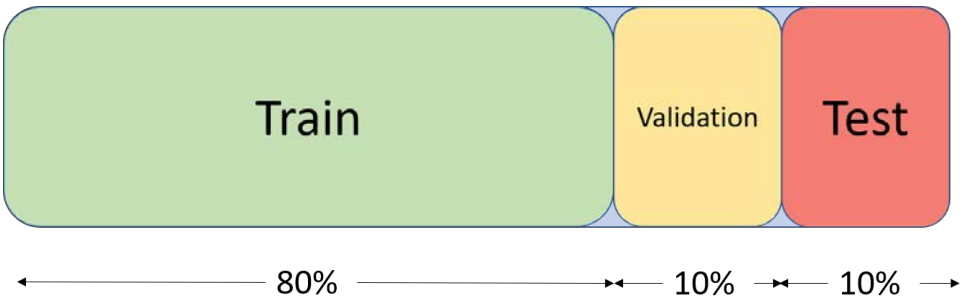


Figure 7.1: Data partition used in this work in order to create and evaluate the models using NN methods.

A random search consists of randomly taking values for the different hyperparameters for N iterations. In the long-run term (N approaching infinity), the best possible combination of the hyperparameters will be found. Luckily, it is not necessary to wait for infinity because, with $N = 60$, there is a 95% probability of finding a good combination of the hyperparameters. In fact, it falls within 5% of the optimal solutions. This can be explained with the following reasoning. If we have an interval of 5%, each random draw has a 5% chance of being within our interval; thus, the probability of missing from n draws is $(1 - 0.05)^n$. So, in order to fall within the interval, the operation becomes $(1-(1-0.05)^n$. Now, to figure out the number of draws needed to have a 95% probability of success, we need to solve the inequation 7.1, whose solution is $n \geq 60$.

$$1 - (1 - 0.05)^n > 0.95 \tag{7.1}$$

where n is the number of random draws needed.

What this means is that, for each of the considered experiments, there will be at least n models trained for each of the NN methodologies using random hyperparameter values. The best model in the validation test is the one chosen to be the candidate model. For those methods with a small number of hyperparameters (or none), we consider a grid search that consists of trying different values for one of the hyperparameters while fixing the values of the others. The possible range of values of each hyperparameter is defined individually for each of the NN methods, because of their idiosyncrasies.

This research proposes a pseudo-random search strategy. Thus, the values considered for each hyperparameter are guided by other research works, common sense and a series of individual experiments that provided intuition about the best value for a considered hyperparameter for parking occupancy data. These individual experiments are presented in Section 7.4, and they consider the following hyperparameters to be calibrated:

- The optimizers SGD with momentum, RMSProp and Adam, all at different learning rates.
- The model capacity: number of layers and number of neurons per layer.
- The normalization in RNN.

For all these experiments, we have considered the same framework, which consists of fixing the values of the other hyperparameters and changing only the hyperparameters that are being calibrated. The data that is used comes from those sectors with the largest number of parking places for each of the cities (for example, the Standard sector is the one considered in the Los Angeles scenario). The intuition gathered from the outcomes of these experiments will allow fixing the values for some of the hyperparameters and reducing the search space for the random search.

Then, for the random search, the hyperparameters considered are constrained to the NN method. The reasons for this are, one, because some of the hyperparameters are exclusive to the method considered (for example, recurrent neural networks can make use of a dropout technique named "recurrent dropout", which is applied to their hidden state) and, two, because the considered range of values for a hyperparameter can be different from one method to another.

Unless otherwise specified, the considered values for the hyperparameters are:

- A power of two value for the number of neurons in the hidden layers that are constrained by the range provided in the calibration procedure.
- A maximum number of hidden layers in order to avoid exploding/vanishing gradient problems, which are also constrained by the range provided in the calibration procedure.
- Linear, sigmoid and ReLu as possible activation functions.
- Different batch sizes have been considered because, first, each one depends on the problem at hand and, second, it is hard to provide reasons for why smaller batch sizes for some data delivers better results than larger batch sizes, and vice versa. Those sizes are 32, 64, 128, 256 and 672.
- For those other hyperparameters that are not mentioned (for example, the initial bias value), the default value of the programming library is used because those are set as the most commonly used values in the literature.

7.2.3 Model choice

Once we have obtained a combination of hyperparameters that result in good candidate models, all data from the training and validation sets are used to train the candidate model and then they are tested with the test set. This will be done at least M times, and the scores obtained will be averaged for each of the times the model has been recreated. This is done in order to overcome the stochasticity presented in the methods considered, as the final results can be affected by some of their inner mechanisms (random initialization of the weights, use of first-order methods for optimization that converge to local optima, etc.).

The loss function used to train the models is the mean squared error (MSE) that is presented in Equation 7.2 (notice that is the same as the L2 distance of Euclidean

spaces and presented at 5.37). The root mean squared error (6.1) function has been used as the models' goodness-of-fit metric to the unseen data, and it also allows us to compare each model's errors with either other models built from the same methodology or from another methodology (for example ARIMA).

$$MSE(y, \hat{y}) = n^{-1} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{7.2}$$

where $n \in \mathbb{N}$ is the number of samples in the current batch, y_i is the outcome of the sample i and $\hat{y}_i$ is the predicted outcome of the model when using x_i as input.

The number of trained models in the random search strategy is 100, and the number of test trials is $M = 20$. The number of epochs is 400, and an early stop strategy has been used to save computation and time for those models that diverge in the early steps of the training.

7.3 Data structure

All the data available from the sectors of each of the scenarios have been organized in a tensor X of dimensions $(batchSize, 24, 1)$, where batch size is a hyperparameter of the model. The decision to create sequences of 24 consecutive elements comes from the ACF/PACF and the visualization of the frequency domain (see Chapter 3 for detailed information). The last dimension (features) is one, because only the endogenous variable is considered. This tensor is used as the set of independent variables or predictors. Then, we create a matrix Y of dimensions $(batchSize, n)$ in which each of the rows (sequences) represents the real values of the following occupancies/hour for each of the sequences in the tensor X. These sequences are used to train the model in a supervised framework and to evaluate the validation and test sets. An example of the organization of the data set can be seen in Table 7.2

The decision to organize the ground truth as a matrix of dimensions $(batchSize, n)$ is because NN outputs can be vectors. Each element of the output vector of the model is

Predictors							**Response**			
$\frac{occupancy}{hour}_{t-6}$	$\frac{occupancy}{hour}_{t-5}$	$\frac{occupancy}{hour}_{t-4}$	$\frac{occupancy}{hour}_{t-3}$	$\frac{occupancy}{hour}_{t-2}$	$\frac{occupancy}{hour}_{t-1}$	$\frac{occupancy}{hour}_{t}$	$\frac{occupancy}{hour}_{t+1}$	$\frac{occupancy}{hour}_{t+2}$	$\frac{occupancy}{hour}_{t+3}$	$\frac{occupancy}{hour}_{t+4}$
0.1	0.2	0.4	0.3	0	0.3	0.5	0.7	0.8	0.7	0.9
0.2	0.4	0.3	0	0.3	0.5	0.7	0.8	0.7	0.9	0.8
0.4	0.3	0	0.3	0.5	0.7	0.8	0.7	0.9	0.8	0.8
0.3	0	0.3	0.5	0.7	0.8	0.7	0.9	0.8	0.8	1
0	0.3	0.5	0.7	0.8	0.7	0.9	0.8	0.8	1	1

Table 7.2: Data set organization example. In this example, the dimensions of the tensor X are $(5, 7, 1)$. On the other hand, the matrix Y has dimensions $(5, 4)$. Notice that consecutive rows represents moving the values to the left (taking out the first value from X and moving the first element of Y towards the last element of its corresponding X), as a representation of the flow of time.

linked to a neuron in the output layer, so all the NN methods used have an output layer of six neurons. What is intended with this strategy is to make the models learn the patterns so that they can output the next six forecasting steps and provide them all at once. This technique is not possible in other models like ARIMA, in which a multi-step strategy is needed in order to obtain predictions for consecutive time steps.

Because of this strategy, the RMSE of the NN models can be computed for the whole matrix Y, although the RMSE for each step is also provided for the candidate model in order to be able to compare it with models from ARIMA.

7.3.1 A note on overlapping windows

The major point that differentiates the MLP method compared to recurrent ones is that MLP requires a fixed length for the time window while recurrent ones do not, as they are iteratively exposed to data at each time step. Moreover, recurrent methods make use of the hidden state and, theoretically, this is the key element that allows decoupling the model whose length of a time window includes past realizations, as the hidden state is responsible for learning the patterns of the series. Despite this, if the dependencies between two realizations of a time series are far apart, in practice the model would not learn them. In order to overcome this, it is possible to use a stateful recurrent model. In a stateful RNN model, the hidden state is preserved across batches of data, meaning that the hidden state of the last step of the last batch is used as the starting hidden state for the first element of the next batch. Although this sounds good, it must be taken into account that backpropagation over time does not allow backpropagating the gradients further from the current batch; thus, if the batch does not contain useful information, the hidden state will reflect this and the model will act in the same way as a stateless one.

In the stateful context, the strategy of modeling the data as sequences using an overlapping window is no longer needed because the last sample from a batch must be continued by the first sample of the next batch. This reduces the amount of data available when compared with the strategy of overlapping windows because, for time series of size T with a window of size w, this provides $T - 1$ individual sequences while the strategy needed for the stateful model provides T/w sequences. This becomes a handicap when modeling time series with coarse granularity in time, as the number of samples becomes scarce and would not be enough to train an NN model. Another issue is that time series data from real world processes are subject to changes in time, which, when added to the scarcity, also discourages the use of a stateful model. Both reasons mentioned here also occur in the problem studied in this work, as one or two years of hourly parking occupancy data (depending on the sector) become between 8760 to 17520 samples. In the case of two years of data using an overlapping window of size 24, we get 17519 samples; while for a stateful model we get 730 samples. Moreover, if after the first year of data the distribution parameters (mean and standard deviation) change like they do in Los Angeles scenario, the risk of not being able to successfully train an NN model increases.

7.4 Hyperparameter calibration

This section presents the experiments conducted to constrain the values of some of the hyperparameters used in the random search. Unless otherwise specified, the baseline configuration used is:

- 512 neurons.
- One hidden layer.
- No dropout.
- ReLu activation function.
- 256 as batch size.
- The He normal as weight initializer.

7.4.1 Optimizers

The goal of this experiment is to compare the behaviour of the optimizers SGD with momentum, Adam and RMSProp for each proposed method. Specifically, it is of interest to observe the evolution of the loss (mean squared error) and the RMSE for the validation dataset at different learning rates for each of the optimizers.

7.4.1.1 MLP

Figure 7.2 presents the RMSE curve of the validation set for the Antwerp Pelikaanstraat scenario. It can be observed that, at low learning rate values, the RMSE of the trained models are unstable and provide high error values. At around 10^{-4}, the RMSE becomes stable for the Adam and RMSProp models. For learning rates of around 10^{-1}, the SGD models start to diverge (no representation in the visualization, as the RMSE becomes positive infinity due to exploding gradients); the RMSE of the Adam models becomes unstable; and the RMSE from the RMSProp models starts to increase. The fact that similar learning rate values offer similar RMSE values can be interpreted as the achievement of the same local optima by the optimizer, thus indicating its stability. Clearly, RMSProp seems to be the most stable of the three optimizers while offering competitive RMSE values.

Figure 7.3 shows the evolution of the loss throughout the training epochs for the Antwerp scenario. The RMSProp optimizer shows that the training loss and the validation loss achieve minimum values with a wider range of learning rate values when compared to SGD and Adam. The learning rates between 10^{-4} and 10^{-2} are the ones that provide lower validation loss and faster convergence.

Notice that, as training progresses, the curves of the validation loss start to increase after the achievement of a minimum. This effect is less noticeable at greater learning

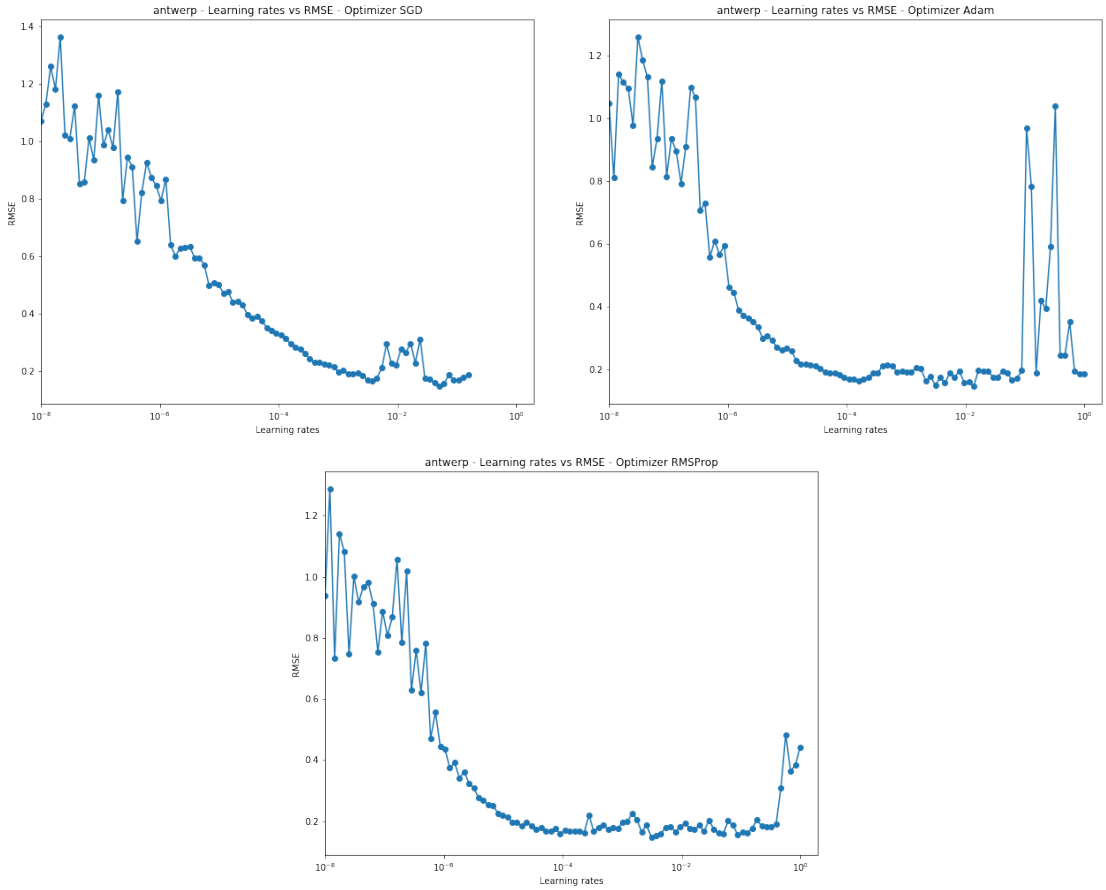


Figure 7.2: Antwerp validation RMSE at different learning rates for the optimizers SGD with momentum (top left), Adam (top right), and RMSProp (bottom).

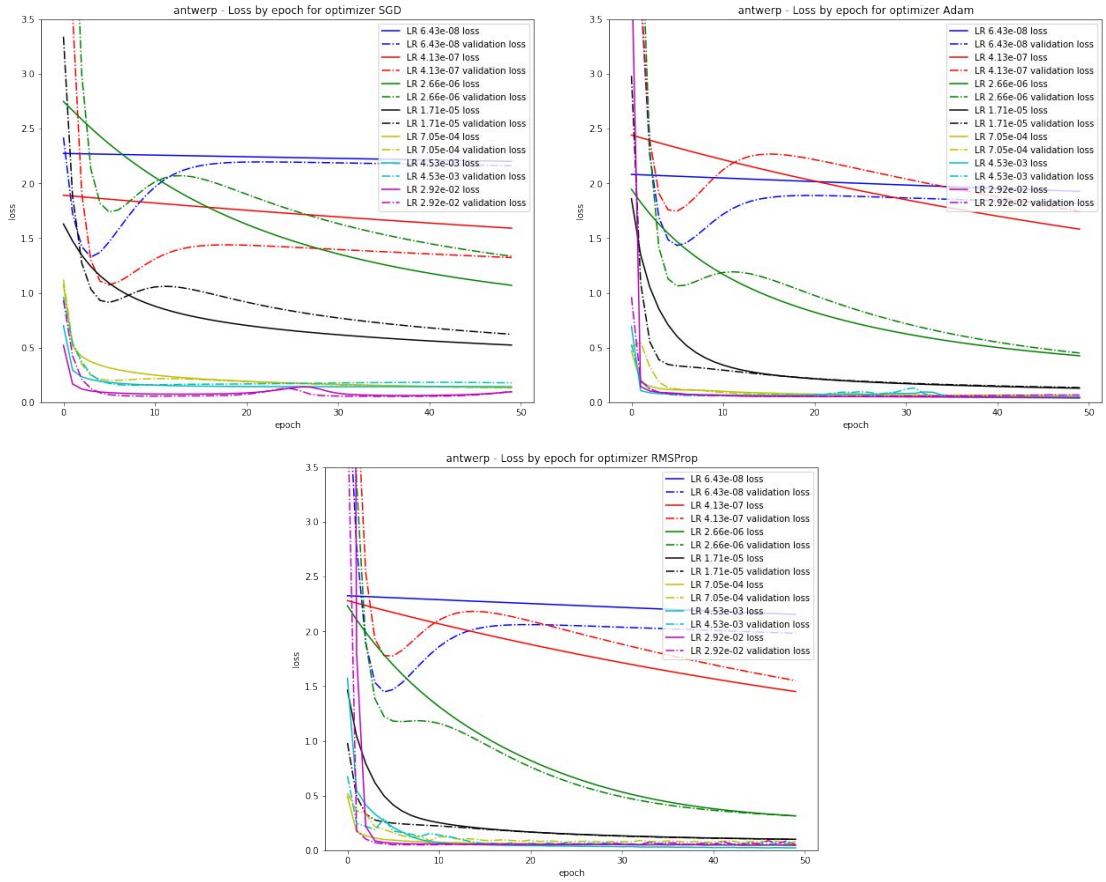


Table 7.3: Evolution of the loss throughout the training epochs for the Antwerp scenario. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

SGD		Adam		RMSProp	
RMSE	LR	RMSE	LR	RMSE	LR
0.1464	0.051	0.1453	0.0138	0.1462	0.0031
0.1564	0.061	0.1486	0.0031	0.1530	0.0038
0.1608	0.042	0.1560	0.0045	0.1554	0.0890
0.1662	0.038	0.1584	0.0095	0.1585	0.00009
0.1674	0.089	0.1595	0.0115	0.1593	0.0509

Table 7.4: Antwerp sorted validation RMSE values and learning rates by optimizer.

rates. This happens in the first epochs because training loss is computed before updating the weights, and validation loss is computed at the same epoch but after updating the weights. Another fact to take into account is that, with the appropriate learning rate, less than fifteen epochs are enough to obtain model convergence.

As the training goes further, training loss and validation loss becomes closer, meaning that the models are not overfitting the training data. Greater learning rates offer tighter proximity between training and validation losses.

Table 7.4 presents the validation RMSE and learning rate values for each optimizer. SGD is the most stable one, as similar learning rates provide similar RMSE values. Adam and RMSProp provide similar validation RMSE with different learning rate values.

For the Barcelona scenario (Figure 7.3), Adam and RMSProp offer a wider range of learning rates and lower values are achieved when compared to SGD with momentum. In this case, the Adam optimizer presents more stable validation RMSE values than RMSProp at higher learning rate values (learning rate over 10^{-1}).

When comparing the behaviour of the loss as training progresses (Figure 7.5), Adam and RMSProp converge faster with large learning rates and also have tighter gaps between training and validation losses.

The table of sorted RMSE values (Table 7.6) for the Barcelona scenario shows that the RMSProp optimizer is the one with fewer errors at similar learning rates.

Figure 7.4 presents the validation RMSE for the different learning rates of the Wattens scenario (Inside sector) for the optimizers SGD with momentum, Adam and RMSProp. Like the Antwerp and Barcelona scenarios, Adam and RMSProp present wider ranges of learning rates that achieve minimum validation RMSE values. Similarly to the Barcelona scenario, RMSProp presents greater instability at higher learning rate values.

The loss behaviours throughout the training epochs are presented in Figure 7.7. Adam and RMSProp offer similar behaviours with lower validation loss values at learning rates of between 10^{-4} and 10^{-2}, and this coincides with the observations from the previous scenarios.

The table of sorted RMSE values (Table 7.8) for the Inside Wattens scenario presents lower values from the optimizers Adam and RMSProp. Between them, RMSProp provides more scores with similar error values at similar learning rates.

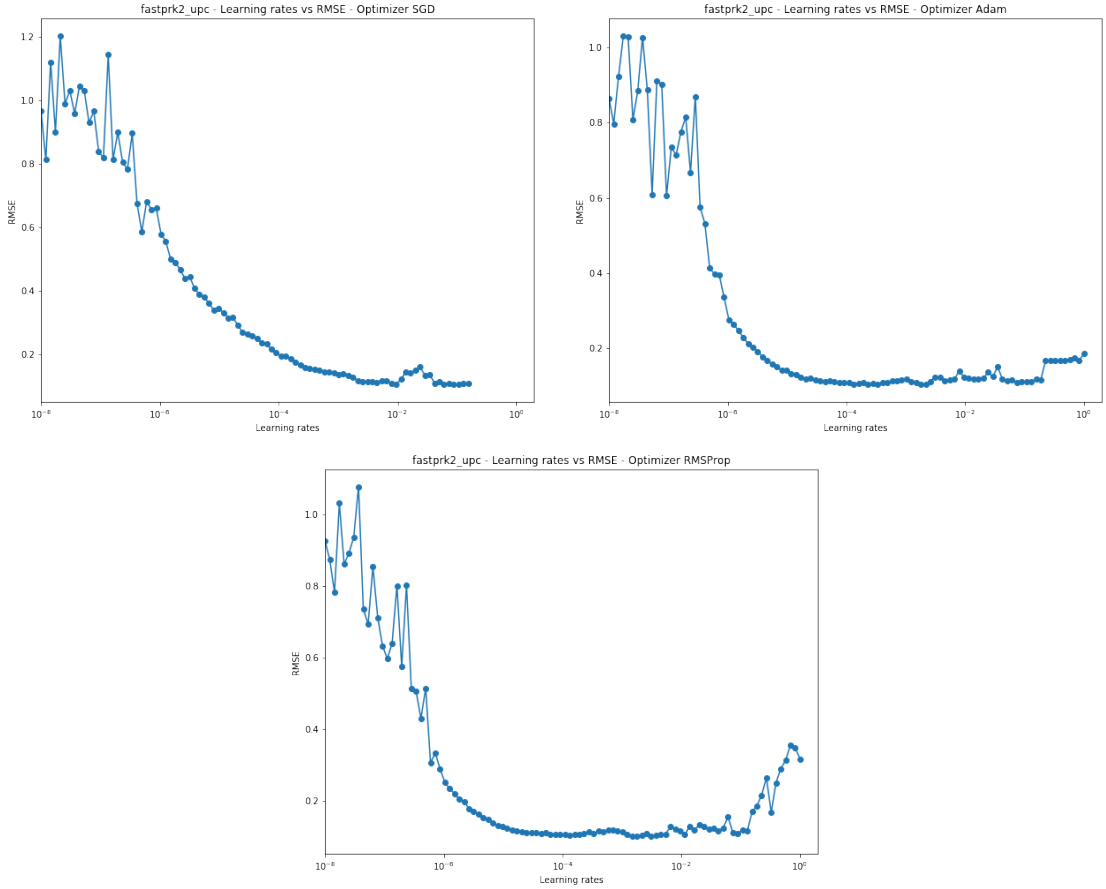


Figure 7.3: Barcelona validation RMSE at different learning rates for the optimizers SGD with momentum (top left), Adam (top right), and RMSProp (bottom).

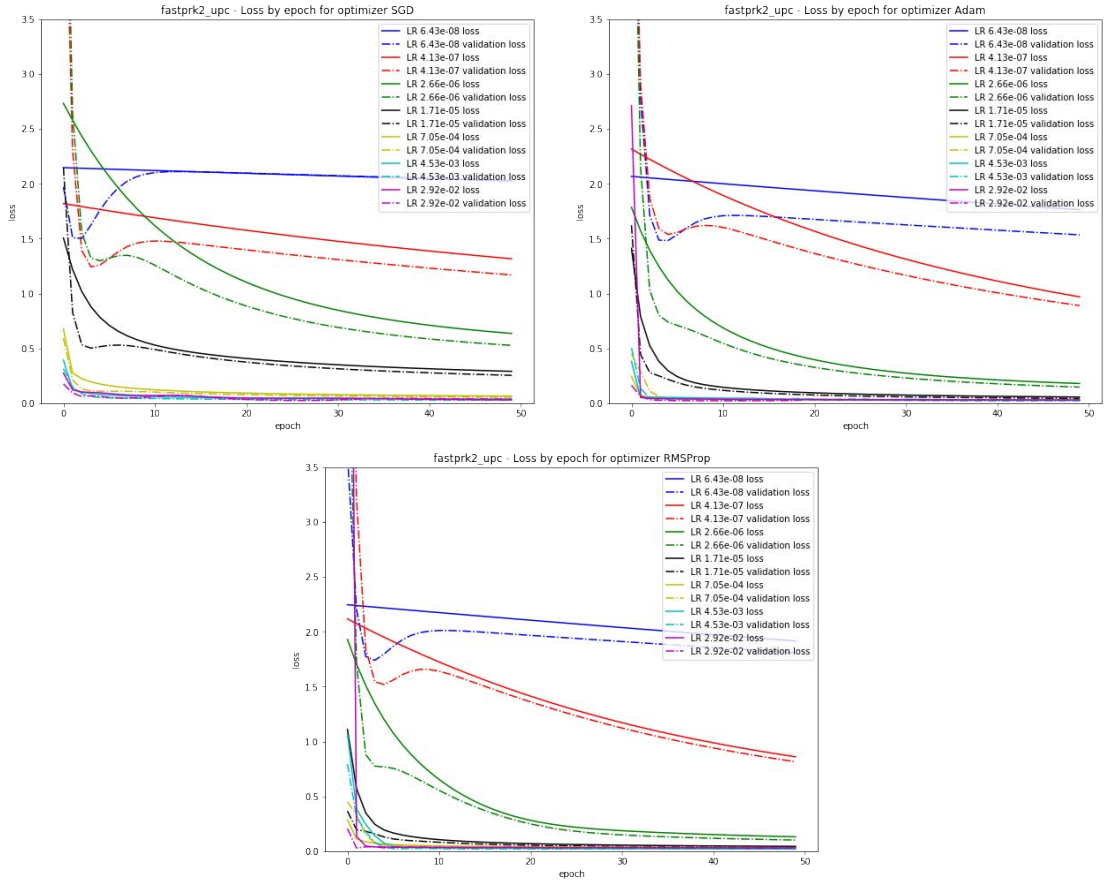


Table 7.5: Training loss and validation loss evolution throughout training epochs for the Barcelona scenario. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

SGD		Adam		RMSProp	
RMSE	LR	RMSE	LR	RMSE	LR
0.1054	0.089	0.1034	0.0022	0.0972	0.0038
0.1055	0.0614	0.1046	0.0018	0.1015	0.0022
0.1057	0.1072	0.105	0.0002	0.104	0.0031
0.1066	0.0095	0.1052	0.0001	0.104	0.0001
0.1073	0.0739	0.1058	0.0003	0.1048	0.0015

Table 7.6: Barcelona sorted validation RMSE values and learning rates by optimizer.

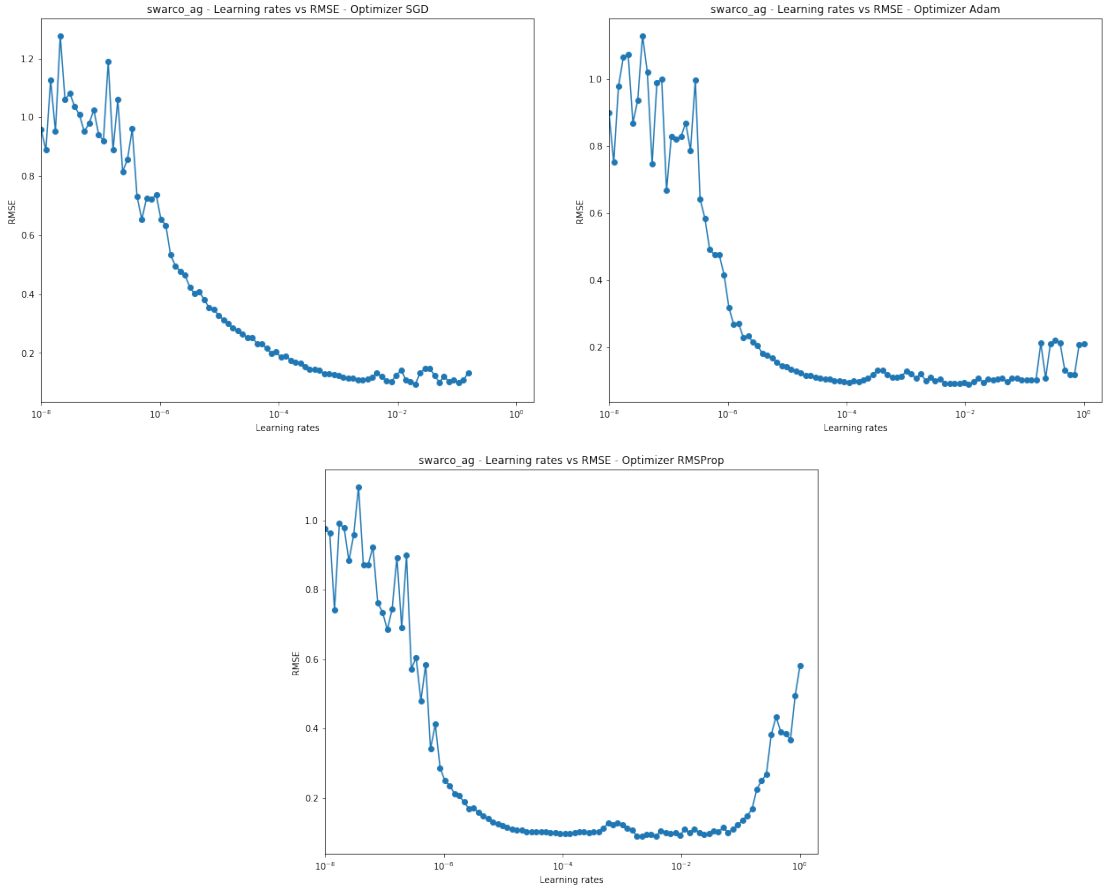


Figure 7.4: Wattens validation RMSE at different learning rates for the optimizers SGD with momentum (top left), Adam (top right), and RMSProp (bottom).

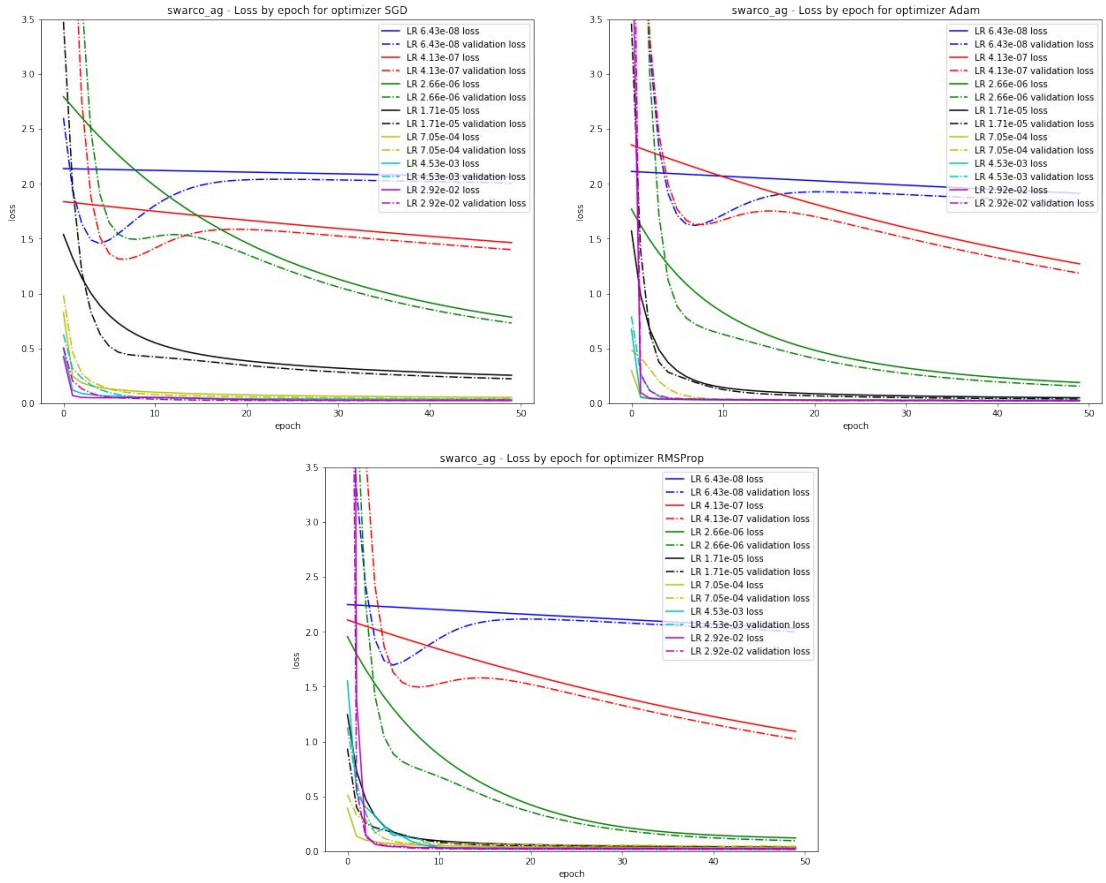


Table 7.7: Wattens training loss and validation loss evolution throughout training epochs. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

SGD		Adam		RMSProp	
RMSE	LR	RMSE	LR	RMSE	LR
0.0928	0.0201	0.0895	0.0115	0.0892	0.0038
0.0995	0.1072	0.0921	0.0066	0.0893	0.0022
0.1004	0.0509	0.0923	0.0079	0.0908	0.0018
0.1025	0.0739	0.0928	0.0055	0.0935	0.0095
0.1032	0.0079	0.0937	0.0045	0.0939	0.0031

Table 7.8: Wattens sorted validation RMSE values and learning rates by optimizer.

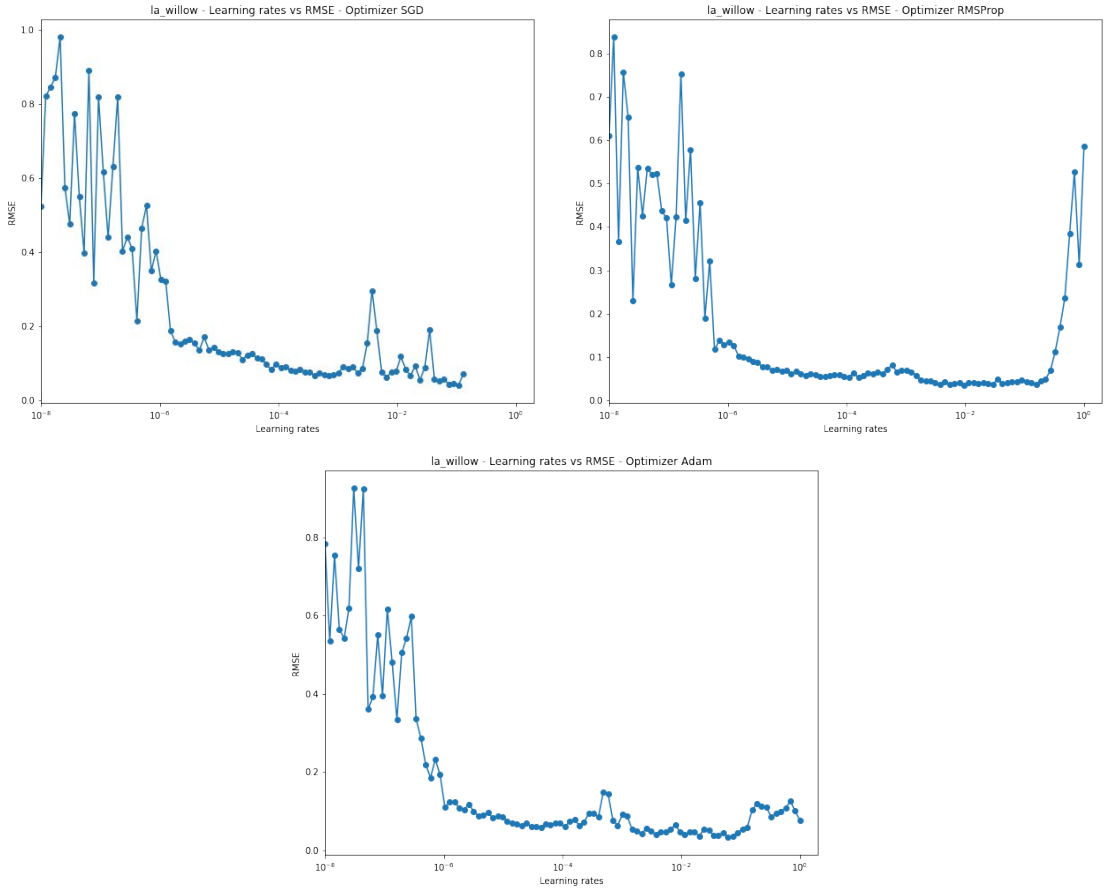


Figure 7.5: Los Angeles Standard sector validation RMSE at different learning rates for the optimizers SGD with momentum (top left), Adam (top right), and RMSProp (bottom).

The visualizations in Figure 7.5 present the validation RMSE values at different learning rates for the optimizers SGD with momentum, Adam and RMSProp for the Standard sector in Los Angeles. Similarly to previous scenarios, higher learning rates provide unstable values, especially for SGD and RMSProp. The Adam optimizer appears to be the most stable one.

Figure 7.9 presents the evolution of the losses as training progresses. In this case, validation loss is lower than training loss because the validation data include the period with works in the parking zone. This could indicate an adaptation of the model to changes in the parking behaviour caused by external factors. Similarly to previous scenarios, learning rates in the range of 10^{-4} to 10^{-2} converge fast and with the lowest validation loss values.

Table 7.10 presents the validation RMSE and the learning rate used to obtain it for each of the optimizers. Adam and RMSProp provide similar RMSE, which are lower than those from SGD with momentum. The values of the learning rates from Adam are closer

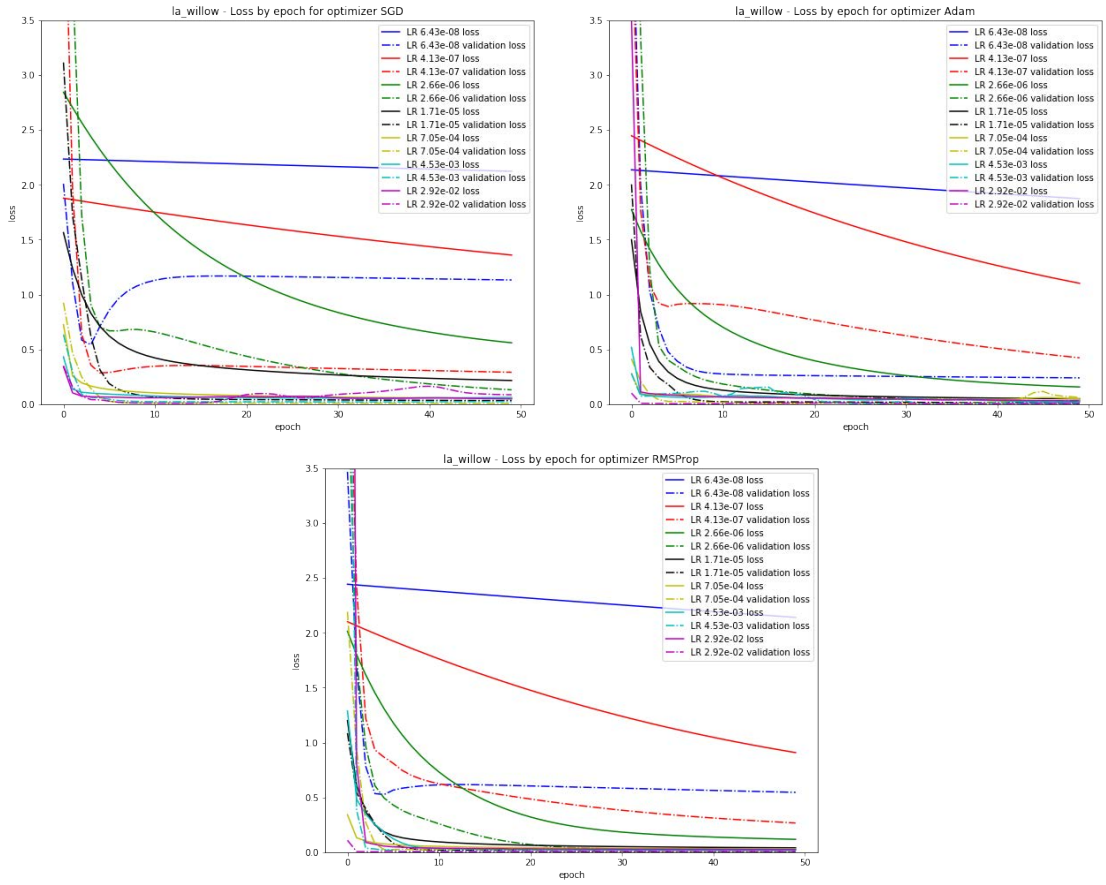


Table 7.9: Los Angeles training loss and validation loss evolution throughout training epochs. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

SGD		Adam		RMSProp	
RMSE	LR	RMSE	LR	RMSE	LR
0.0389	0.1072	0.0324	0.0614	0.0339	0.0095
0.0427	0.0739	0.0353	0.0201	0.0355	0.0055
0.044	0.089	0.0357	0.0739	0.0364	0.1556
0.0503	0.0509	0.0375	0.0351	0.0368	0.0038
0.054	0.0242	0.0383	0.0423	0.0371	0.0292

Table 7.10: Los Angeles sorted validation RMSE values and learning rates by optimizer.

than those from RMSProp, indicating better stability from Adam.

For the Riyadh scenario, the parking data of sector-3 have been used. Figure 7.6 again shows that Adam and RMSProp present more learning rate values with lower RMSE when compared to SGD with momentum. Also, similarly to previous scenarios, Adam seems to be more stable at higher learning rates when compared with the other two optimizers.

Loss curves throughout the training epochs are presented in Figure 7.11 for the Riyadh scenario. Again, models with learning rates in the range of 10^{-4} to 10^{-2} converge faster and achieve lower validation loss. In the RMSProp case, ten epochs are enough to reach convergence.

Table 7.12 presents the validation RMSE and the learning rate used to obtain it for each of the optimizers. In this scenario, RMSProp offers lower RMSE values at similar learning rates when compared to SGD and Adam.

In conclusion, after comparing SGD, Adam and RMSProp, Adam and RMSProp prove to be good candidates for training forecasting models with parking occupancy data using an MLP method. Although both provide very close results in the RMSE for the validation set, RMSProp is more stable than Adam when not taking into account high values for the learning rate. What is more, it provides equal or lower RMSE values than Adam in nearly all scenarios. In light of this evidence, this work uses the RMSProp optimizer in order to train the different MLP models with a learning rate value of between 10^{-4} and 10^{-2}.

7.4.1.2 Recurrent Neural Network methods: LSTM and GRU

Figure 7.7 presents the RMSE validation values at different learning rates for the Antwerp Pelikaanstraat scenario when using SGD with momentum, Adam and RMSProp optimizers for the LSTM and GRU models. At low values of the learning rate parameter, all optimizers present high and unstable RMSE values. At a learning rate of 10^{-6}, Adam and RMSProp achieve low values of RMSE that are reached by SGD at a learning rate of 10^{-4}. With a learning rate of 10^{-2}, RMSProp provides high RMSE values while SGD and Adam similarly achieve the lowest RMSE values. Both methodologies present very similar behaviours at the same values for the learning rates.

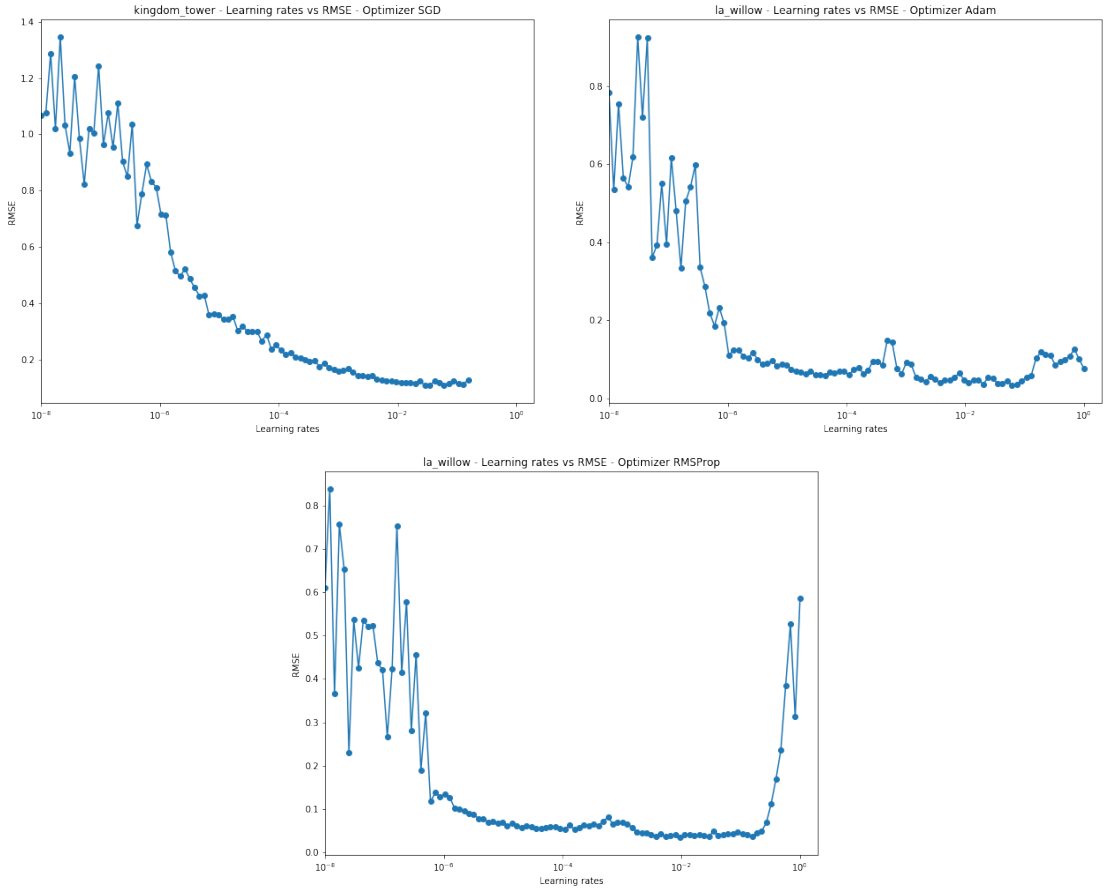


Figure 7.6: Riyadh sector-3 validation RMSE at different learning rates for the optimizers SGD with momentum (top left), Adam (top right), and RMSProp (bottom).

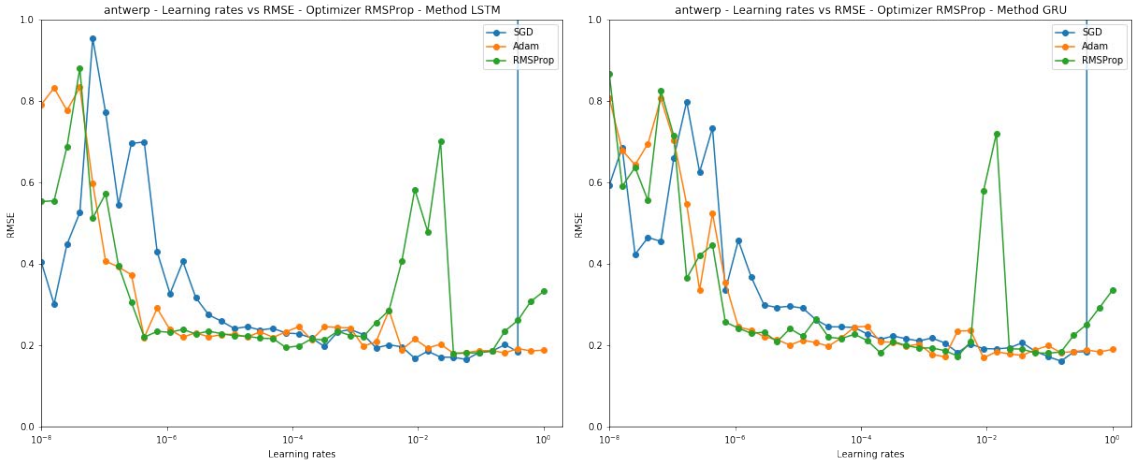


Figure 7.7: Antwerp validation RMSE at different learning rates for the optimizers SGD with momentum, Adam and RMSProp. Left corresponds to the LSTM models and right to the GRU models.

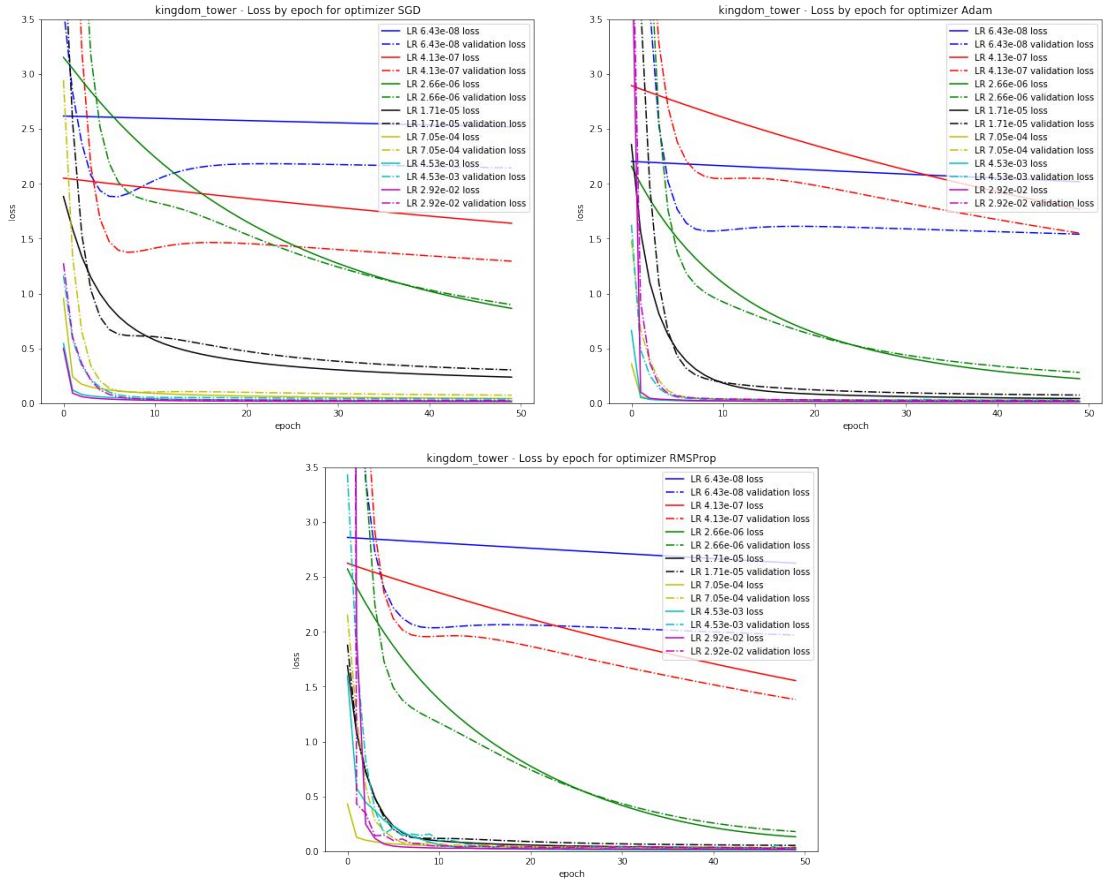


Table 7.11: Evolution of the training and validation losses throughout training epochs for the Riyadh scenario. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

SGD		Adam		RMSProp	
RMSE	LR	RMSE	LR	RMSE	LR
0.1083	0.0351	0.1049	0.0201	0.0999	0.0079
0.1096	0.0292	0.1057	0.0115	0.1007	0.0022
0.1105	0.0614	0.1064	0.0614	0.1039	0.0055
0.1133	0.1292	0.1066	0.0351	0.1044	0.0242
0.114	0.1072	0.1074	0.0018	0.105	0.0292

Table 7.12: Riyadh sorted validation RMSE values and learning rates by optimizer.

SGD				Adam				RMSProp			
LSTM		GRU		LSTM		GRU		LSTM		GRU	
RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR
0.1651	0.0588	0.1611	0.1512	0.1804	0.0367	0.169	0.0089	0.18	0.0367	0.1736	0.0035
0.1679	0.0089	0.1722	0.0943	0.1811	0.2424	0.1714	0.0022	0.1807	0.0588	0.1805	0.0943
0.1706	0.0367	0.1822	0.0035	0.1818	0.0588	0.1752	0.0367	0.1816	0.0943	0.1811	0.0588
0.1708	0.0229	0.1839	0.2424	0.186	0.6236	0.1781	0.0013	0.1856	0.1512	0.1818	0.0002
0.1825	0.0943	0.1840	0.3888	0.1867	0.0943	0.1787	0.0229	0.1944	0.0001	0.1834	0.1512

Table 7.13: Antwerp sorted validation RMSE values and learning rates by optimizer.

About the evolution of the loss curves during training, it is of interest to find at which learning rates the training and validation loss curves overlap throughout the epochs. A signal of overfitting occurs when the training loss is flat and the validation loss increases as the epochs grow. As the training continues, it is also of interest to observe the smoothness of the losses as well as the ups and downs, as this can be used to gain intuition about the local optima achieved by the model.

Figures 7.9 and 7.8 show the evolution of the loss throughout the training epochs for, respectively, the LSTM and GRU models that were trained with different learning rates for the Antwerp scenario. At first glance, one could believe that both methods are behaving equally, but the results show that LSTM obtains lower validation values with a wider range of learning rate values. More specifically, for the SGD with momentum in LTSM, it can be seen that learning rates in the range of 10^{-5} to 10^{-1} provide lower and more stable loss values, while for GRU, the lowest validation losses are achieved when using learning rates between 10^{-6} to 10^{-2} , but the validation loss became unstable when losses above 10^{-3} are used. For the Adam optimizer, the best range of values for LSTM is between 10^{-6} to 10^{-4}, with instability occurring at around 10^{-4} to 10^{-2}. For GRU, the lowest validation loss values are between 10^{-6} to 10^{-3}, with instability happening in the range of 10^{-3} to 10^{-2}. Lastly, for the RMSProp optimizer, LSTM shows the best validation loss values in the range between 10^{-6} to 10^{-1}, with instability occurring at around 10^{-3} to 10^{-2}, while for GRU the behaviour is similar but more stable than LSTM at 10^{-3} .

Table 7.13 presents the validation RMSE and their associated learning rate values for each optimizer used in the LSTM and GRU models, ordered from the lowest to the highest RMSE (only the top five are presented). Clearly, the SGD with momentum provides better RMSE values than the other two optimizers at very close learning rate values, this being an indication of its stability.

Figure 7.10 presents the validation RMSE values for the Barcelona scenario when using SGD with momentum, Adam and RMSProp optimizers at different learning rates for the LSTM and GRU models. Similarly to the Antwerp scenario, low learning rates provide high errors and are not stable. RMSE values reach the minimum values at 10^{-6} to 10^{-3} for the Adam and RMSProp optimizers, while SGD with momentum reaches

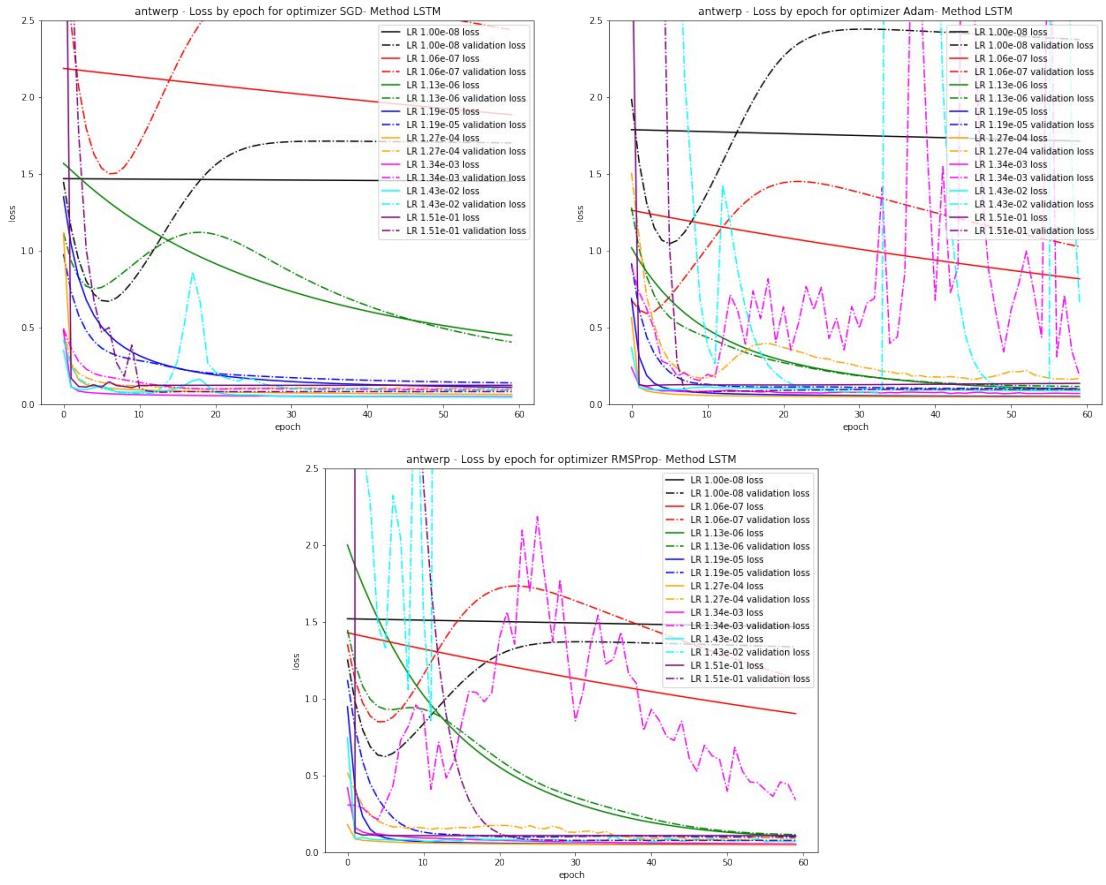


Figure 7.8: Evolution of the loss throughout the training epochs for the Antwerp scenario corresponding to the LSTM model. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

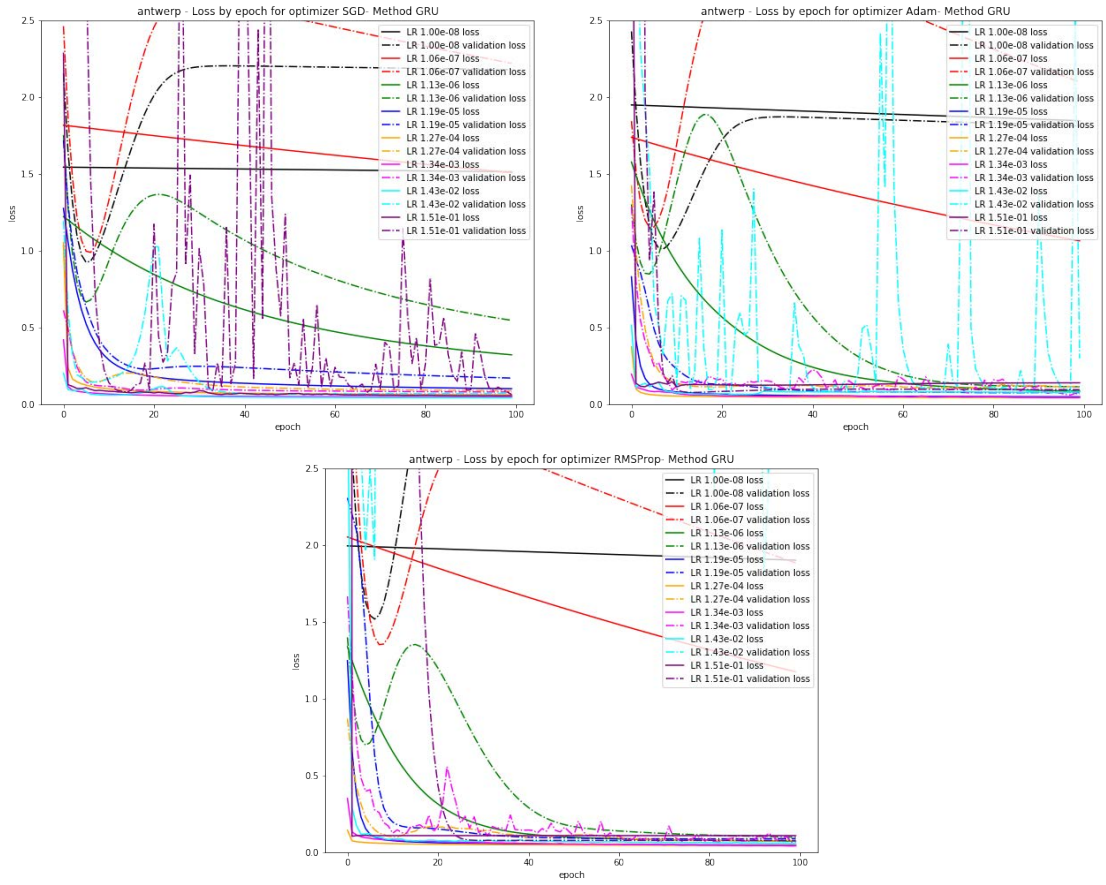


Figure 7.9: Evolution of the loss throughout the training epochs for the Antwerp scenario corresponding to the GRU models. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

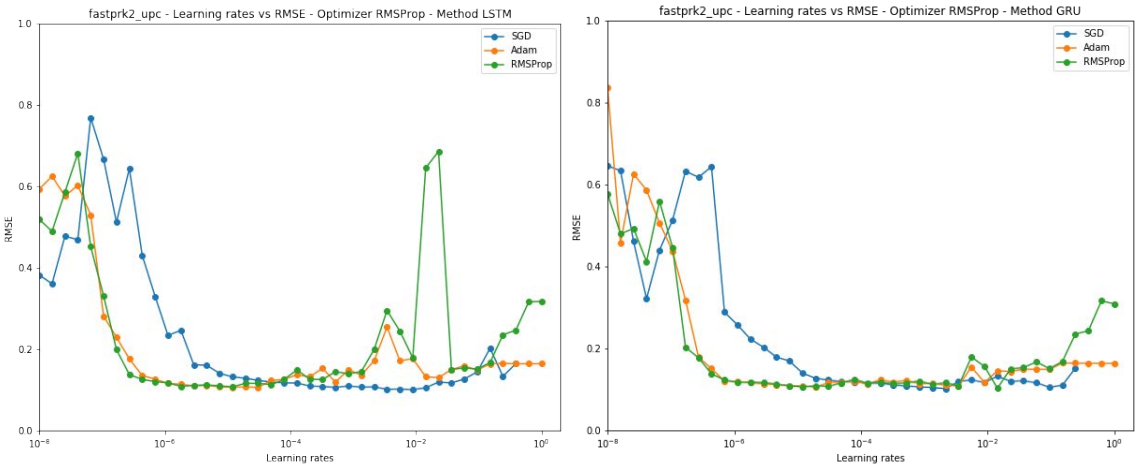


Figure 7.10: Barcelona scenario validation RMSE at different learning rates for the optimizers SGD with momentum, Adam and RMSProp. Left corresponds to LSTM and right to GRU.

the minimum RMSE values at learning rate values of between 10^{-4} to 10^{-1}. Both methods perform similarly, although Adam and RMSProp show more stable values in GRU around a learning rate value of 10^{-2}.

Training and validation loss curves are presented in Figures 7.11 and 7.12. For the LSTM, SGD with momentum presents lower and overlapping losses in the range between 10^{-6} to 10^{-2} and presents instability at higher learning rates. For GRU, the behaviour is similar but less unstable at higher learning rates. Both LSTM and GRU with Adam reach the lowest loss values, with learning rate values between 10^{-6} to 10^{-1}, and they present instability at around 10^{-2}. The RMSProp optimizer presents lower values with losses in the range between 10^{-6} to 10^{-4} for the LSTM method. On the other hand, the GRU method presents more stability at higher learning rates.

Table 7.14 shows the lowest five RMSE values obtained with each optimizer and the learning rates used to accomplish them for the LSTM and GRU methods. SGD with momentum seems to be the most adequate of the three, as it provides lower RMSE values at similar learning rates.

Figure 7.13 presents the validation RMSE for the LSTM and GRU models that were trained for the Wattens Inside sector with the optimizers SGD with momentum, Adam and RMSProp. In both methods, the optimizers behave in a similar fashion. The optimizers Adam and RMSProp reach lower validation RMSE values at learning rate values that are near 10^{-6}, while SGD with momentum reaches the lowest RMSE values past the 10^{-4} learning rate values. The most notable difference is that, at higher RMSE values (around 10^{-2}), the RMSE values for the RMSProp optimizer greatly increases for LSTM in comparison with GRU.

Loss curves from training and validation are presented in Figures 7.14 and 7.15. For both methods using the SGD with momentum optimizer, losses reach their lowest levels

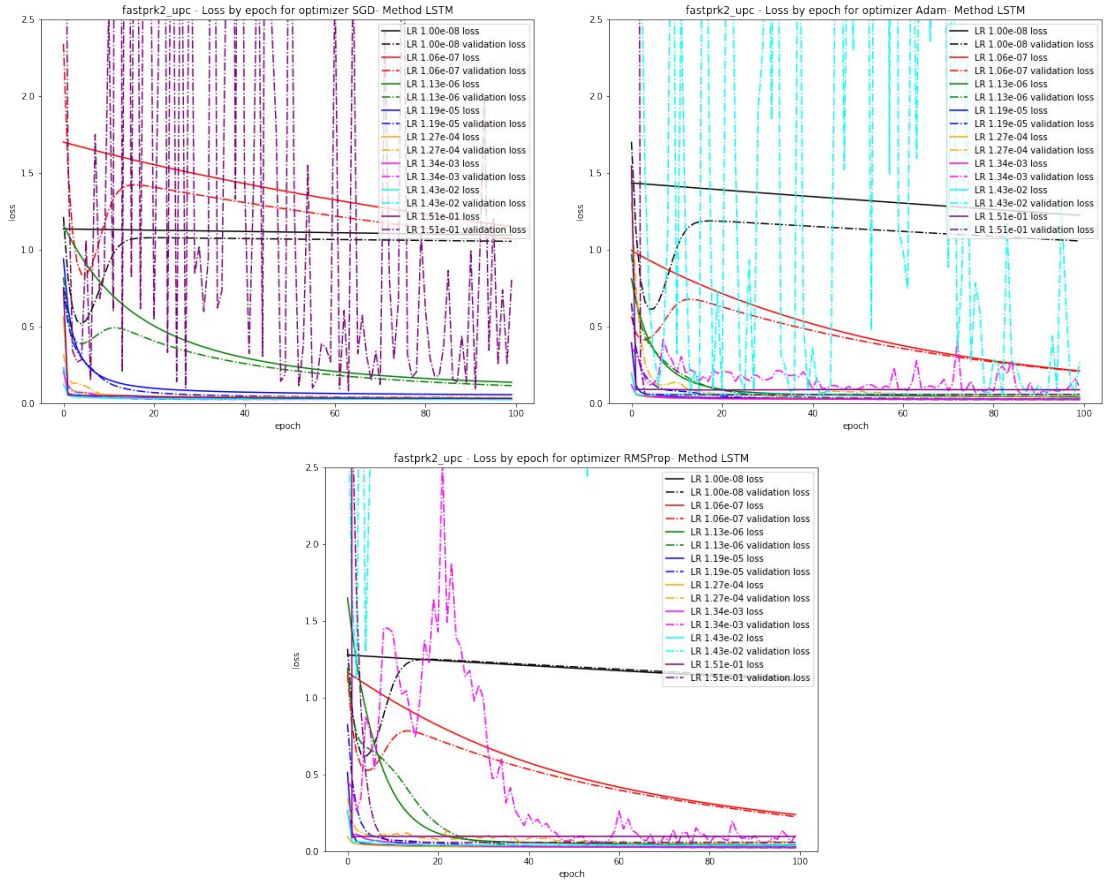


Figure 7.11: Evolution of the loss corresponding to the LSTM method throughout the training epochs for the Barcelona scenario. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

SGD				Adam				RMSProp			
LSTM		GRU		LSTM		GRU		LSTM		GRU	
RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR
0.1008039	0.0088862	0.10188	0.00215	0.10657	0.00003	0.1056	0.00002	0.107654	0.00001	0.10313	0.01425
0.101714	0.0034551	0.10456	0.00134	0.106781	0.00001	0.1085	0.00001	0.108910	0.000002	0.10608	0.00001
0.1022197	0.005541	0.10505	0.09427	0.107445	0.000012	0.1089	0.00346	0.110213	0.000007	0.10749	0.00003
0.1054118	0.014251	0.10616	0.00084	0.107666	0.000007	0.1091	0.00001	0.111487	0.000003	0.1084	0.00346
0.1063732	0.0005223	0.10825	0.00052	0.110158	0.000003	0.1098	0.00215	0.112470	0.000005	0.10884	0.00001

Table 7.14: Barcelona top five validation RMSE values with corresponding learning rates for each optimizer.

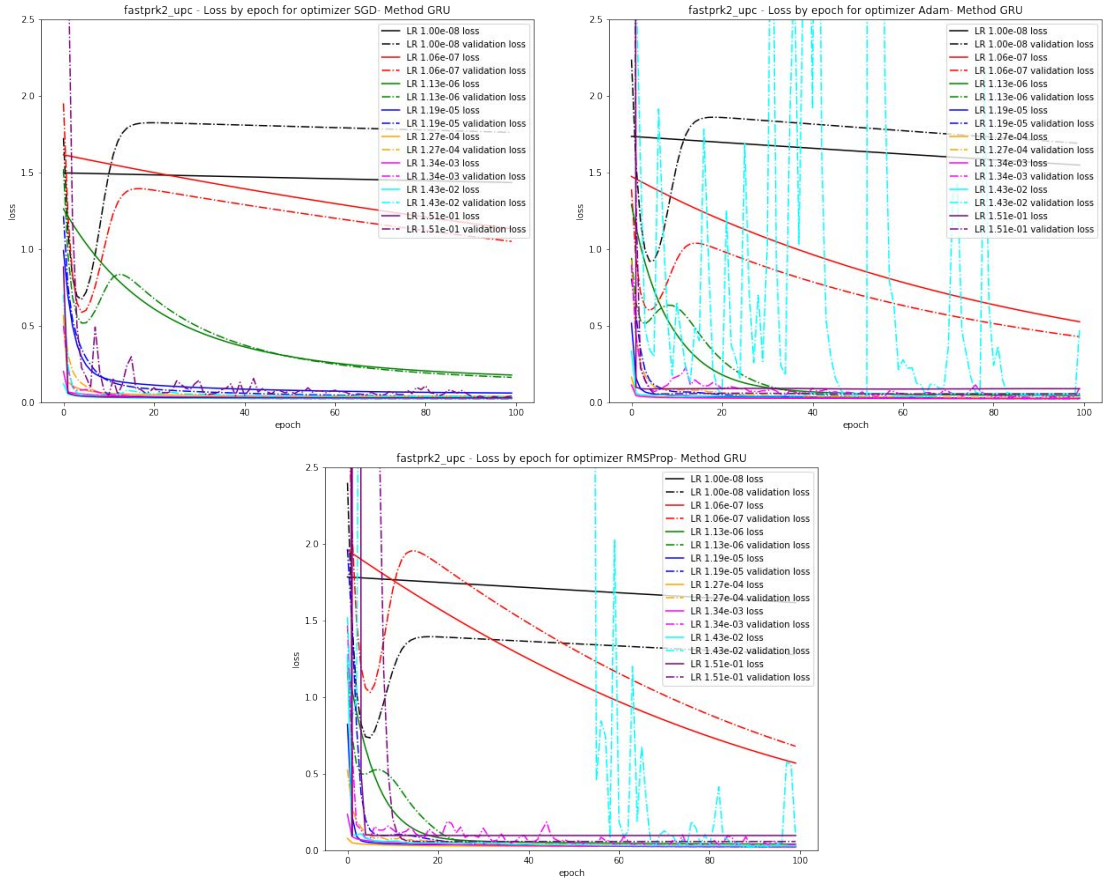


Figure 7.12: Evolution of the loss corresponding to the GRU method throughout the training epochs for the Barcelona scenario. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp. Top corresponds to the LSTM method and bottom to GRU.

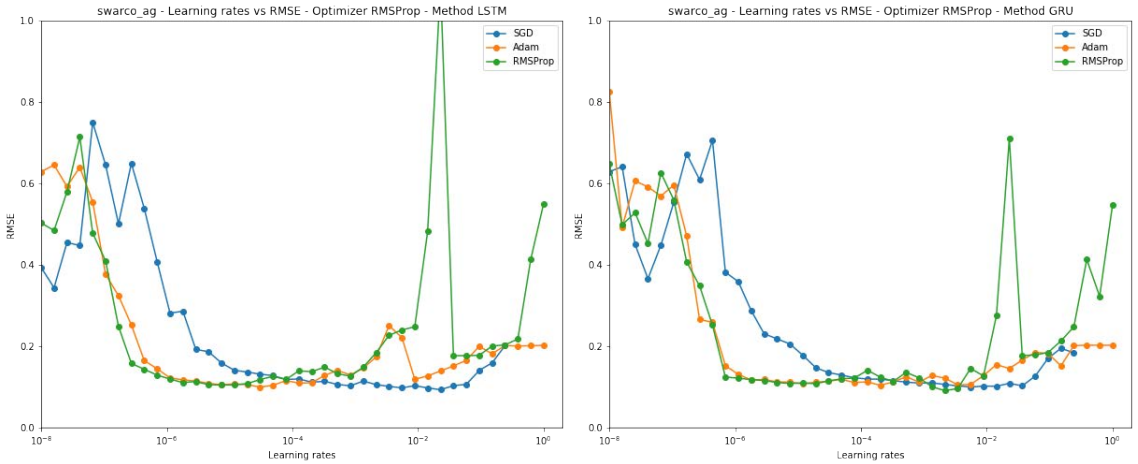


Figure 7.13: Validation RMSE for the Wattens scenario at different learning rates for the optimizers SGD with momentum, Adam and RMSProp. Left corresponds to the LSTM method and right to GRU.

SGD				Adam				RMSProp			
LSTM		GRU		LSTM		GRU		LSTM		GRU	
RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR
0.0953	0.0089	0.10002	0.00554	0.1083	0.000019	0.10404	0.0002	0.1009	0.000012	0.09098	0.00215
0.0988	0.0055	0.10193	0.01425	0.1095	0.000007	0.1055	0.00346	0.1053	0.000031	0.09614	0.00346
0.0988	0.0035	0.10211	0.00889	0.1099	0.000012	0.10611	0.00554	0.1058	0.000007	0.10063	0.00134
0.1006	0.0143	0.1028	0.00346	0.1115	0.000005	0.10746	0.00001	0.1068	0.000049	0.10796	0.00002
0.1013	0.0022	0.10289	0.03665	0.1128	0.000079	0.11088	0.00008	0.1072	0.000005	0.10849	0.00001

Table 7.15: Wattens top five validation RMSE values with corresponding learning rates for each optimizer.

when using learning rates in the range between 10^{-5} to 10^{-3}; and learning becomes unstable when using a loss near 10^{-1}. For the Adam optimizer, both methods perform similarly with good results at the learning rate range between 19^{-6} to 10^{-3}, and unstable validation loss values occur at learning rates with values of around 10^{-3}. Lastly, for the RMSProp optimizer, learning rates in the range between 10^{-6} to 10^{-4} are the ones that work best for both optimizers, but they present instability at learning rates of around 10^{-2}.

Table 7.15 shows the lower five RMSE values obtained with each optimizer and the learning rates used for the LSTM and GRU methods. For LSTM, SGD with momentum seems to be the most adequate of the optimizers considered, as it provides lower RMSE values at similar learning rates. On the other hand, the optimizer that presents better results for GRU is the RMSProp, as it presents 0.01 less validation RMSE when compared to SGD and Adam. Also, its validation RMSE values are accomplished with learning rates of similar value.

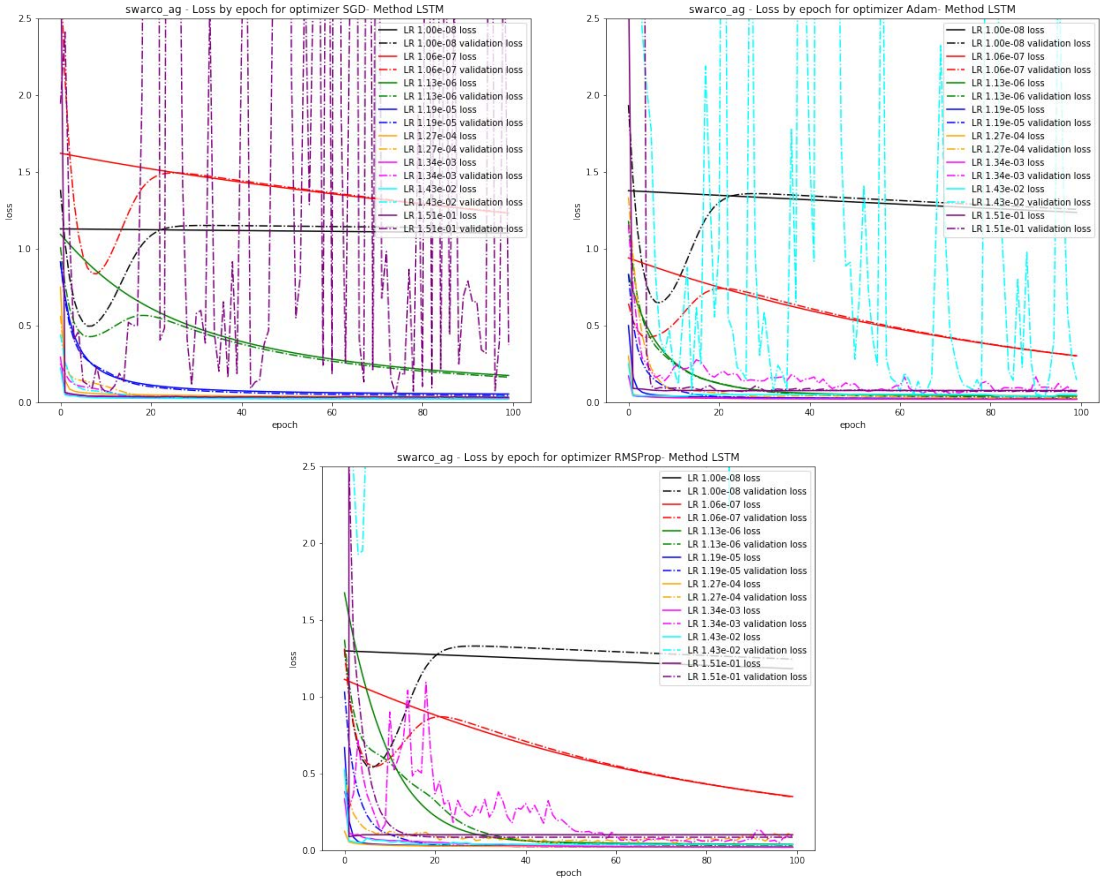


Figure 7.14: Evolution of the loss corresponding to the LSTM models throughout the training epochs for the Wattens scenario. From top (left and right) to bottom, the optimizers are: SGD with momentum, Adam and RMSProp.

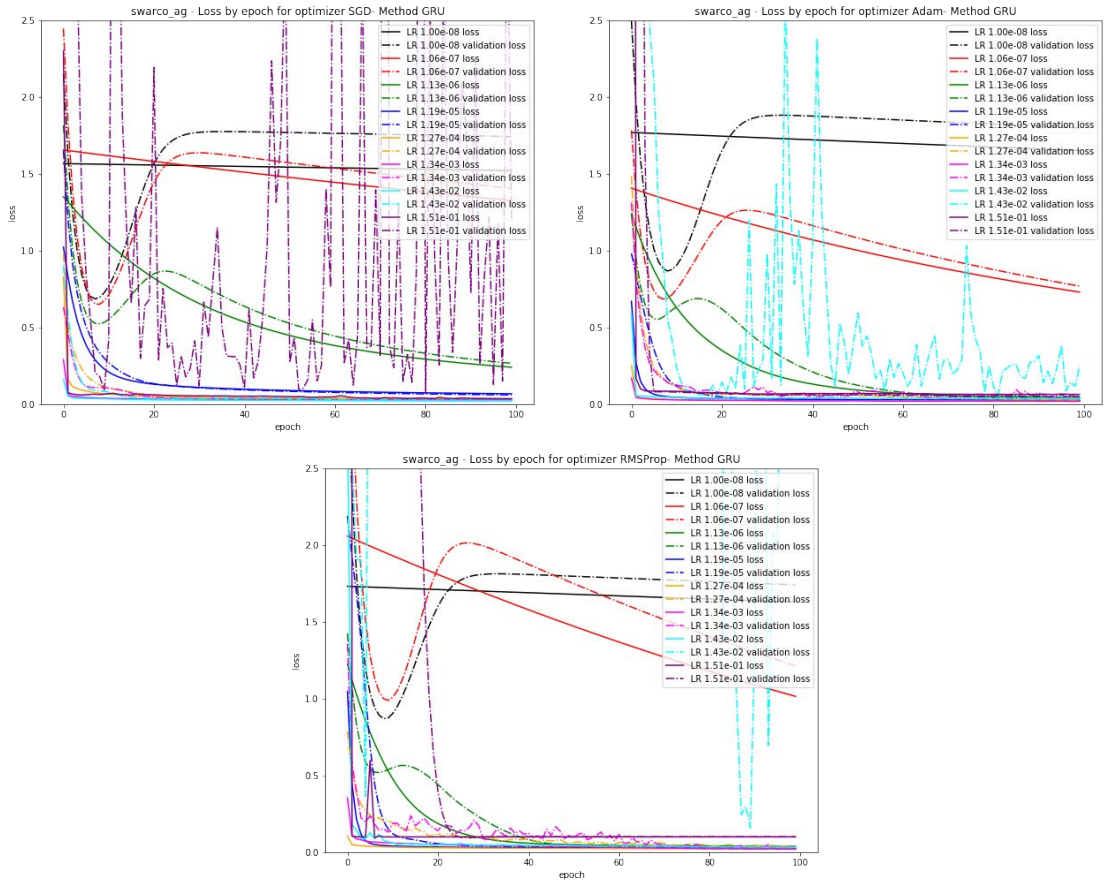


Figure 7.15: Evolution of the loss corresponding to the GRU models throughout the training epochs for the Wattens scenario. From top (left and right) to bottom, the optimizers are: SGD with momentum, Adam and RMSProp.

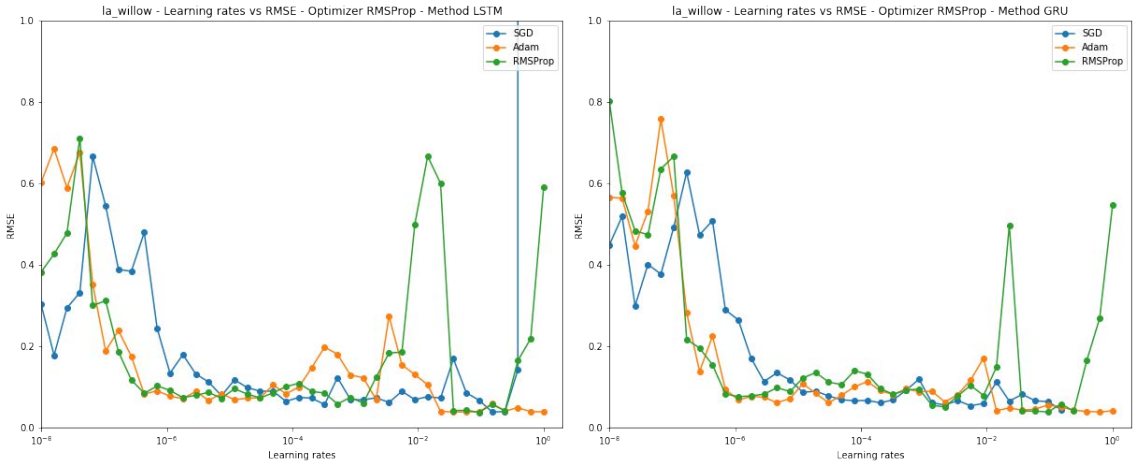


Figure 7.16: Los Angeles scenario validation RMSE at different learning rates for the optimizers SGD with momentum, Adam and RMSProp. Left corresponds to the LSTM method and right to GRU.

Figure 7.16 shows the validation RMSE for the optimizers SGD with momentum, Adam and RMSProp for the Los Angeles Standard sector with the LSTM and GRU methods. Similarly to the other scenarios, both methods presents similar RMSE behaviours across the different learning rates. Adam and RMSProp provide lower and stable validation RMSE values at the range between 10^{-6} to 10^{-3} while SGD with momentum accomplish the lowest and stable RMSE values at the range between 10^{-5} to 10^{-1}. Both methods present high RMSE values of around 10^{-2} and at the highest values of the learning rates considered.

Loss curves from training and validation are presented in Figures 7.17 and 7.18. SGD with momentum for the LSTM method presents smooth curves, where the best learning rates are located in the range of values between 10^{-6} to 10^{-1}. GRU presents a similar behaviour, but at around 10^{-2} it presents signals of instability. For the Adam optimizer, LSTM's best range of values is between 10^{-6} to 10^{-1}, with instabilities when using a learning rate of around 10^{-2} for the first 30 epochs. The GRU models present a similar behaviour as LSTM, but the instabilities at 10^{-2} are more noticeable. Lastly, for the RMSProp optimizer, models from both methods behave similarly, with the best learning rates in the range of 10^{-6} to 10^{-3}. Also, both methods present unstable learning when using a learning rate of around 10^{-2}.

Table 7.16 shows the lower five RMSE values obtained with each optimizer and the learning rates used. For LSTM, the best optimizer (lower validation RMSE) is the Adam optimizer. For GRU, it is possible to consider Adam as the best one, but RMSProp provides a very similar performance. Notice that LSTM and GRU provide very close RMSE values at the same learning rates for the RMSProp optimizer.

The last scenario corresponds to Sector-1 in Riyadh. Figure 7.19 shows the validation RMSE for models from the the LSTM and GRU methods when using the optimizers

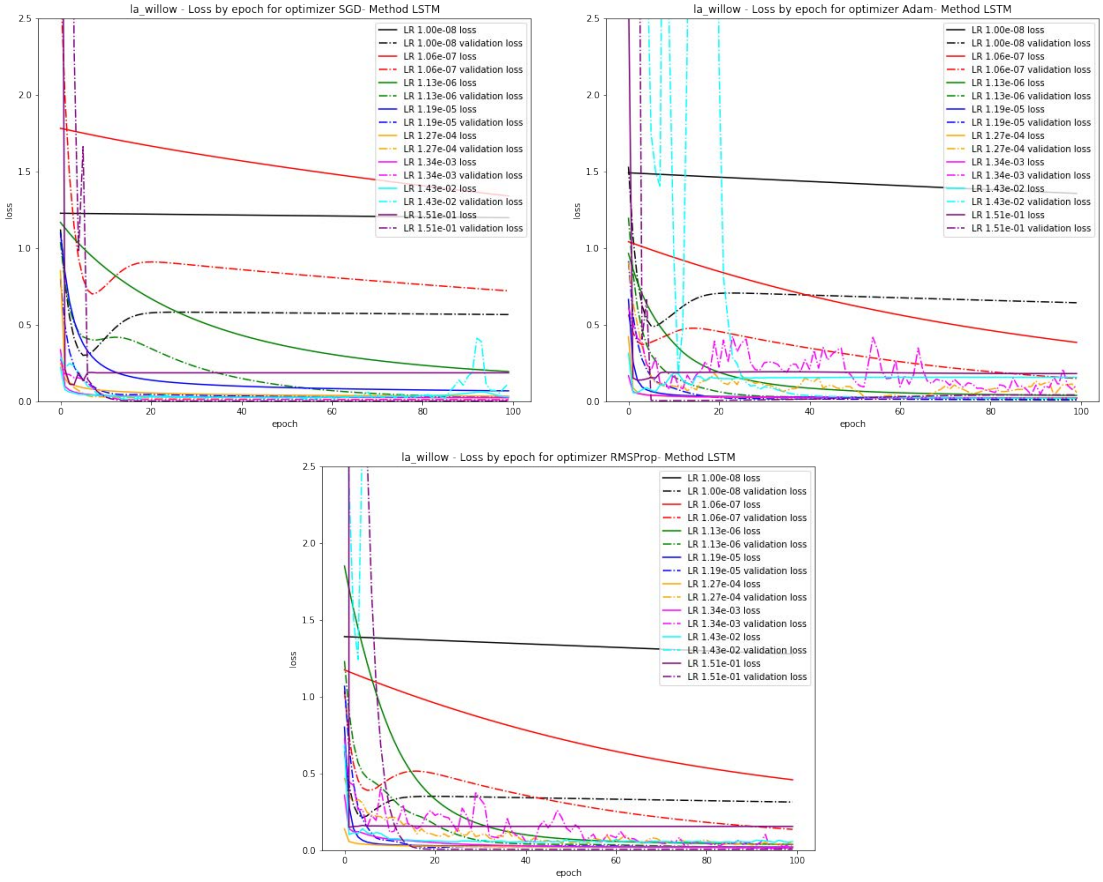


Figure 7.17: Evolution of the loss throughout the training epochs for the Los Angeles sector Standard for the LSTM methods. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

SGD				Adam				RMSProp			
LSTM		GRU		LSTM		GRU		LSTM		GRU	
RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR
0.0387	0.1512	0.0446	0.1512	0.0386	0.0367	0.0386	0.6236	0.0379	0.0943	0.0385	0.0943
0.0395	0.2424	0.0539	0.0055	0.0387	0.0588	0.04	0.3888	0.0417	0.2424	0.0405	0.2424
0.0569	0.0003	0.0557	0.0022	0.039	1.	0.0415	1.	0.0419	0.0367	0.04090	0.0588
0.0621	0.0035	0.0596	0.0089	0.0392	0.0943	0.0421	0.0143	0.0428	0.0588	0.041	0.0367
0.0648	0.0001	0.0613	0.0002	0.0394	0.6236	0.043	0.2424	0.0573	0.1512	0.0507	0.0022

Table 7.16: Top five validation RMSE values for the Los Angeles scenario, with their corresponding learning rates for each optimizer.

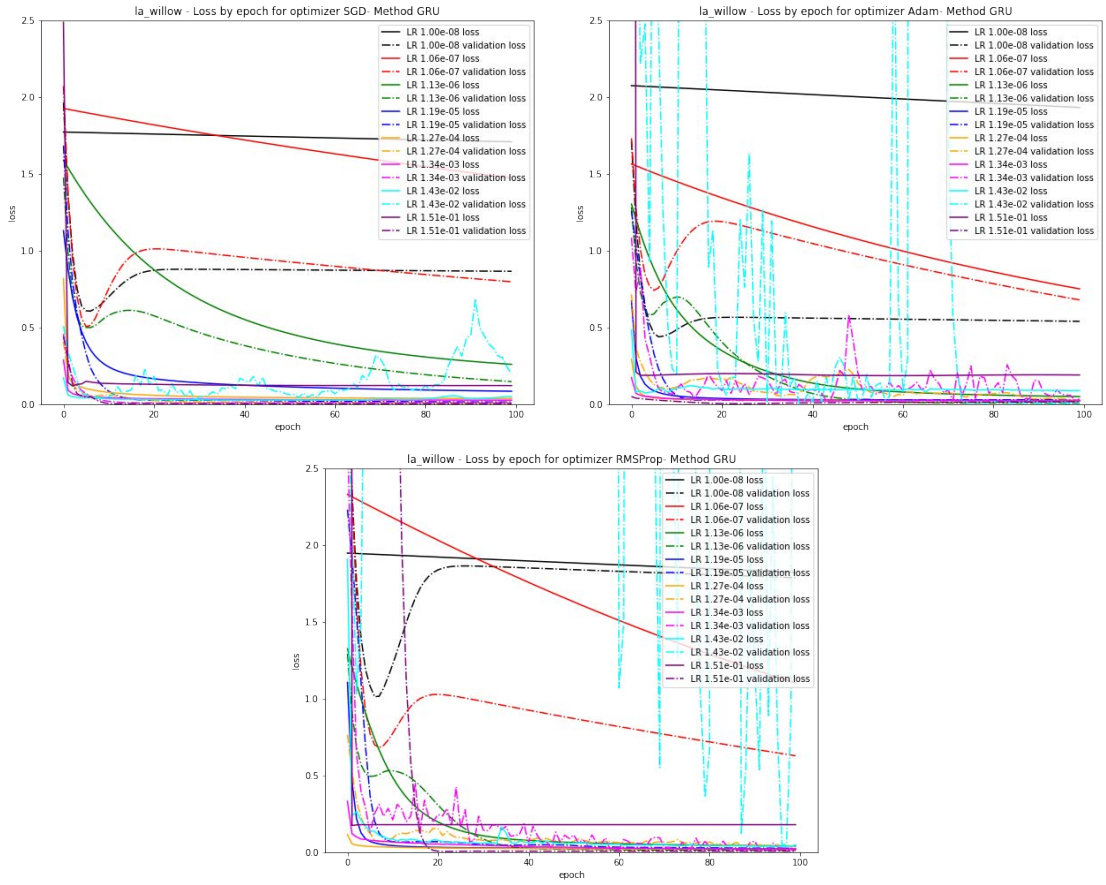


Figure 7.18: Evolution of the loss throughout the training epochs for the Los Angeles sector Standard for the GRU methods. Top left is the SGD with momentum optimizer, top right Adam, and bottom RMSProp.

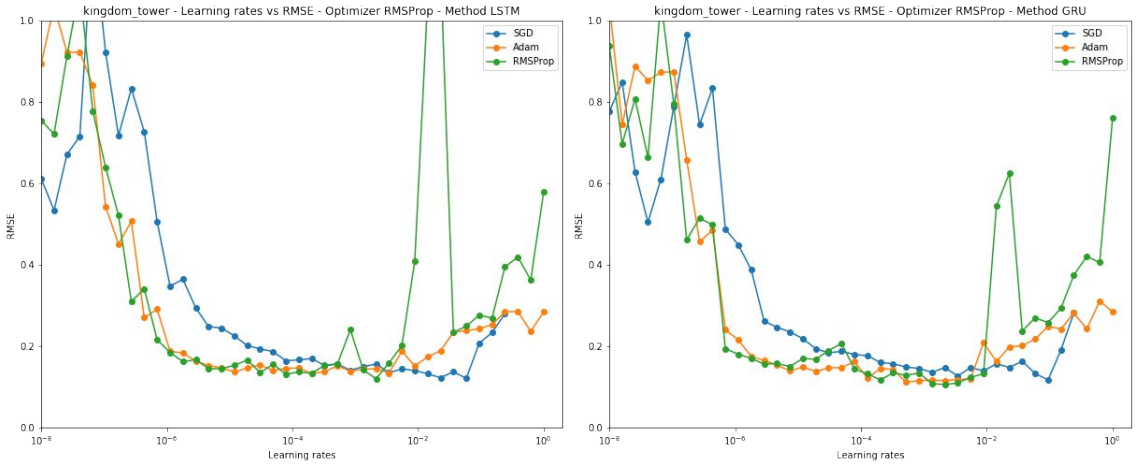


Figure 7.19: Riyadh Sector-1 validation RMSE at different learning rates for the optimizers SGD with momentum, Adam and RMSProp for the methods LSTM (left) and GRU (right).

SGD with momentum, Adam and RMSProp at forty different learning rates that are equally spaced between 10^{-8} to 10^{0}. For both methods, all three optimizers behave similarly in providing high RMSE values at learning rates below 10^{-6} and above 10^{-1}. All three optimizers similarly reach the lowest RMSE values, but Adam and RMSProp accomplish this in the range between 10^{-5} to 10^{-3} while SGD does so at learning rates between 10^{-3} to 10^{-1}. In both methods, RMSProp presents a high validation RMSE value at learning rates of around 10^{-2}. Lastly, SGD with momentum diverges when the learning rate is closer to 1.

Loss curves from training and validation losses are presented in Figures 7.20 and 7.21. The optimizer SGD with momentum shows smooth curves when using the LSTM models, with the best learning rates in the range between 10^{-5} to 10^{-1}. In the case of GRU models, SGD with momentum works better at the learning rate range between 10^{-5} to 10^{-2}. In the case of the Adam optimizer, the lowest validation RMSE values are found in the range between 10^{-6} to 10^{3} for both methods. Lastly, for the RMSProp optimizer, GRU is more stable than LSTM at high learning rates (10^{-3} or higher), and both methods show that, at around at 10^{-2}, the training diverges. The best range of values for the learning rates is between 10^{-5} to 10^{-3}.

Table 7.17 shows the lowest five RMSE values obtained with each optimizer and the learning rates used for the LSTM and GRU methods. For the LSTM method, the best suited optimizer for the data of Riyadh is SGD with momentum, as it provides the lowest validation RMSE values at similar learning rates while Adam is the one that performs the worst. On the other hand, the lowest validation RMSE values for the GRU method come from the RMSProp optimizer at learning rates of similar value. Notice that the top RMSE values for both methods have learning rate values in common; for example, LSTM and GRU achieve the lowest RMSE with RMSProp when using a learning rate value of 0.00215.

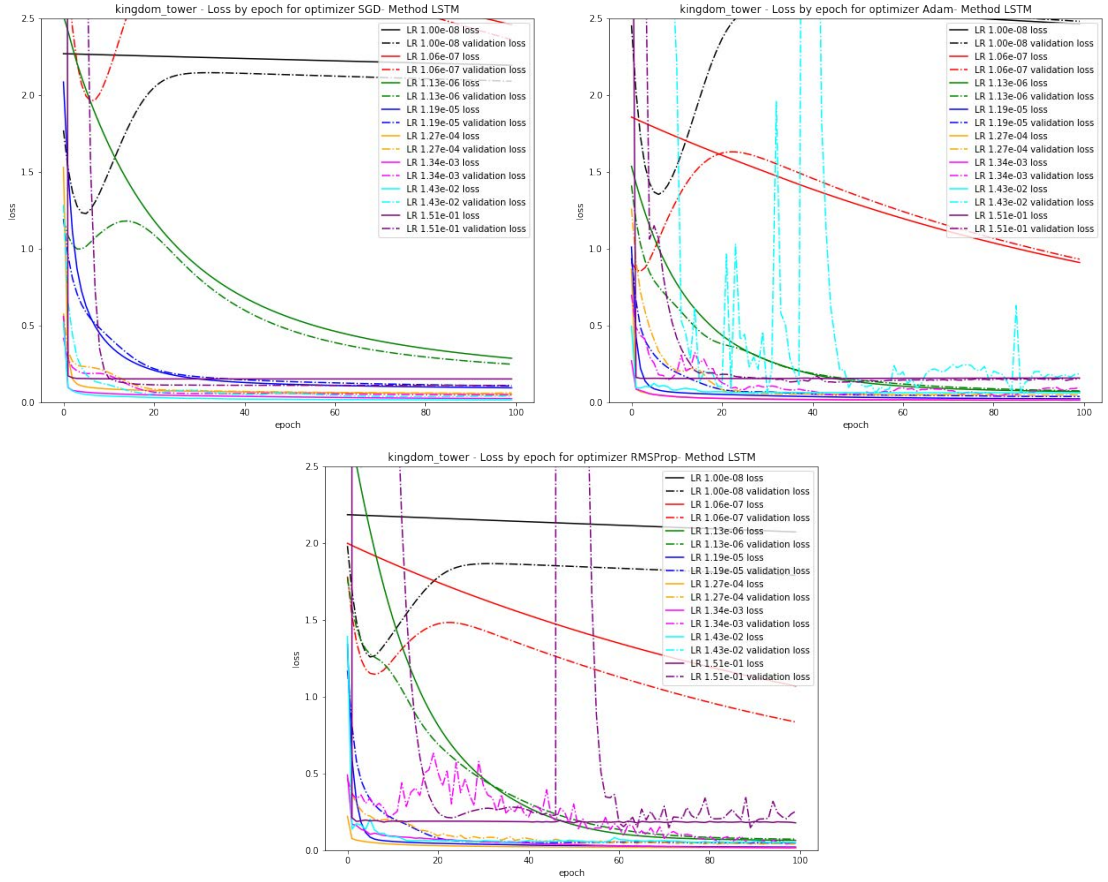


Figure 7.20: Training and validation loss curves corresponding to the LSTM models at different learning rates for the optimizers SGD with momentum optimizer (top left), Adam (top right), and RMSProp (bottom).

SGD				Adam				RMSProp			
LSTM		GRU		LSTM		GRU		LSTM		GRU	
RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR	RMSE	LR
0.12133	0.05878	0.11738	0.09427	0.1334	0.0002	0.11173	0.00052	0.12047	0.00215	0.10608	0.00215
0.1229	0.02285	0.12718	0.00346	0.13446	0.00346	0.11545	0.00215	0.13171	0.00008	0.10832	0.00134
0.13223	0.01425	0.13322	0.05878	0.13689	0.00005	0.11551	0.00084	0.13338	0.0002	0.10977	0.00346
0.1351	0.00346	0.13592	0.00134	0.13765	0.00033	0.11636	0.00134	0.13561	0.00003	0.11717	0.0002
0.1373	0.03665	0.1394	0.00889	0.13906	0.00084	0.11826	0.00346	0.13696	0.00013	0.12397	0.00554

Table 7.17: Top five validation RMSE values for Sector-1 in Riyadh for the LSTM and GRU methods. For each RMSE value, the learning rate for obtaining it is also provided.

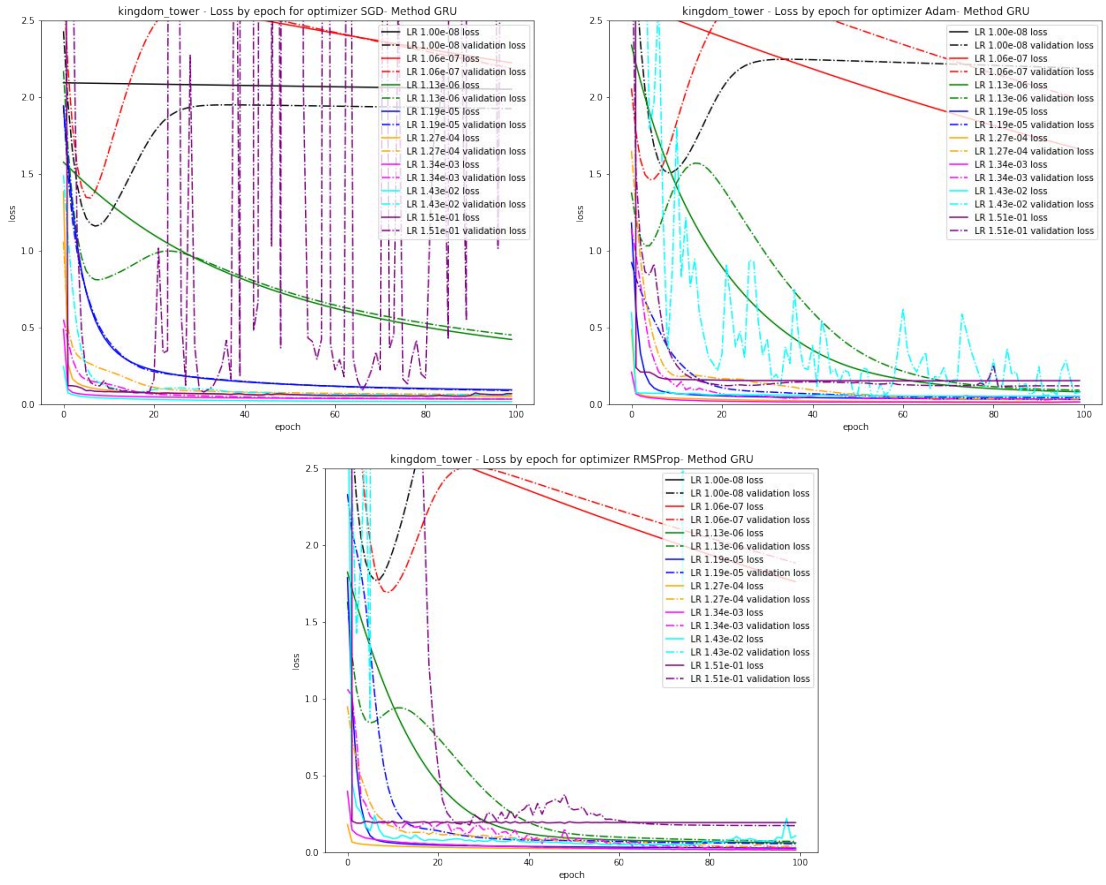


Figure 7.21: Training and validation loss curves corresponding to the GRU models at different learning rates for the optimizers SGD with momentum optimizer (top left), Adam (top right), and RMSProp (bottom).

In conclusion, for the the LSTM method, the SGD with momentum is the optimizer that provides the lowest RMSE values when compared to Adam and RMSProp. Furthermore, the best range of values for the learning rates is between 10^{-3} to 10^{-1}. For the GRU method, RMSProp slightly outperforms the results of SGD with momentum. In general, the best range for learning rates has been found to be between 10^{-5} to 10^{-2}.

7.4.2 Model capacity

The aim of this experiment is to obtain the effect of the model capacity (number of layers and number of neurons at each hidden layer) on the validation RMSE and study the loss curves.

For this experiment, the optimizers and learning rates used are the ones that provided the best score in the optimizer experiments (Section 7.4.1). The other hyperparameters remain the same as they are explained at the start of Section 7.4. The number of hidden layers used varies from one to six and the number of neurons from one to 128, using powers of two between them. For each number of layers considered, each of the neurons considered is used, thus obtaining a total of forty-eight different models.

7.4.2.1 MLP

Figure 7.22 presents the validation RMSE for the Pelikaanstraat sector in Antwerp for different combinations of hidden layers and neurons considered. Clearly, a low model capacity (one hidden layer or less than eight neurons) presents a higher number of errors. As the number of neurons increases, the RMSE is maintained or improved only by those models with higher numbers of hidden layers. This can be seen in the one-layer models, which present an increasing validation RMSE value when the number of neurons is higher than 32.

The behaviour of the losses (Figure 7.23) shows how the model capacity increasing leads to the risk of overfitting also increasing, as the gap between training and validation losses becomes larger (taking into account that the goal of regularization techniques is to prevent this effect). Loss curves show that a high number of neurons with a small number of layers presents unstable behaviour for the validation loss; while, on the other hand, a small number of layers and a small number of neurons offer stability and convergence to nearly the same loss values. As the number of layers increases, models with lower numbers of neurons start to have unstable losses and models with a high number of neurons become stable. Models in between present stability around all considered numbers of hidden layers.

In order to more closely observe the model capacities, Table 7.18 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for the different numbers of hidden layers considered. Thirty two seems to be the predominant number of neurons at a low number of hidden layers; but as the number of hidden layers increases, 128 offers the best RMSE values.

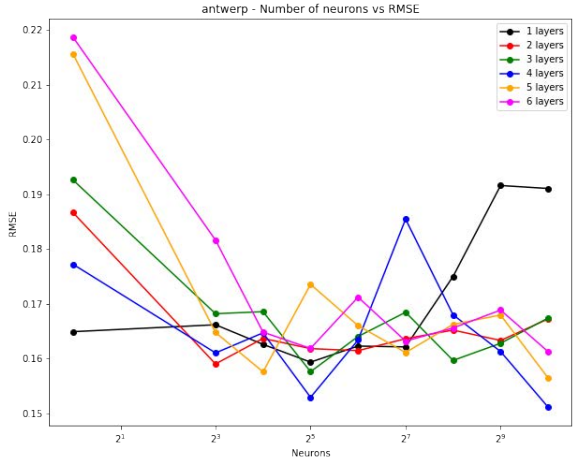


Figure 7.22: Antwerp validation RMSE comparison of models with different numbers of layers and neurons.

One hidden layer		Two hidden layers		Three hidden layers		Four hidden layers		Five hidden layers		Six hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.1593	32	0.159	8	0.1576	32	0.1511	1024	0.1564	1024	0.1612	1024
0.1621	128	0.1614	64	0.1596	256	0.1529	32	0.1576	16	0.1618	32
0.1623	64	0.1618	32	0.1628	512	0.161	8	0.1611	128	0.1632	128
0.1626	16	0.1633	512	0.164	64	0.1613	512	0.1647	8	0.1648	16
0.1649	1	0.1636	128	0.1674	1024	0.1634	64	0.166	64	0.1656	256

Table 7.18: Antwerp top validation RMSE values for different numbers of layers.

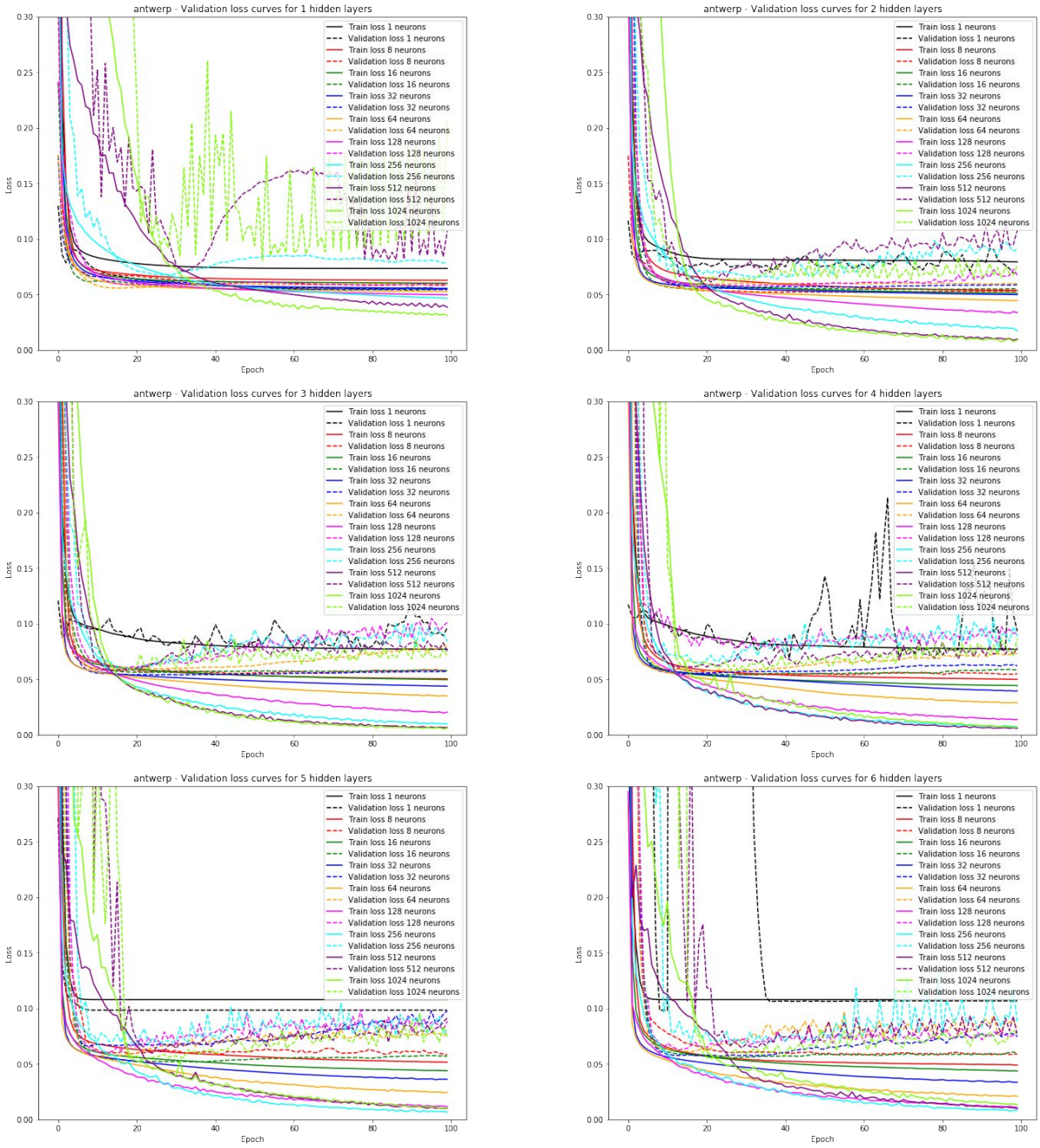


Figure 7.23: Antwerp losses as the number of hidden layers increases.

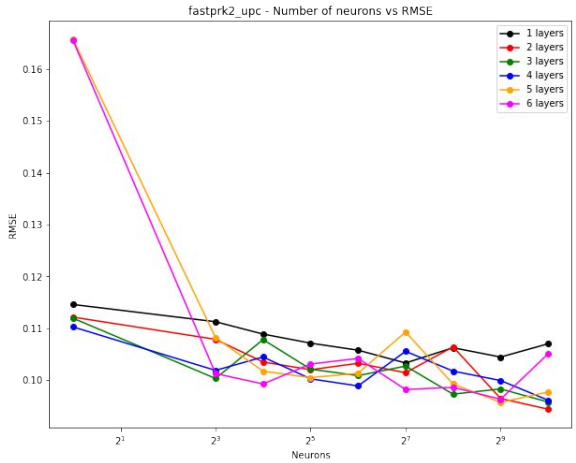


Figure 7.24: Barcelona validation RMSE comparison of models with different numbers of layers and neurons.

One hidden layer		Two hidden layers		Three hidden layers		Four hidden layers		Five hidden layers		Six hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.10334	128	0.09440	1024	0.0958	1024	0.0962	1024	0.0958	512	0.0964	512
0.10449	512	0.0965	512	0.0973	256	0.0989	64	0.0978	1024	0.0982	12
0.1058	64	0.1015	128	0.9834	512	0.1000	512	0.0993	256	0.0987	256
0.10628	256	0.102	32	0.1004	8	0.1003	32	0.1005	32	0.0993	16
0.10704	1024	0.1033	64	0.1009	64	0.1018	256	0.1014	64	0.1013	8

Table 7.19: Barcelona top validation RMSE values for different numbers of layers.

Figure 7.24 presents the validation RMSE for the Barcelona scenario when considering different numbers of hidden layers and neurons. The models with a high number of layers but a low number of neurons are the ones that present higher error rates; but as the number of neurons grows, the error is reduced. The visualization shows that as the number of neurons increases, the validation RMSE is reduced.

Figure 7.25 shows the loss curves for the models with one to six hidden layers. Observe that models with higher capacity overfit the training data and that as the number of hidden layers increases, the validation loss curves of the models become more unstable. The behaviour of the one-neuron models is interesting, namely in that as the number of layers increases, the validation loss becomes more and more unstable until at one point (five hidden layers) it converges at the same value for that point and any subsequent numbers of hidden layers.

Table 7.19 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for the different numbers of hidden layers considered. The number of neurons that presents the lowest RMSE values is 1024, followed by 512.

For the Wattens scenario (Inside sector), Figure 7.26 shows a similar behaviour of the validation RMSE for different hidden layers and neurons combinations but the decrease

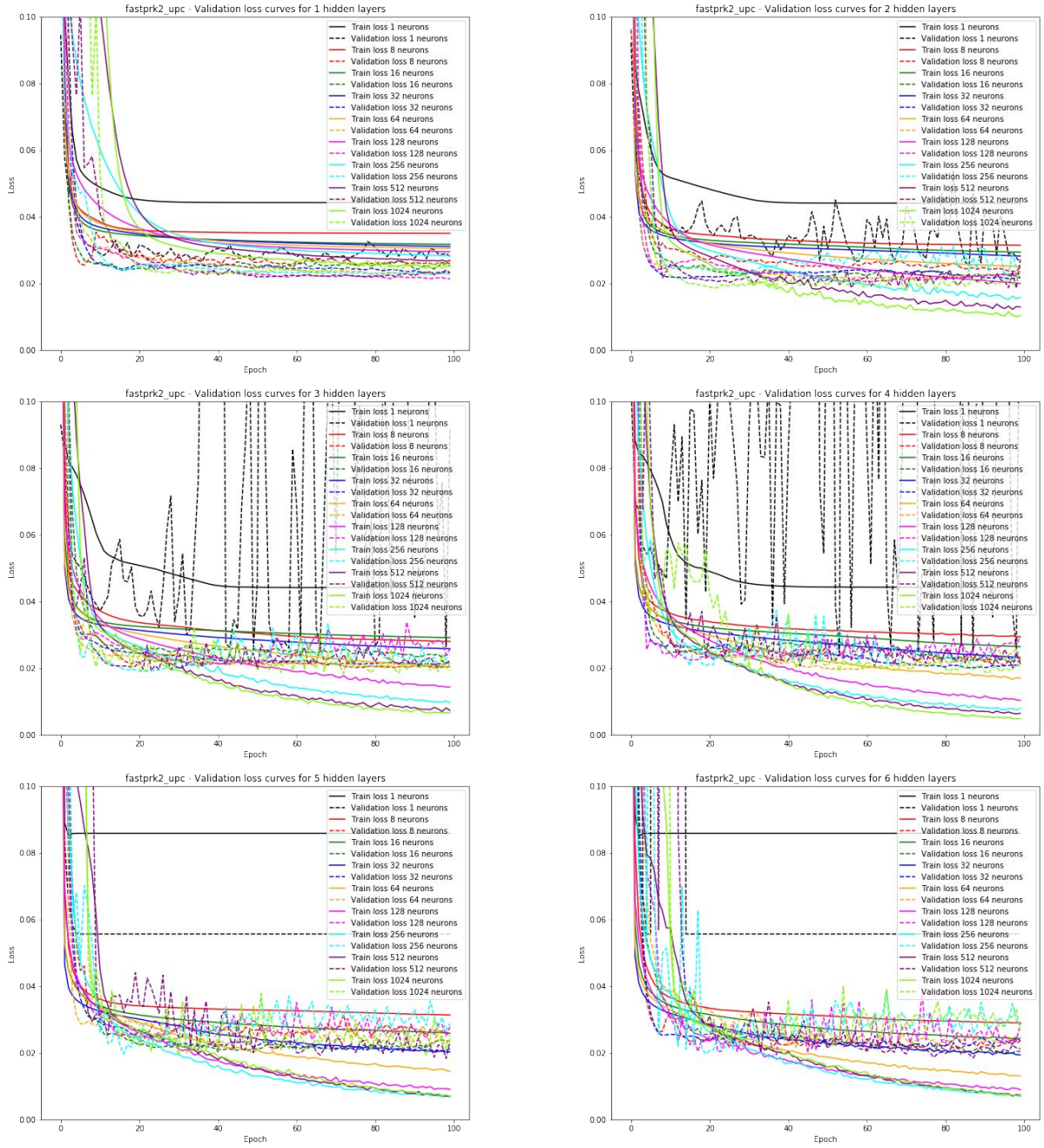


Figure 7.25: Barcelona losses as the number of hidden layers increases.

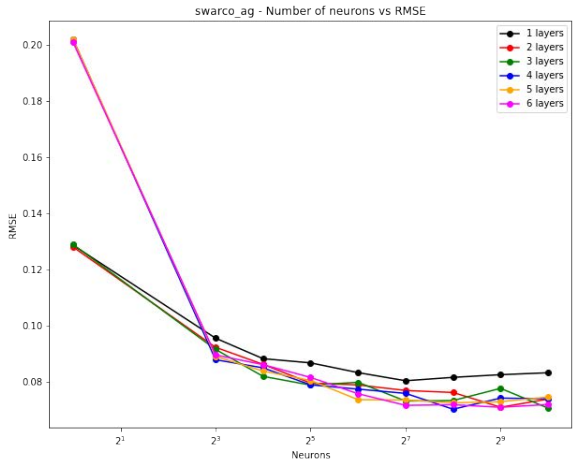


Figure 7.26: Wattens validation RMSE comparison of models with different numbers of layers and neurons.

One hidden layer		Two hidden layers		Three hidden layers		Four hidden layers		Five hidden layers		Six hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.0805	128	0.0711	512	0.0707	1024	0.0704	256	0.0726	256	0.0710	512
0.0817	256	0.07400	1024	0.0733	128	0.0741	1024	0.0730	512	0.0717	128
0.0826	512	0.0763	256	0.0734	256	0.0743	512	0.0736	128	0.0719	256
0.08330	1024	0.0770	128	0.0778	512	0.0760	128	0.0737	64	0.0720	1024
0.0834	64	0.079	64	0.0789	32	0.0775	64	0.0747	1024	0.0759	64

Table 7.20: Wattens top validation RMSE values for different numbers of layers.

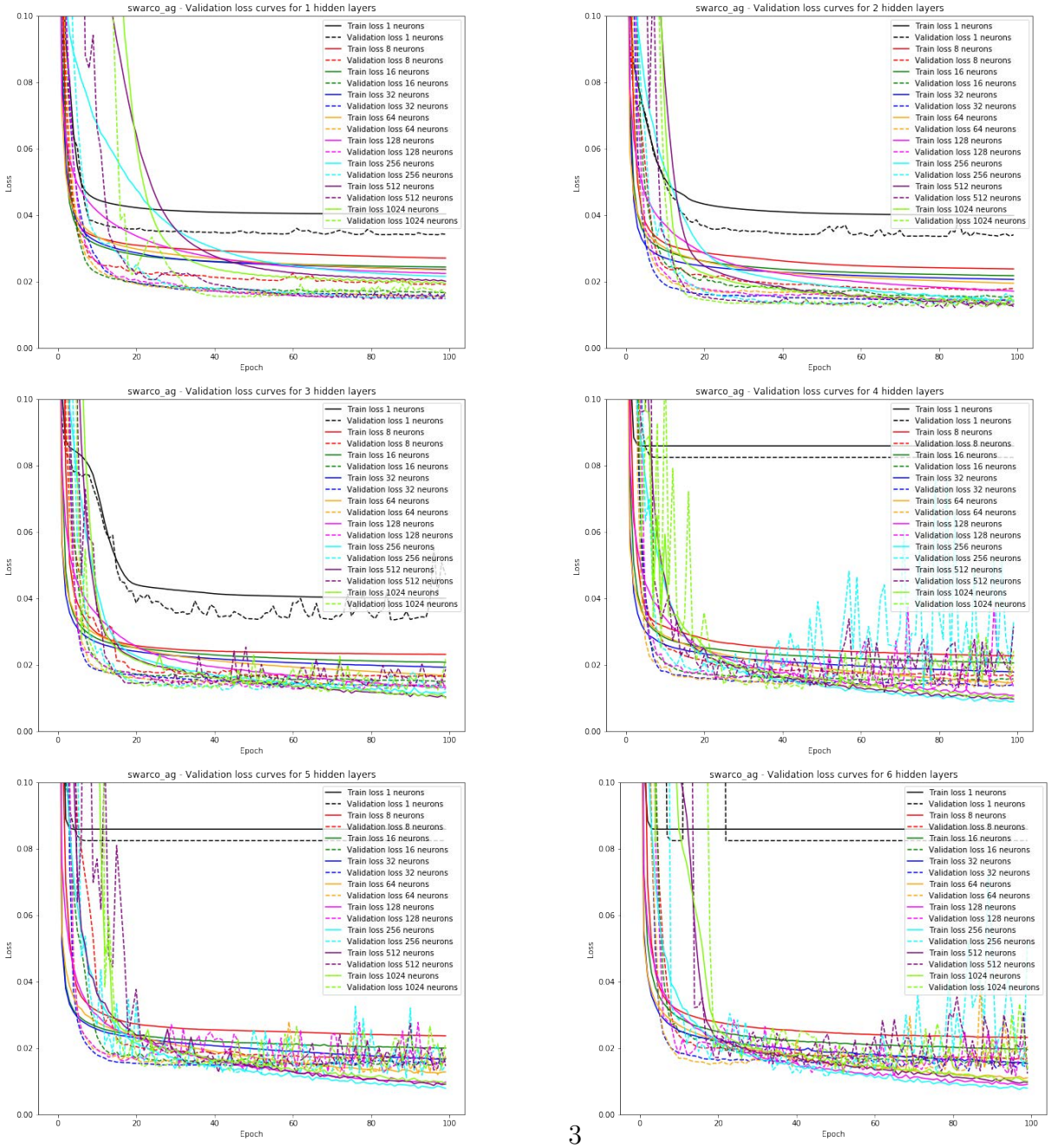
or increase in the RMSE stops independently of the number of layers when the number of neurons is higher than 64.

Figure 7.27 shows the loss curves for the models with one to six hidden layers. Similar observations to those extracted from the Barcelona scenario can be concluded in this case. The only noticeable difference is that the gap between training and loss curves increases very slightly as model capacity grows when compared to previous scenarios.

Table 7.20 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for the different numbers of hidden layers considered. As observed in the validation RMSE visualization, similar RMSE (the lowest) values are obtained with different numbers of neurons: 256, 512 and 1024.

Figure 7.28 shows the validation RMSE values for the Standard sector from Los Angeles at different combinations of hidden layers and neurons. For eight neurons, all models offer the same RMSE value and, at that point, the models with higher numbers of neurons obtain lower RMSE and those with low numbers of neurons present higher RMSE.

Figure 7.29 shows the loss curves for the models with one to six hidden layers. Observe that validation losses are much lower than training losses, and the cause of this is because



3

Figure 7.27: Wattens losses as the number of hidden layers increases.

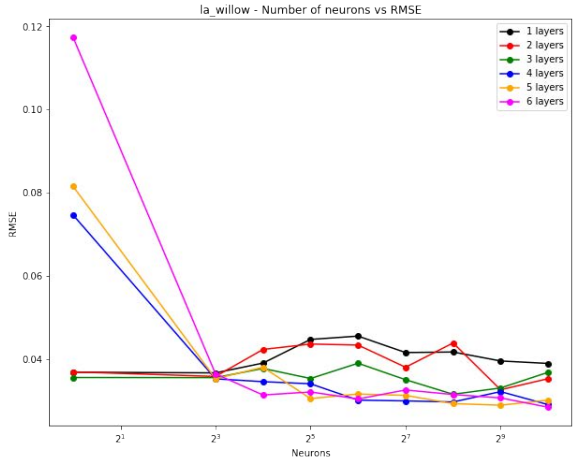


Figure 7.28: Los Angeles validation RMSE comparison of models with different numbers of layers and neurons.

One hidden layer		Two hidden layers		Three hidden layers		Four hidden layers		Five hidden layers		Six hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.0367	8	0.0326	512	0.0316	256	0.02910	1024	0.0289	512	0.02840	1024
0.0369	1	0.0350	1024	0.0331	512	0.02970	256	0.0293	256	0.0304	64
0.0390	1024	0.0358	8	0.0351	128	0.0300	128	0.03010	1024	0.0307	512
0.0391	16	0.0369	1	0.0353	32	0.0301	64	0.0305	32	0.0314	16
0.0396	512	0.0380	128	0.0356	8	0.0322	512	0.0313	128	0.0315	256

Table 7.21: Los Angeles top validation RMSE values for different numbers of layers.

validation data corresponds to the period of works in the sector (2019). All models with more than 32 neurons converge towards very similar validation loss values. Similarly to other scenarios, as the number of hidden layers increases, those models with lower numbers of neurons become more unstable.

Table 7.21 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for the different numbers of hidden layers considered. There is no huge difference in the RMSE for the different numbers of neurons, but as the number of hidden layers increases, the ones achieving the lowest validation RMSE are 512 and 128. Also, the effect of having more hidden layers is reflected as a direct decrease in the validation RMSE value.

The last scenario considered pertains to Sector-1 in Riyadh. The validation RMSE values with different combinations of hidden layers and neurons at each layer is presented in visualization 7.30. Similarly to the Barcelona scenario, the increasing number of hidden layers and neurons provides lower values for the validation RMSE; but at a certain point (64 neurons), the reduction in the RMSE value becomes lower.

Figure 7.31 shows the loss curves for the models with one to six hidden layers. The behaviour of the validation and training losses is similar to those presented in the Barcelona

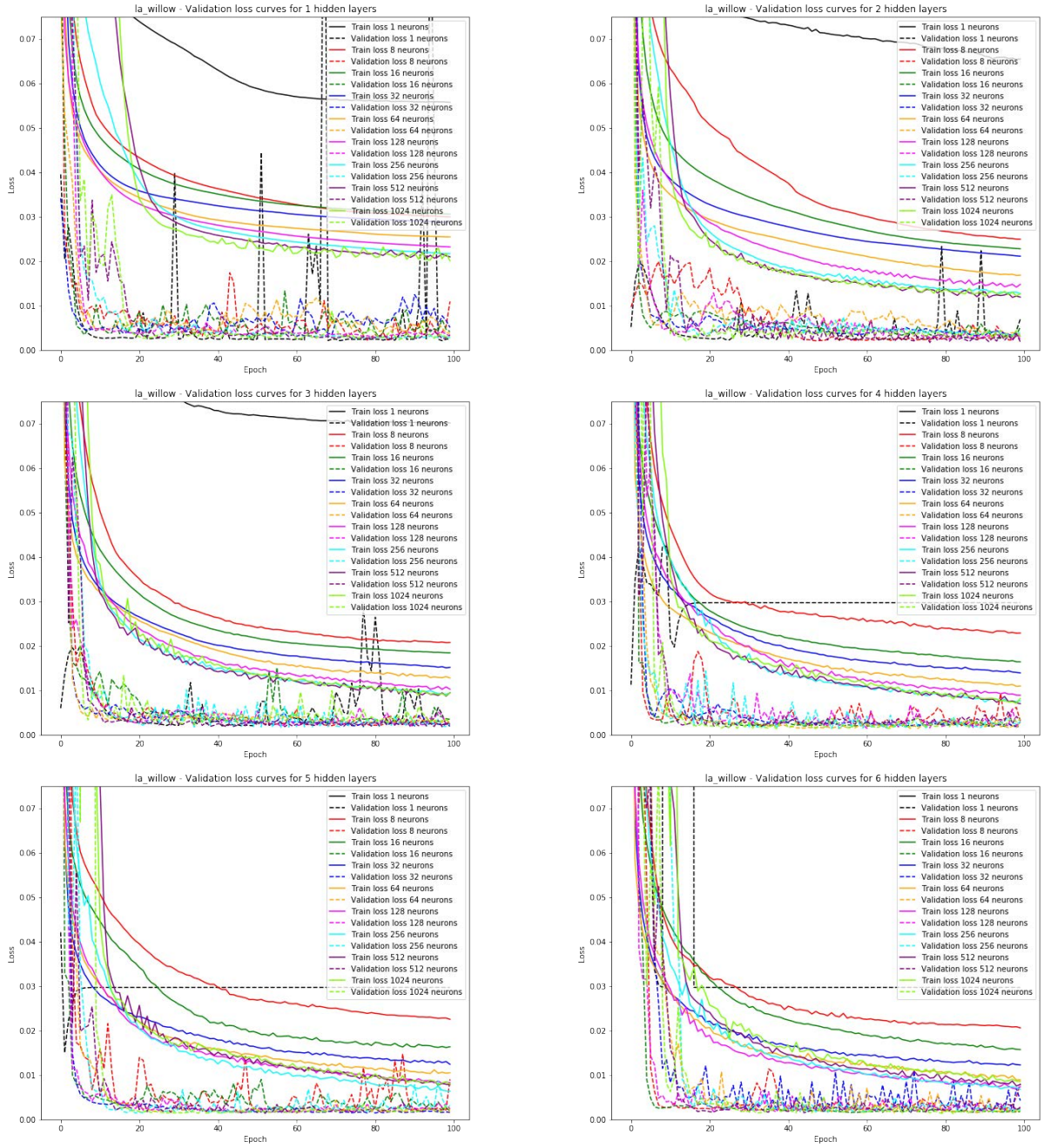


Figure 7.29: Los Angeles losses as the number of hidden layers increases.

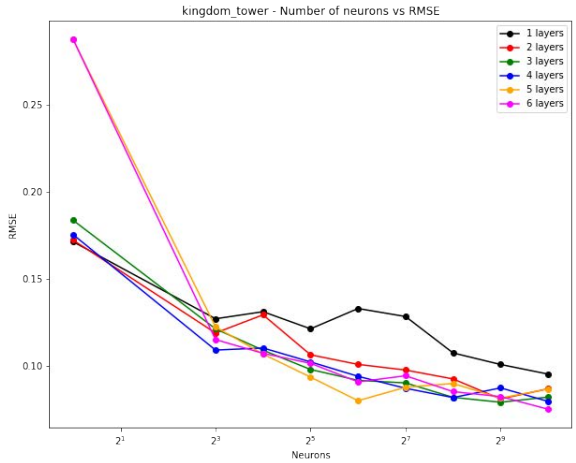


Figure 7.30: Riyadh validation RMSE comparison of models with different numbers of layers and neurons.

One hidden layer		Two hidden layers		Three hidden layers		Four hidden layers		Five hidden layers		Six hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.09540	1024	0.0811	512	0.0793	512	0.07970	1024	0.0801	64	0.07530	1024
0.1009	512	0.08700	1024	0.0812	256	0.0819	256	0.0816	512	0.0825	512
0.1075	256	0.0926	256	0.0822	1024	0.0873	128	0.08690	1024	0.0854	256
0.1213	32	0.0977	128	0.0903	128	0.0875	512	0.0879	128	0.0909	64
0.1271	8	0.101	64	0.0918	64	0.0941	64	0.09	256	0.0945	128

Table 7.22: Riyadh top validation RMSE values for different numbers of layers.

scenario. Notice that as the number of hidden layers becomes higher, the validation loss curve becomes more unstable, independently of the number of neurons. Models with 64 neurons seem to have the more stable behaviour across the increasing number of layers. Lastly, at a high number of hidden layers, models with greater than or equal to 64 neurons converge towards the same value.

Table 7.22 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for the different numbers of hidden layers considered. Clearly, 128 and 512 neurons provide the best validation RMSE values.

In conclusion, the MLP models with between two and five hidden layers and 512 and 128 neurons provide the lowest RMSE values across the considered scenarios.

7.4.2.2 Recurrent methods

For recurrent NN, namely LSTM and GRU, the number of considered hidden layers has been reduced to between one and three, while the number of neurons considered is the same as in the MLP experiment.

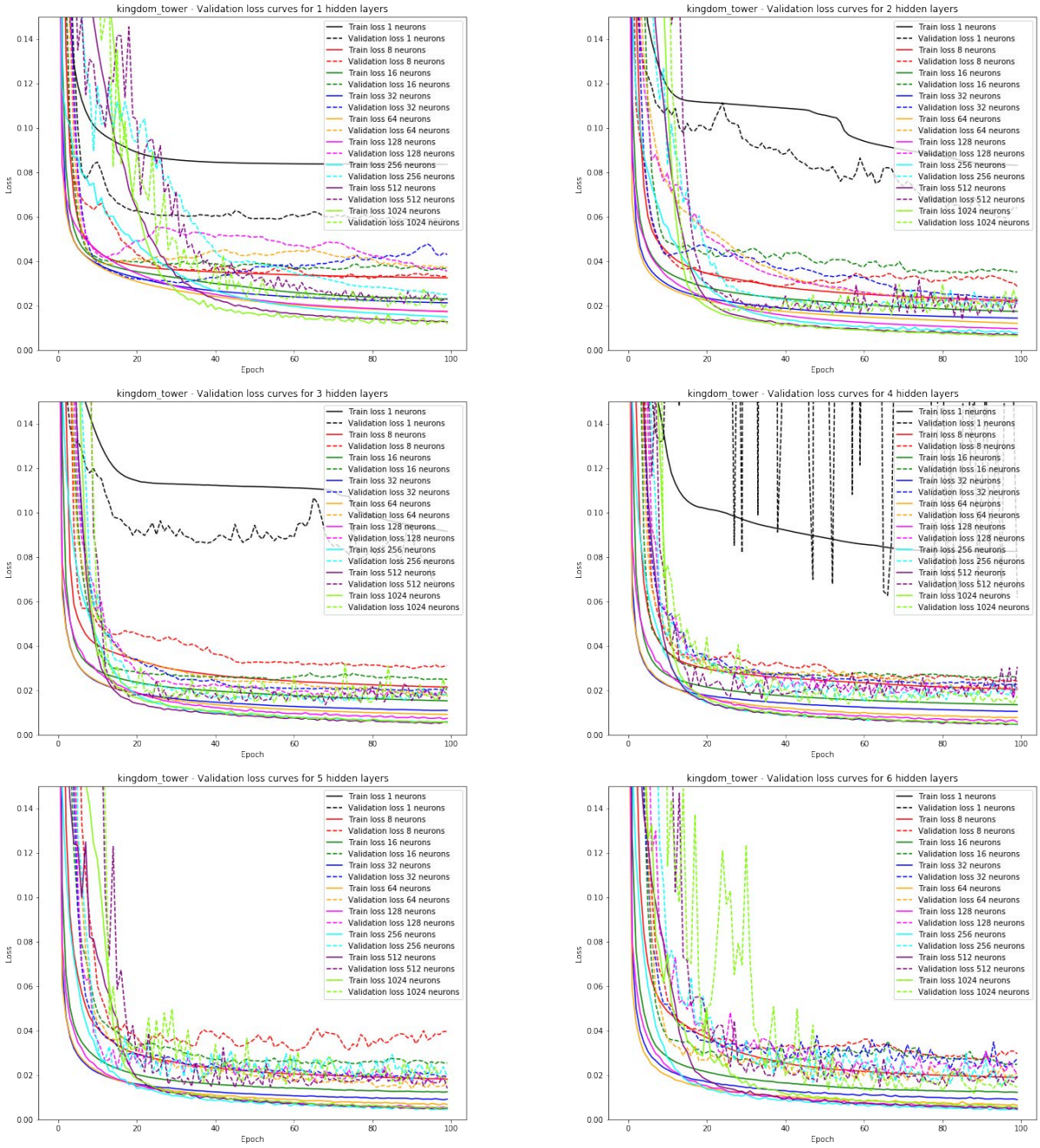


Figure 7.31: Riyadh losses as the number of hidden layers increases.

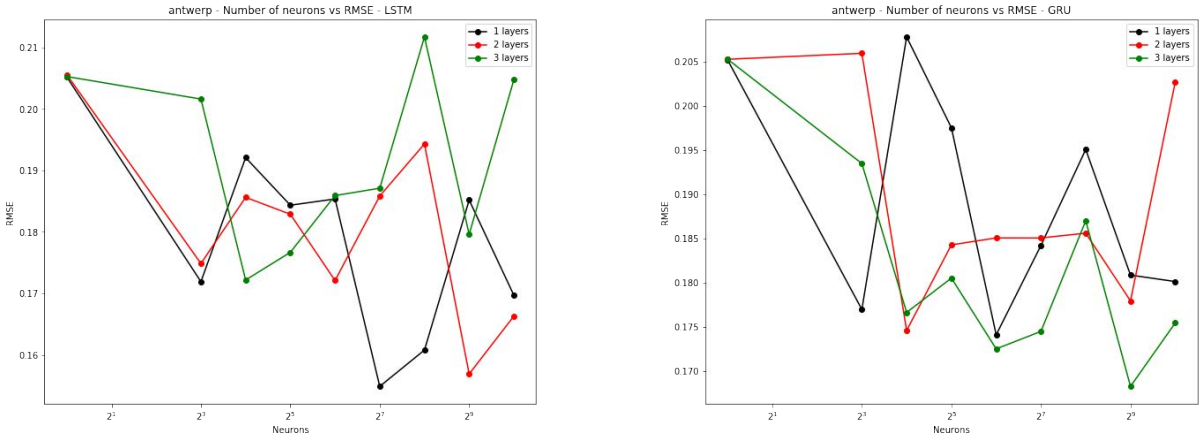


Figure 7.32: Antwerp validation RMSE comparison of models with different numbers of layers and neurons. Left corresponds to the LSTM models and right to GRU.

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.1549	128	0.1569	512	0.1722	16
0.16018	256	0.1663	1024	0.1767	32
0.1697	1024	0.1721	64	0.1796	512
0.1719	8	0.1749	8	0.1859	64
0.1843	32	0.1829	32	0.1871	128

(a) LSTM

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.174080	64	0.174568	16	0.168244	512
0.176952	8	0.177921	512	0.168244	64
0.180112	1024	0.184268	32	0.174470	128
0.1808441	512	0.185047	128	0.175433	1024
0.1842004	128	0.185050	64	0.1766165	16

(b) GRU

Table 7.23: Antwerp top validation RMSE values for different numbers of layers.

Figure 7.32 presents the validation RMSE for the Pelikaanstraat sector in Antwerp for different combinations of hidden layers and neurons considered. The visualization shows that, independently of the number of layers, there is no clear pattern among the number of neurons.

The behaviour of the losses (Figure 7.33 for LSTM and Figure 7.34 for GRU) offers detailed information about the model training. As observed in the validation RMSE visualization, the models have more stable behaviour as the number of hidden layers increases. Notice that, independently of the number of hidden layers, the models with 32, 64, 128 and 256 neurons present a more stable validation loss.

The top validation RMSE values for different model capacities are presented in Table 7.23. As seen previously in the visualizations, there is no clear pattern for distinguishing the effect of different numbers of neurons. Models with one and two hidden layers present similar validation RMSE values, which are lower than those presented by the models with three hidden layers.

Figure 7.35 presents the LSTM and GRU validation RMSE for the Barcelona scenario when considering different numbers of hidden layers and neurons. Contrary to the

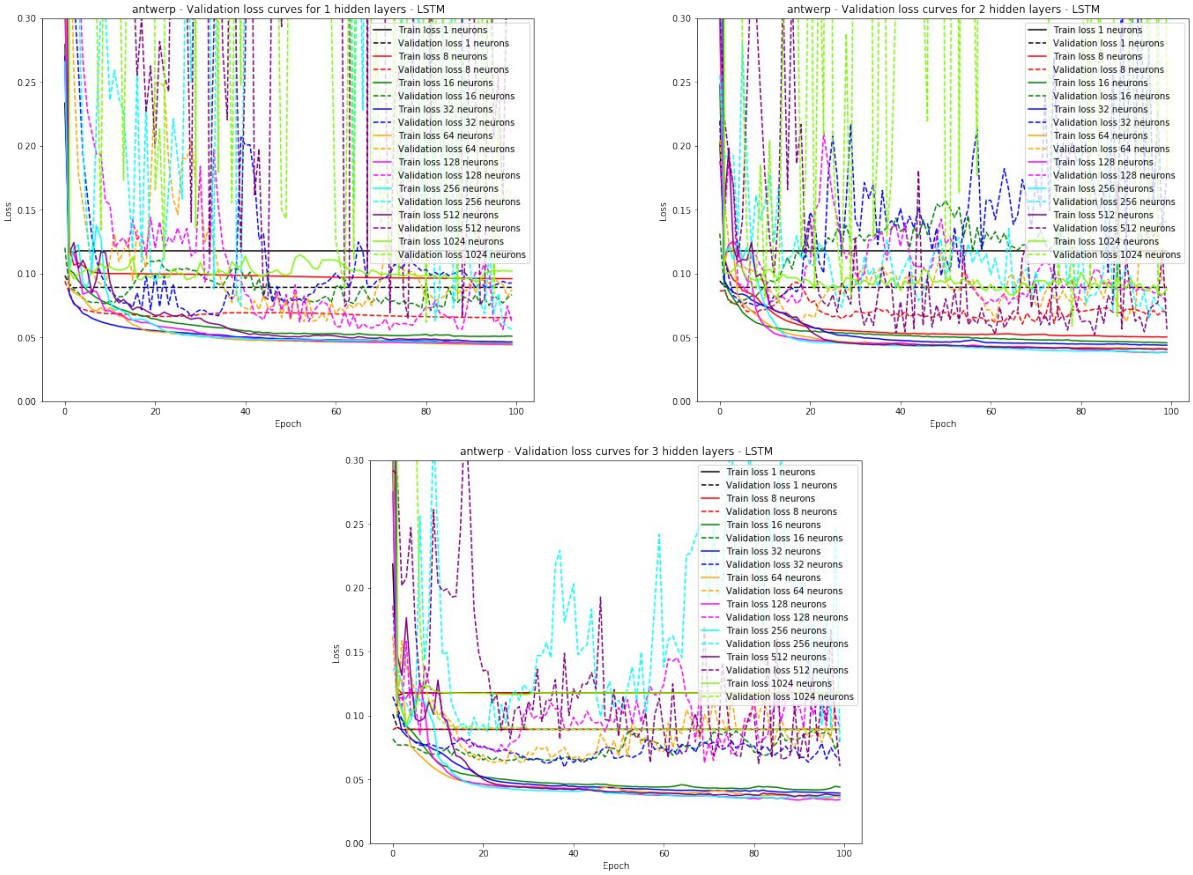


Figure 7.33: Antwerp losses as the number of hidden layers increases for LSTM.

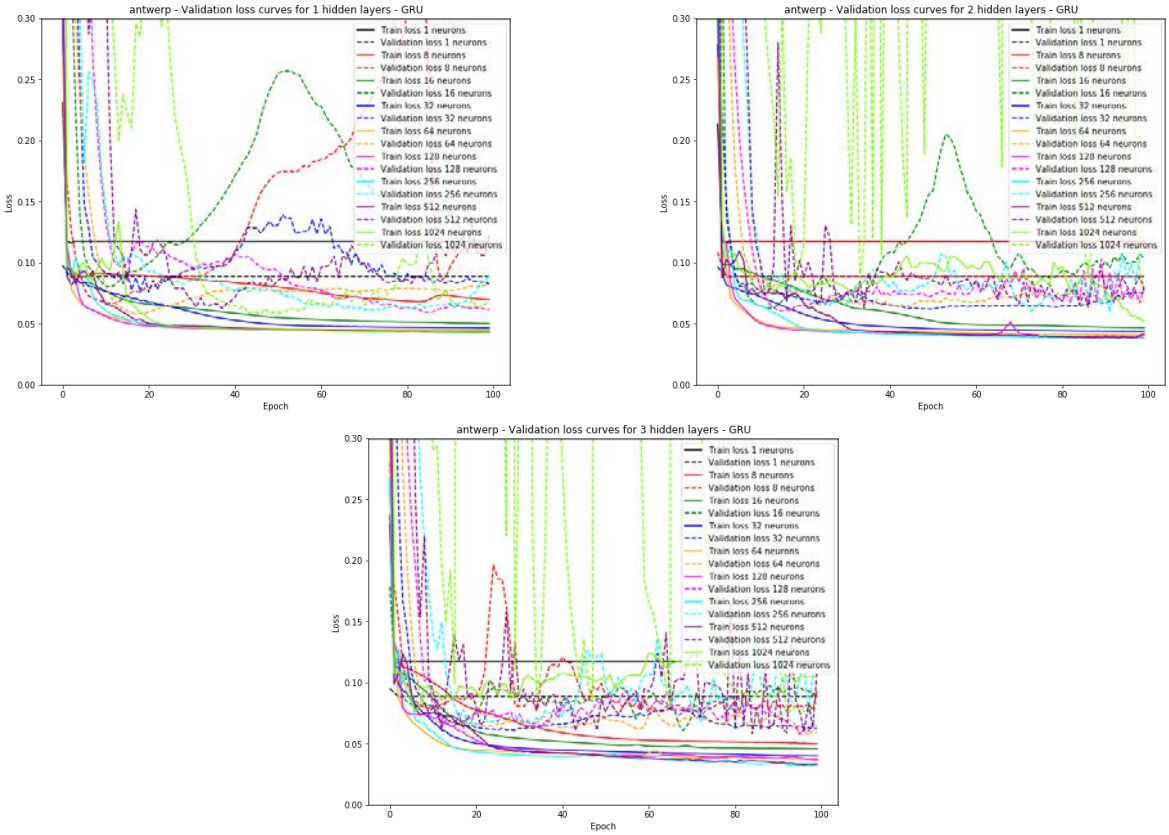


Figure 7.34: Antwerp losses as the number of hidden layers increases for GRU.

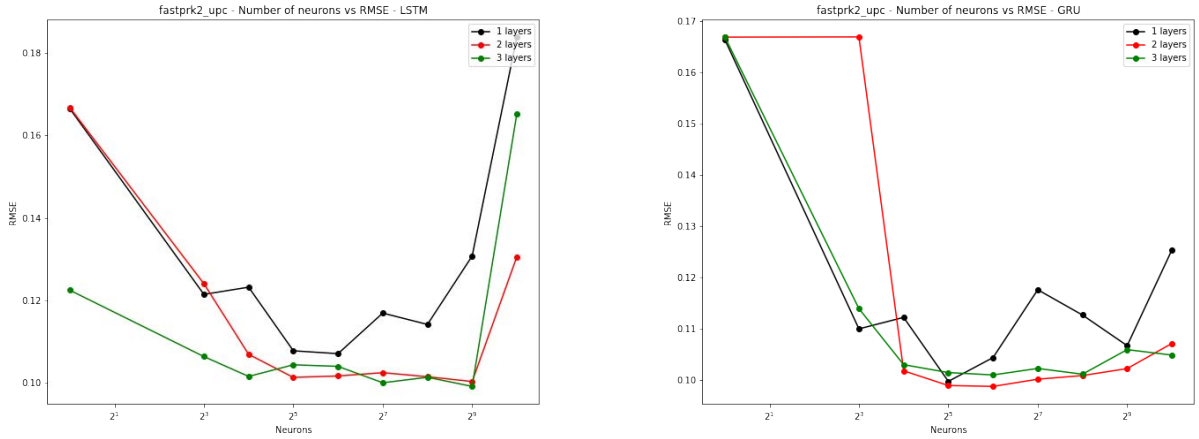


Figure 7.35: Barcelona validation RMSE comparison of the LSTM and GRU models with different numbers of layers and neurons.

results from the Antwerp scenario, this scenario clearly presents the effect of increasing the number of layers and neurons. The visualizations show that low and high numbers of neurons have a negative effect on the models and that the lowest validation RMSE values can be achieved at different but nearly the same numbers of neurons.

Figure 7.36 shows the loss curves for the different numbers of layers considered. The LSTM models with one hidden layer have a more unstable validation loss curve, but as the number of layers increases, more models with different numbers of neurons converge towards near validation loss values. Also, models with 124 neurons have unstable losses for the three cases.

Figure 7.37 presents the validation losses for the GRU models. Unlike LSTM, the validation losses present stable values, independently of the number of neurons/hidden layers.

Table 7.24 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for the different numbers of hidden layers considered for recurrent models. Observe that as the number of hidden layers increases, the validation RMSE becomes lower. The best scores for two and three hidden layers are achieved with 512 neurons, while 64 neurons achieve the lowest validation RMSE in the case of one hidden layer.

Figure 7.38 presents the RMSE validation values for the Inside sector in Wattens under different combinations of hidden layers and numbers of neurons. Similarly to the Barcelona scenario, the number of hidden layers has an effect on the validation RMSE, as two and three hidden layers offer lower RMSE values than the models with one hidden layer. Furthermore, a too low or too high number of neurons also worsens the RMSE values, which is more noticeable in LSTM than in GRU.

The Wattens scenario training and validation losses from the LSTM models are presented in Figure 7.39 for different combinations of hidden layers and neurons. Loss curves show

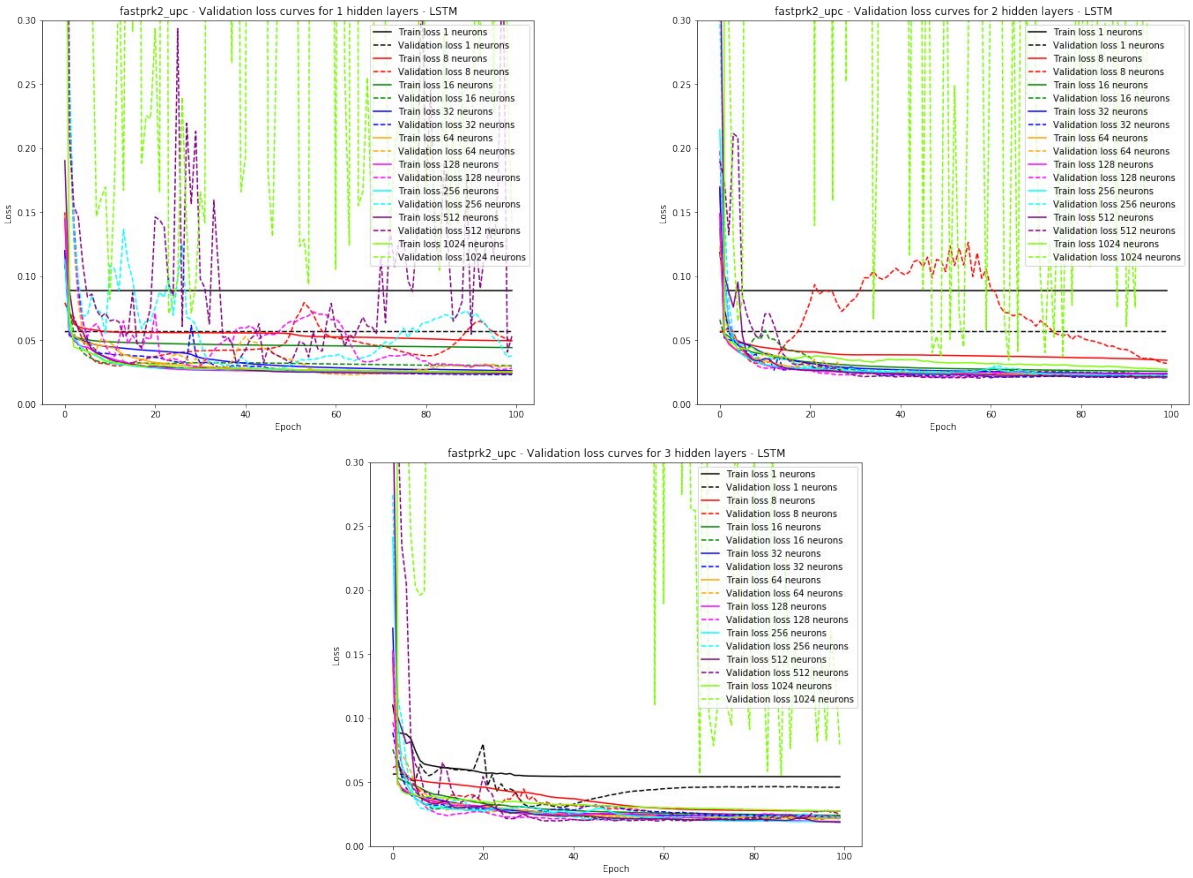


Figure 7.36: Barcelona LSTM losses as the number of hidden layers increases.

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.1071	64	0.1003	512	0.0992	512
0.1078	32	0.1013	32	0.10	128
0.1141	256	0.1015	256	0.1014	256
0.1169	128	0.1017	64	0.1016	16
0.1214	8	0.1025	128	0.1040	64

(a) LSTM

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.0998	32	0.0988	64	0.101	64
0.1044	64	0.099	32	0.1012	256
0.1068	512	0.1002	128	0.1015	32
0.11	8	0.1009	256	0.1023	128
0.1123	16	0.1018	16	0.103	16

(b) GRU

Table 7.24: Top validation RMSE values for the Barcelona scenario at different combinations of hidden layers and neurons.

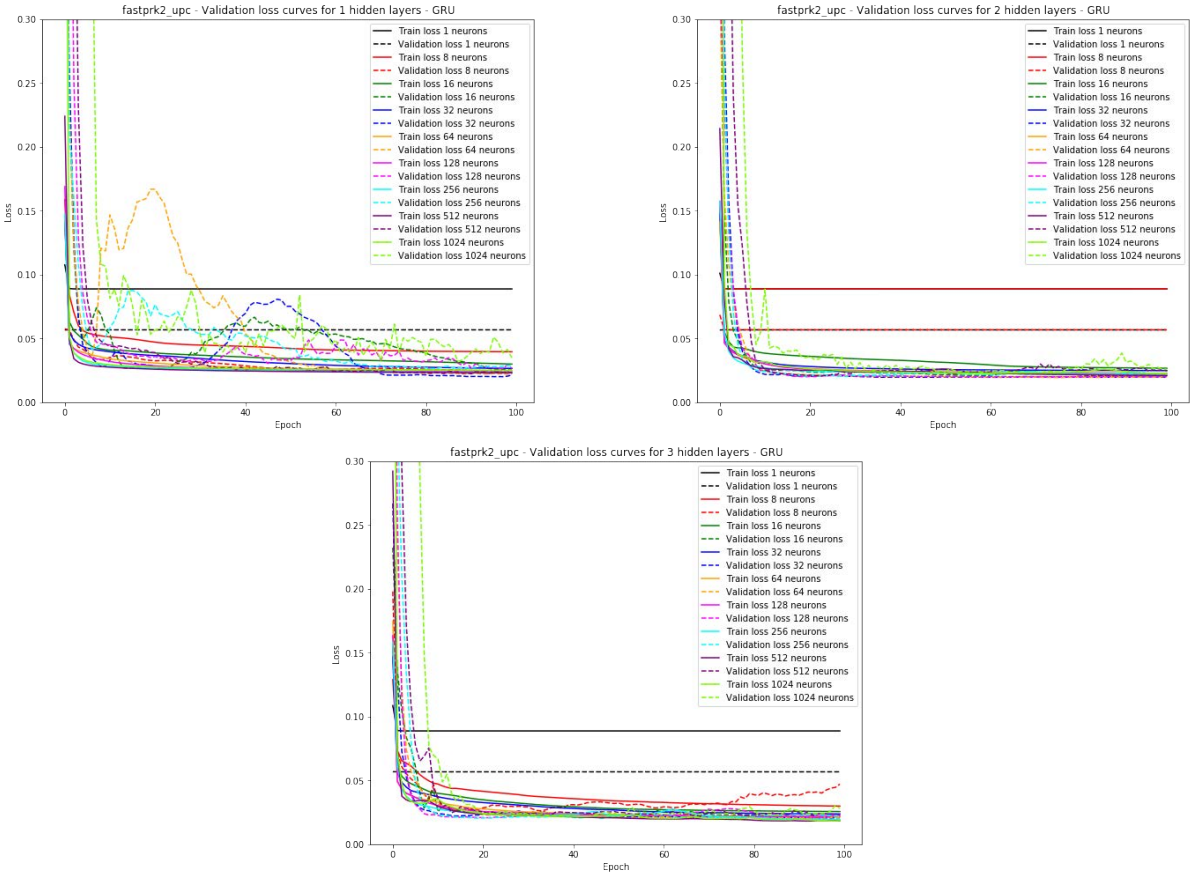


Figure 7.37: Barcelona GRU losses as the number of hidden layers increases.

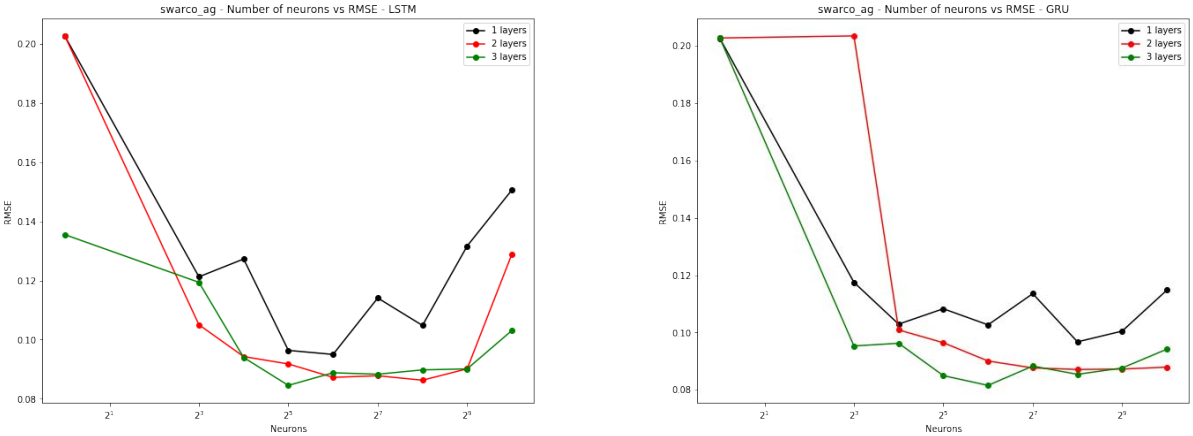


Figure 7.38: Wattens validation RMSE comparison of models with different numbers of layers and neurons. Left corresponds to the LSTM models and right to GRU.

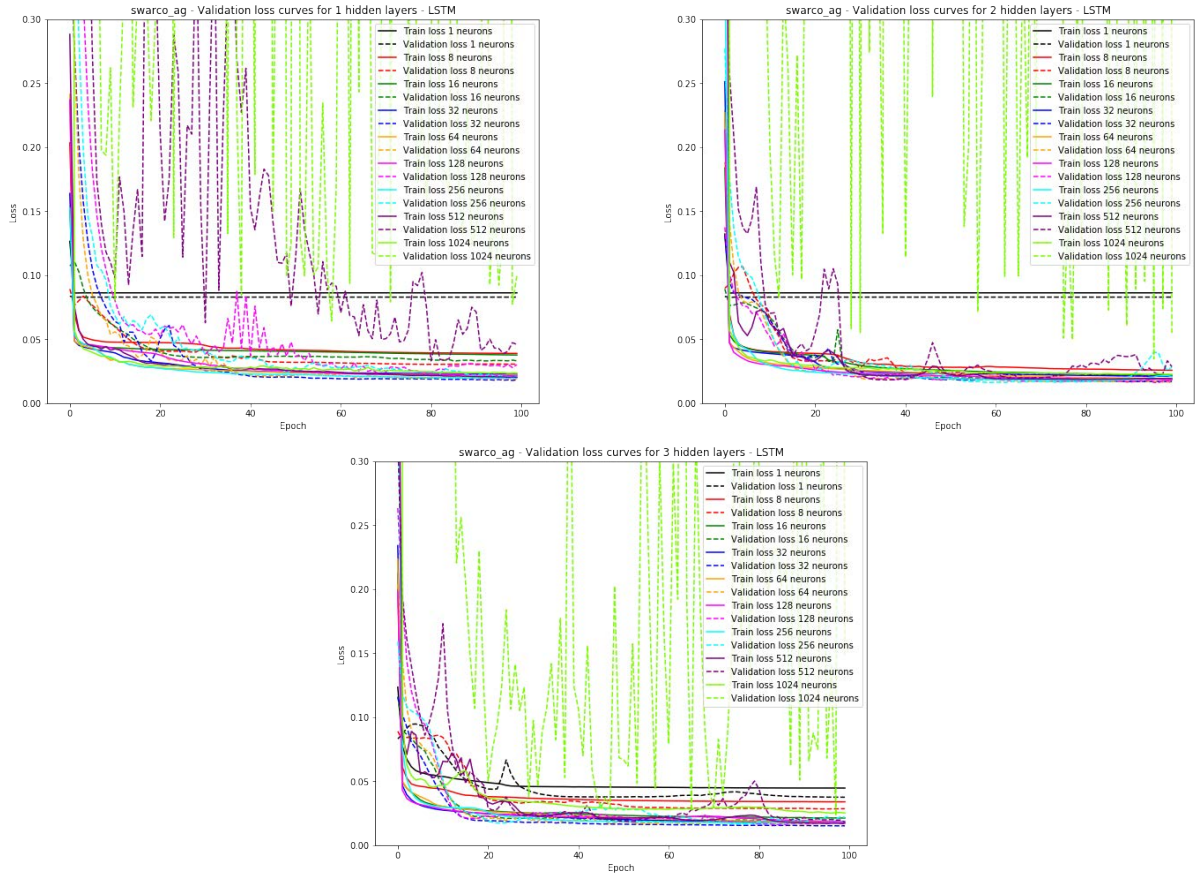


Figure 7.39: Wattens LSTM losses as the number of hidden layers increases.

that, at the three levels of hidden layers considered, those models with 128 neurons are not able to converge on the validation loss (overfit on the training data). What is more, a similar effect can be seen for one hidden layer and 512 neurons. Notice that the number of neurons has an effect towards the convergence value of the validation loss for one and three hidden layers, but at two hidden layers this effect is diminished.

GRU visualizations are presented in Figure 7.40. The behaviours of the validation losses are like the previous scenarios in which models converge towards similar validation RMSE values, independently of the numbers of neurons and hidden layers.

Table 7.25 presents the top five obtained validation RMSE values, the numbers of neurons used to obtain them, and the different numbers of hidden layers. The number of hidden layers has an effect on the validation RMSE, as two and three hidden layers offer less RMSE than the models with one hidden layer. The number of neurons seems not to be very critical as long as it is not too low or too high.

Figure 7.41 shows the validation RMSE values for the Standard sector from Los Angeles at different combinations of hidden layers and neurons. Those combinations with two and three hidden layers offer lower RMSE values than those with one hidden layer at nearly all the combinations considered. For the LSTM models, the lowest validation

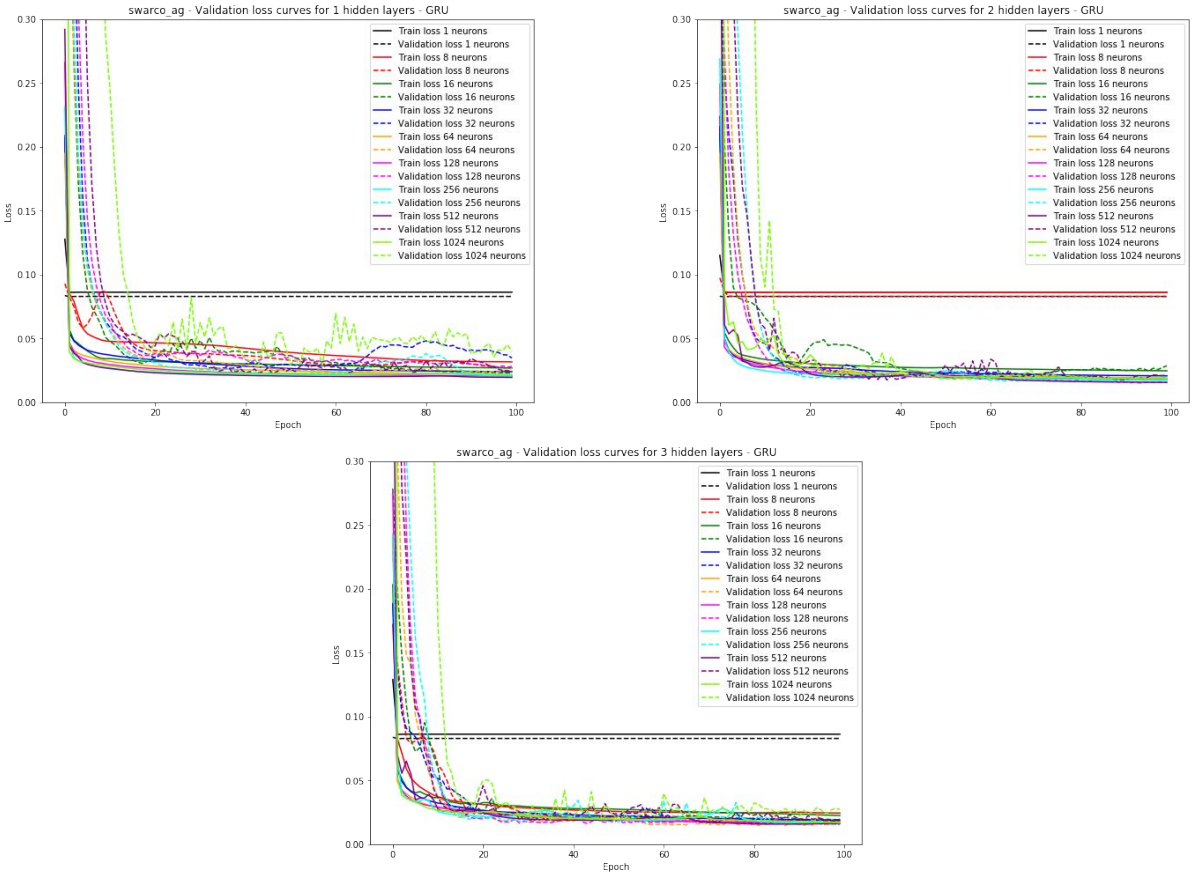


Figure 7.40: Wattens GRU losses as the number of hidden layers increases.

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.095	64	0.0863	256	0.0846	32
0.0964	32	0.0872	64	0.0883	128
0.1049	256	0.0878	128	0.0888	64
0.1141	128	0.0901	512	0.0898	256
0.1212	8	0.0918	32	0.0901	512

(a) LSTM

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.0968	256	0.0871	256	0.0816	64
0.1005	512	0.0873	512	0.085	32
0.1027	64	0.0877	128	0.0854	256
0.1029	16	0.0880	1024	0.0876	512
0.1083	32	0.0901	64	0.0883	128

(b) GRU

Table 7.25: Wattens top validation RMSE values for different numbers of layers.

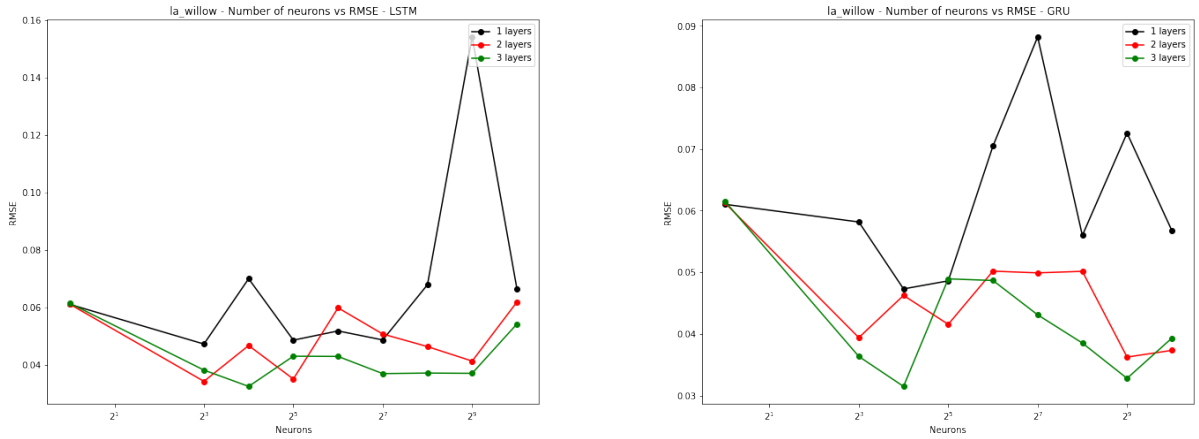


Figure 7.41: Los Angeles validation RMSE comparison of models with different number of layers and neurons. Left corresponds to LSTM and right to GRU.

RMSE values are obtained with three hidden layers when considering different numbers of neurons. GRU presents similar results, but the worsening of validation RMSE values caused by having a high number of neurons is less noticeable.

Training loss and validation loss visualizations for different combinations of hidden layers and neurons are presented in Figure 7.42. At one hidden layer, models are unable to generalize validation data, but as the number of hidden layers increases, the range of models that offer low RMSE values increases. Similarly to other scenarios, a low or high number of neurons can be decisive towards the generalization of the model. Remember that, for this scenario, it is of prime importance that the models generalize because the distribution of the data changes as time goes on.

Figure 7.43 presents the validation losses from the GRU models. In this scenario, the GRU models present unstable validation losses at one hidden layer when a low or high number of neurons is considered. As the number of hidden layers increases, loss curves become more stable and nearly all models converge towards similar values.

Table 7.26 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for the different numbers of hidden layers considered. There is no huge difference in the RMSE for the different numbers of neurons, but there is a difference as the number of hidden layers increases. Having more hidden layers is reflected as a direct decrease in the validation RMSE value.

Figure 7.44 shows the behaviour of the validation RMSE as the numbers of hidden layers and neurons increase. Similarly to other sectors, a low number of neurons has a negative effect on the RMSE values. Moreover, with the adequate number of neurons, similar validation RMSE values are obtained independently of the number of hidden layers. LSTM presents an increase in the RMSE when the number of neurons is greater than 512, an effect that is not noticeable in the GRU models.

Figure 7.45 shows the loss curves for the LSTM models with one to three hidden layers. Similarly to other scenarios, as the number of hidden layers increases, the validation

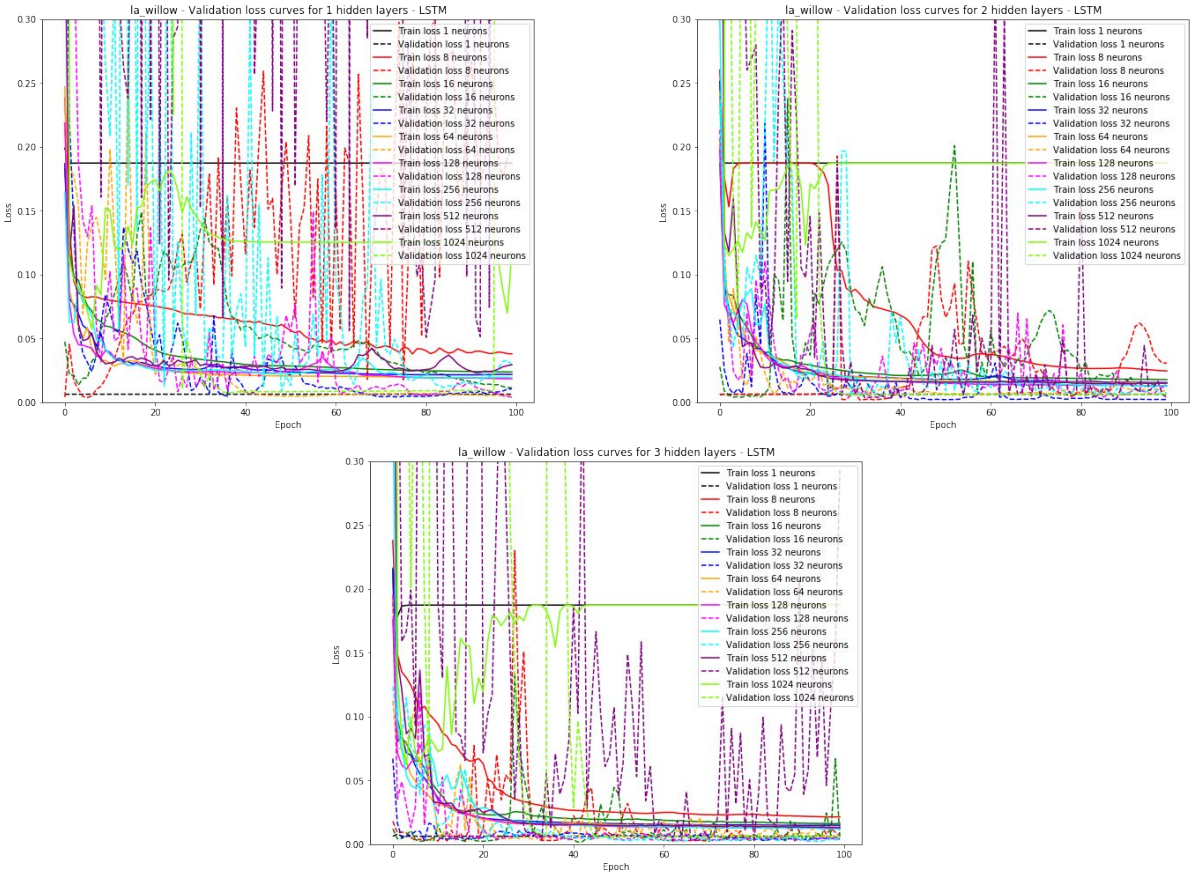


Figure 7.42: Los Angeles LSTM losses as the number of hidden layers increases.

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.0473	8	0.0343	8	0.0326	16
0.0487	32	0.0351	32	0.0370	128
0.0487	128	0.0413	512	0.0371	512
0.0519	64	0.0464	256	0.0372	256
0.0611	1	0.0468	16	0.0382	8

(a) LSTM

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.0473	16	0.0362	512	0.0315	16
0.0486	32	0.0373	1024	0.0328	512
0.0560	256	0.0394	8	0.0363	8
0.0568	1024	0.0416	32	0.0385	256
0.0582	8	0.0463	16	0.0393	1024

(b) GRU

Table 7.26: Los Angeles top validation RMSE values for different numbers of layers.

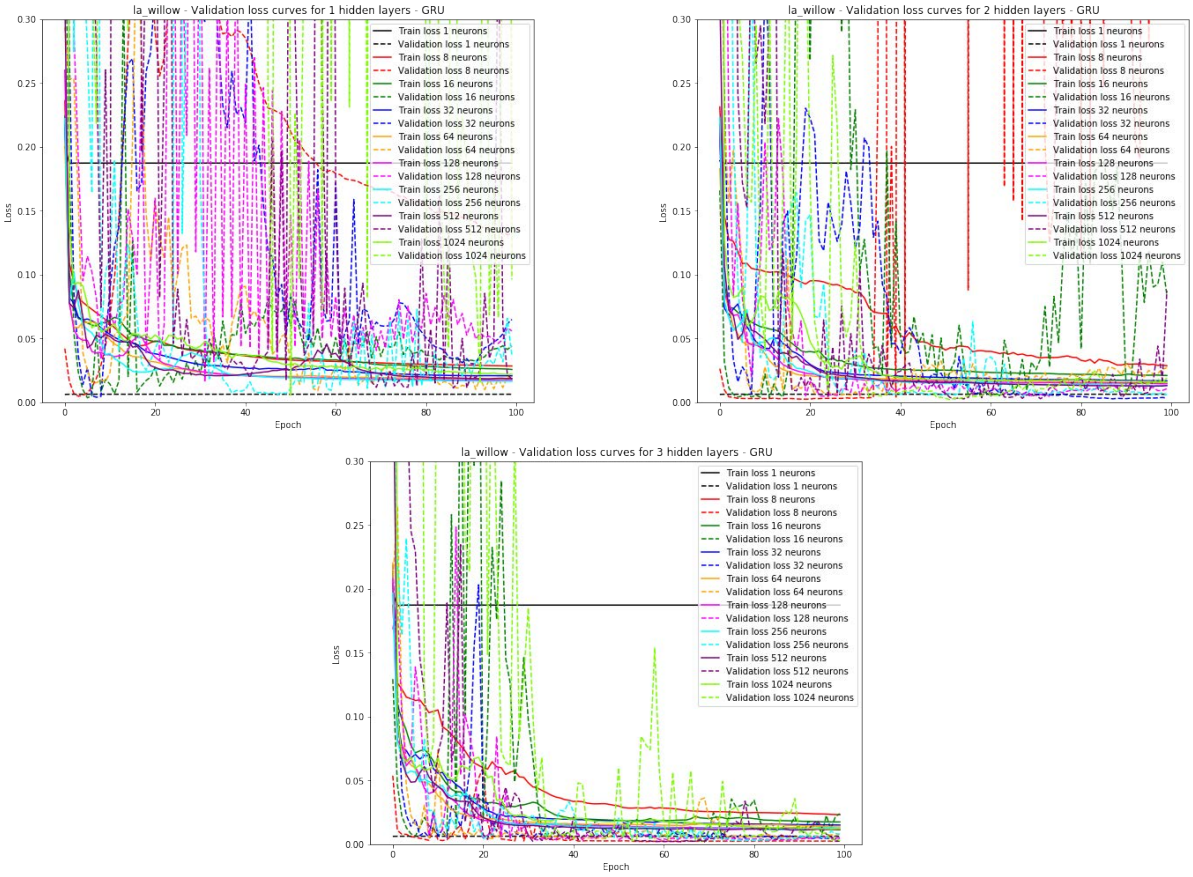


Figure 7.43: Los Angeles GRU losses as the number of hidden layers increases.

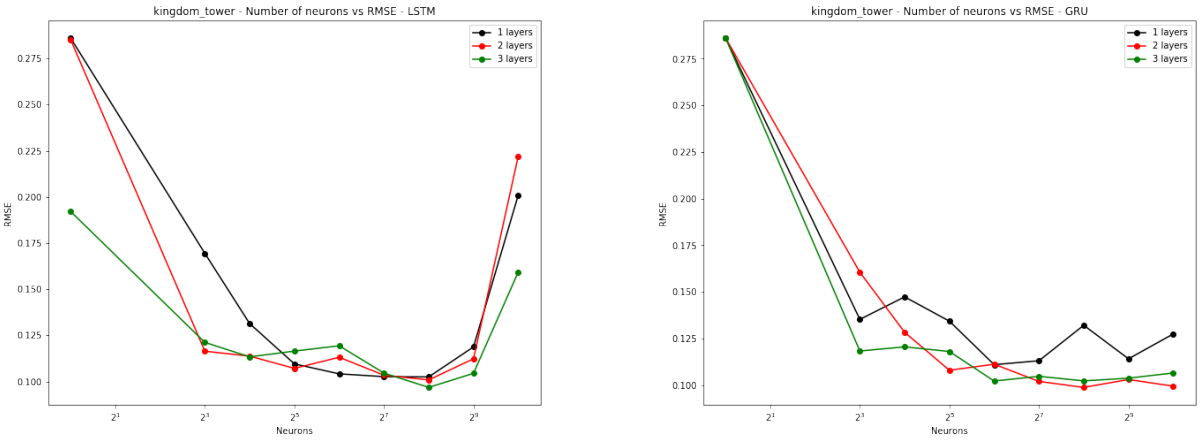


Figure 7.44: Riyadh validation RMSE comparison of models with different numbers of layers and neurons. Left corresponds to LSTM and right to GRU.

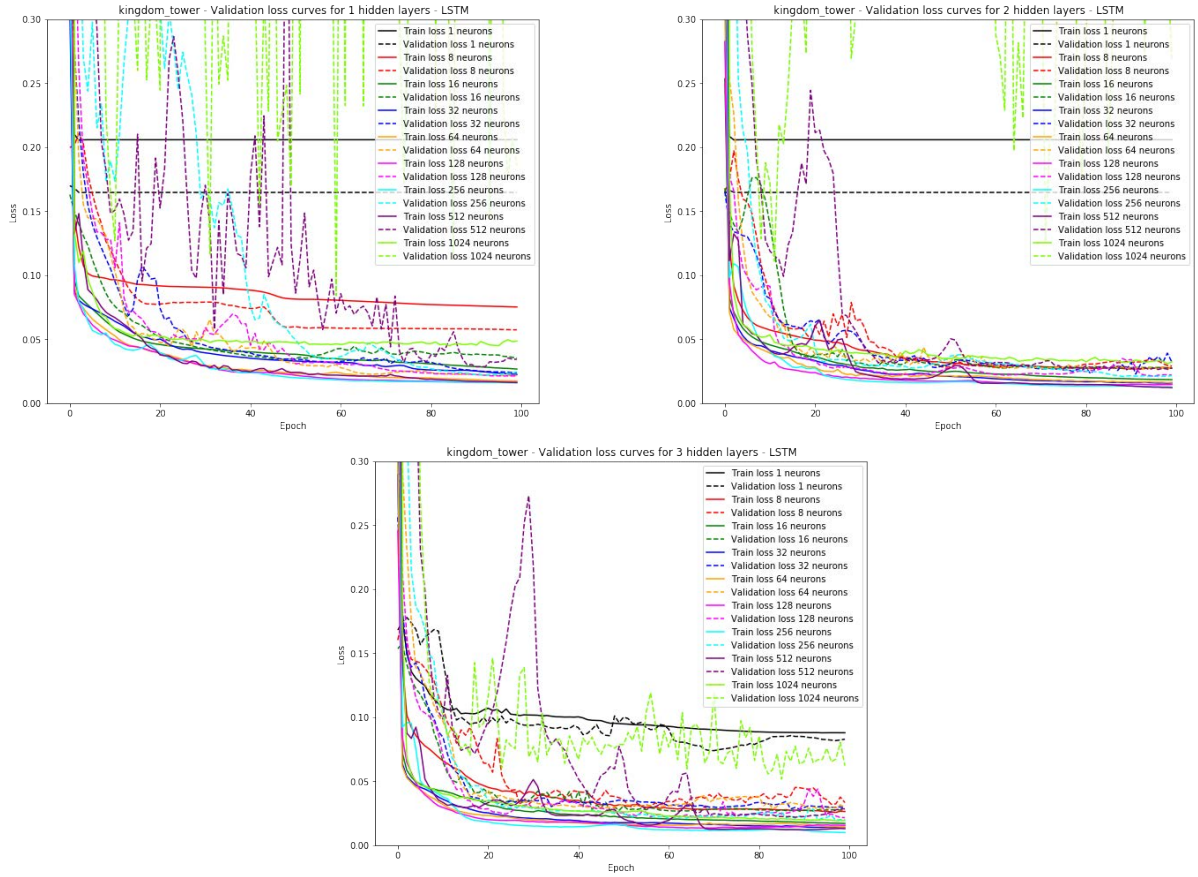


Figure 7.45: Riyadh LSTM losses as the number of hidden layers increases.

loss becomes more stable for the models with a high number of neurons. Clearly, model capacity plays a role, as models with only one neuron have higher loss values than the others.

Figure 7.46 presents the losses for the GRU models. As observed in previous scenarios, GRU presents validation losses that are more stable than GRU at one hidden layer, and the stability also increases as the number of hidden layers increases. With three hidden layers, all models (except the one with one neuron) converge towards similar values.

Table 7.27 presents the top five obtained validation RMSE values and the number of neurons used to obtain them for one, two and three hidden layers for recurrent models LSTM and GRU. For LSTM, 256 neurons offer the lowest validation RMSE values for each of the number of hidden layers considered. Moreover, one and two hidden layers coincide in the number of neurons in the validation RMSE order. GRU also presents the best results at two hidden layers with 256 neurons.

In conclusion, for the scenarios considered, the LSTM and GRU models show lower RMSE values when working in the range of 8 to 256 neurons. Although more hidden layers have the effect of making models with higher numbers of neurons converge, the results are not better than those models with a lower number of layers and neurons,

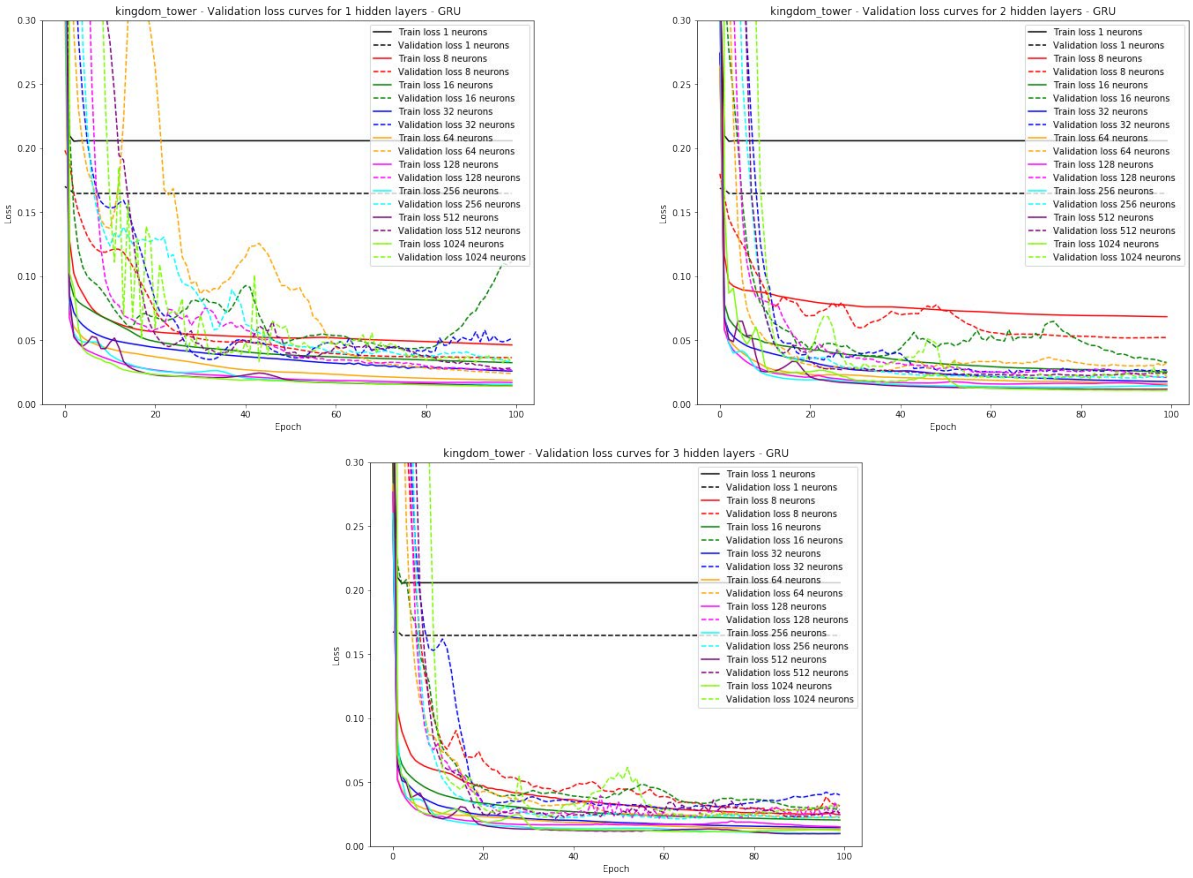


Figure 7.46: Riyadh GRU losses as the number of hidden layers increases.

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.1025	256	0.1009	256	0.0968	256
0.1026	128	0.1034	128	0.1044	512
0.1042	32	0.1071	32	0.1046	128
0.1096	512	0.1123	512	0.1134	16
0.1188	64	0.1131	64	0.1165	32

(a) LSTM

One hidden layer		Two hidden layers		Three hidden layers	
RMSE	Neurons	RMSE	Neurons	RMSE	Neurons
0.111	64	0.0988	256	0.1022	64
0.1131	128	0.0995	1024	0.1023	256
0.1141	542	0.102	128	0.1037	512
0.1274	1024	0.103	512	0.1047	128
0.1321	256	0.108	32	0.1065	1024

(b) GRU

Table 7.27: Riyadh top validation RMSE values for different numbers of layers.

meaning that the model scores do not improve with model complexity.

7.4.3 Normalization in RNN

There exist different techniques for normalizing the data throughout their use between hidden layers. This is because their distribution can change due to the nature of NN methods as they are initialized by random weights that change in time. Normalization of the data helps the models by giving them faster convergence and robustness to hyperparameter values, and it also provides some sort of regularization effect.

Batch normalization (proposed in Ioffe and Szegedy [2015]) is a technique used to normalize the data (mean zero and unit standard deviation) between hidden layers, as those can change (covariate shift) due to the random initialization of the weights, and it is believed to make learning harder for the model. It is called batch normalization because it is applied to each minibatch of data by computing the mean and standard deviation after the activation function of a previous layer, and then these are normalized with the input for the next layer. In addition, two parameters that are learnt by the model are used while training, and these allow reverting the effect of the normalization if the model learns that a non-normalized representation of the data between hidden layers is better. Although batch normalization has been used with success in CNN, it is not clear if it helps in the case of RNN, as they maintain a hidden state based on the time dimension of the data and therefore controlling the statistics over each time step can hurt the model's learning. Another drawback in batch normalization is that the estimates of the statistics become noisy as the number of samples in the mini batch grows smaller. Furthermore, it is not possible to use this technique with a batch of size one.

Another strategy that can be used to normalize the data between layers is layer normalization (proposed in Ba et al. [2016]). The idea is the same as in batch normalization, but instead of normalizing by the batch dimension of the data (using the same mean and standard deviation for all samples in the mini batch), a mean and a standard deviation is computed for each feature in a sample and used to normalize it. This strategy does not allow using a parameterization like in batch normalization, in which the effect of the normalization is reverted. The authors proposed this technique especially for RNN, as there is no need to compute new parameters at each time step, which is what would happen in a batch normalization for recurrent NN. In addition, as the normalization acts on the level of each sample, there is no problem with the size of the batch, as happens in batch normalization.

The sectors where this experiment has been performed are Antwerp Pelikaanstraat, Barcelona Regular, Wattens Inside, Los Angeles Standard (before street works) and Riyadh Sector-1. We want to gain intuition about how RNN models are affected by the normalization techniques, and this is independent of the data used. The developed models have the same architecture and the same initial weights, so they only differ in the normalization technique used: batch normalization, layer normalization or none.

	Antwerp Pelikaanstraat	Barcelona	Wattens Inside	Los Angeles Standard	Riyadh Sector-1
Batch normalization	0.1463	0.1054	0.0928	0.1059	0.0791
Layer normalization	0.1402	0.0992	0.0902	0.0974	0.0807
No normalization	0.1321	0.0987	0.0945	0.1028	0.0934

Table 7.28: Normalization technique RMSE values.

Hyperparameter	Range
Number of neurons	2^i where $i = [6, 10]$
Number of layers	2 to 5
Dropout %	$[0, 0.25, 0.5]$
Weights initializer	He normal and Glorot normal
Activation functions	Linear, sigmoid and ReLu
Batch sizes	$[32, 64, 128, 256, 672]$
Optimizer	RMSProp
Learning rates	$[0.0001, 0.01]$ divided by number of iterations.

Table 7.29: MLP hyperparameter ranges of the random search strategy.

The training losses can be seen in 7.47 and show that, of the three techniques considered, layer normalization is the one that offers smoother training and equal or lower RMSE values in the validation set.

The obtained RMSE values in the validation set are shown in Table 7.28. Although all methods scored similar RMSE values in the validation set, layer normalization is the technique that obtained lower scores in nearly all sectors considered. In addition to the smoothness of the loss while training, this makes it a proper normalization technique to use in RNN methods.

7.4.4 Hyperparameter calibration conclusions

The results from the individual experiments that were performed provide a starting point and intuition about how to process the data and the effect of the different techniques that can be used in NN methods. This reduces the number of possible values to consider in the random search strategy by lowering the computations needed in order to obtain an NN model for each sector with a low error rate on unseen data.

The considered range of values for the hyperparameters of the MLP method are presented in Table 7.29.

Table 7.30 presents the hyperparameter value ranges considered for the LSTM and GRU methods. Because of how these methods work, the numbers of layers and neurons considered are less than those considered for the MLP. Also, the table includes hyperparameters that are exclusive to these methodologies, such as the recurrent dropout, which is the percentage of dropout used for the hidden state. Another hyperparameter that is used

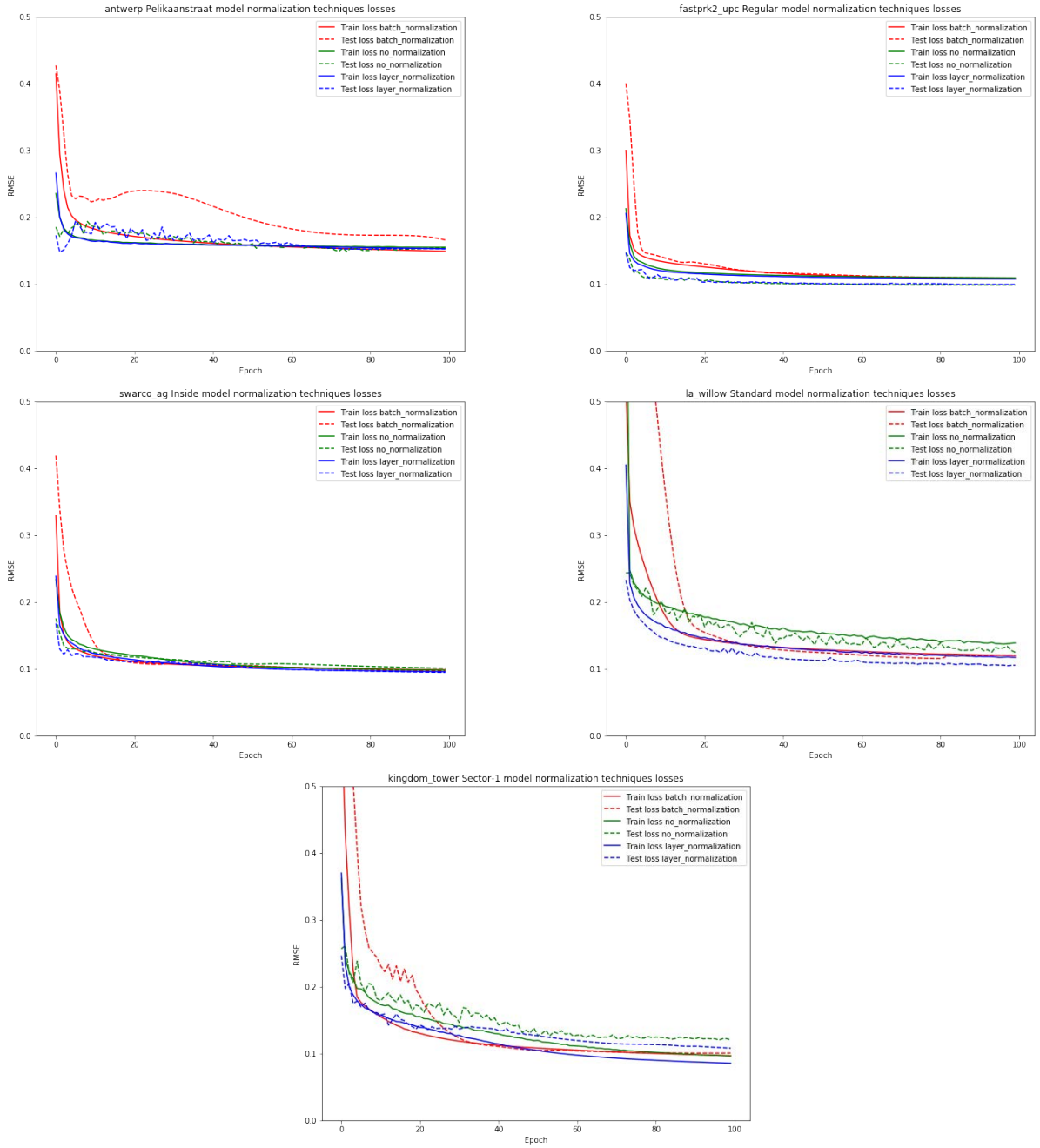


Figure 7.47: Normalization experiment losses. From top to bottom, the losses correspond to Antwerp, Los Angeles and Riyadh.

Hyperparameter	Range
Number of neurons	2^i where $i = [3, 9]$
Number of layers	1 to 3
Dropout %	$[0, 0.25, 0.5]$
Recurrent dropout %	$[0, 0.25, 0.5]$
Weights initializer	He normal and Glorot normal.
Activation functions	Linear, sigmoid and ReLu
Batch sizes	$[32, 64, 128, 256, 672]$
Optimizer	SGD with momentum
Learning rates	$[0.001, 1]$ divided by number of iterations.
Clipping norm	0.1 to 1 at steps of 0.1

(a) LSTM

Hyperparameter	Range
Number of neurons	2^i where $i = [3, 9]$
Number of layers	1 to 3
Dropout %	$[0, 0.25, 0.5]$
Recurrent dropout %	$[0, 0.25, 0.5]$
Weights initializer	He normal and Glorot normal.
Activation functions	Linear, sigmoid and ReLu
Batch sizes	$[32, 64, 128, 256, 672]$
Optimizer	RMSProp
Learning rates	$[0.00001, 0.01]$ divided by number of iterations.
Clipping norm	0.1 to 1 at steps of 0.1

(b) GRU

Table 7.30: Hyperparameter ranges of the random search strategy for recurrent NN methods.

only in the RNN methods (although it is available for all methods, as it is part of the optimizer tool) is the value used for clipping the gradient norm in case of exceedings on it (remember that this technique is used to prevent the exploding gradient phenomenon explained in Section 5.2.2.9).

7.5 NN experiments - Antwerp

Table 7.31 presents the dates of the data considered for the training, validation and test sets for each of the experiments on the Antwerp sectors. The aim of the experiments in this section is to detect the behaviour of the different NN models when training them with time series data from parking occupancy.

Sector	Training	Validation	Test
Pelikaanstraat	2018-02-01, 2019-01-19	2019-01-20, 2019-03-04	2019-03-05, 2019-04-17
Van Wesenbekestraat Zone 1	2018-12-13, 2019-10-16	2019-10-17, 2019-11-23	2019-11-24, 2019-12-31
Van Wesenbekestraat Zone 2	2018-12-13, 2019-10-16	2019-10-17, 2019-11-23	2019-11-24, 2019-12-31
Van Schoonhovestraat Zone 1	2019-02-24, 2019-10-30	2019-10-31, 2019-11-30	2019-12-01, 2019-12-31
Van Schoonhovestraat Zone 2	2019-02-24, 2019-10-30	2019-10-31, 2019-11-30	2019-12-01, 2019-12-31
Van Schoonhovestraat Zone 3	2019-02-24, 2019-10-30	2019-10-31, 2019-11-30	2019-12-01, 2019-12-31

Table 7.31: Antwerp sector intervals of dates used for the training, validation and test sets.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_endogenous_antwerp_pelikaanstraat	64	3	0.25	He normal	ReLu	672	RMSProp	0.00028786
MLP_endogenous_antwerp_vanwesenbekestraat	512	4	0.25	Glorot normal	Linear	64	RMSProp	0.00436647
MLP_endogenous_antwerp_vanwesenbekestraat2	128	3	0.25	Glorot normal	ReLu	128	RMSProp	0.00265265
MLP_endogenous_antwerp_vanSchoonhovestraat1	64	3	0	He normal	ReLu	256	RMSProp	0.00363
MLP_endogenous_antwerp_vanSchoonhovestraat2	64	4	0.25	Glorot normal	Linear	64	RMSProp	0.000418
MLP_endogenous_antwerp_vanSchoonhovestraat3	64	5	0.5	He normal	Linear	64	RMSProp	0.0002242

Table 7.32: Antwerp MLP candidate model characteristics.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_endogenous_antwerp_pelikaanstraat	0.110130	0.133542	0.150384	0.162346	0.172676	0.179238	0.153562
MLP_endogenous_antwerp_vanwesenbekestraat	0.095566	0.116724	0.125885	0.131969	0.133297	0.133658	0.124019
MLP_endogenous_antwerp_vanwesenbekestraat2	0.101069	0.124634	0.135584	0.143275	0.146959	0.148462	0.134506
MLP_endogenous_antwerp_vanSchoonhovestraat1	0.123471	0.139036	0.148395	0.154933	0.158351	0.161390	0.149402
MLP_endogenous_antwerp_vanSchoonhovestraat2	0.157954	0.204540	0.235772	0.255512	0.268585	0.280565	0.237706
MLP_endogenous_antwerp_vanSchoonhovestraat3	0.13	0.156846	0.166445	0.173264	0.176913	0.177004	0.164277

Table 7.33: Antwerp MLP candidate model RMSE values in the test set.

7.5.1 MLP

Hyperparameter values obtained by the random search strategy in order to obtain candidate MLP models for the Antwerp scenario are presented in Table7.32. The most common values are 64 neurons in each hidden layer, 3 hidden layers, and unbounded activation functions (ReLu and linear ones).

Table 7.33 presents the RMSE scores for the MLP experiments in Antwerp based on forecasting hours one to six and their mean. The error rate obtained for the first forecasting hours is higher than those from ARIMA models, but it becomes less as the forecasting steps go further. Similarly to ARIMA, the sectors with more regular patterns present fewer errors than those that are more chaotic, especially for those with a low number of parking places. In particular, the model for the sector Van Schoonhovestraat 2 obtained very high error rates at forecasts for six hours in advance, due to the uncertainty that is inherent to the sector.

Figure 7.48 presents the visualizations of the different forecasting horizon steps one to six for the experiment MLP_endogenous_antwerp_pelikaanstraat. The visualizations give the impression that the model has learnt the different patterns of the parking occupancy for days and weeks . It can be seen that the values of the forecasts are a bit lower when compared to those of the true values. It is also noticeable that, as the forecasting steps go further, the forecasted values become closer to the mean value of the time series. This effect is the model's response to the uncertainty about the future.

From now on, the visualizations will focus only on the first forecasting steps for all the sectors in order to compare them with those from the ARIMA models. Figure 7.49 presents the visualizations for the first forecasting steps for all Antwerp sectors. The visualization of the Pelikaanstraat candidate model is well adjusted to the changes in patterns due to multiple seasonalities in the data, and thus it provides lower or higher values, depending on the day of the week and after making distinctions between

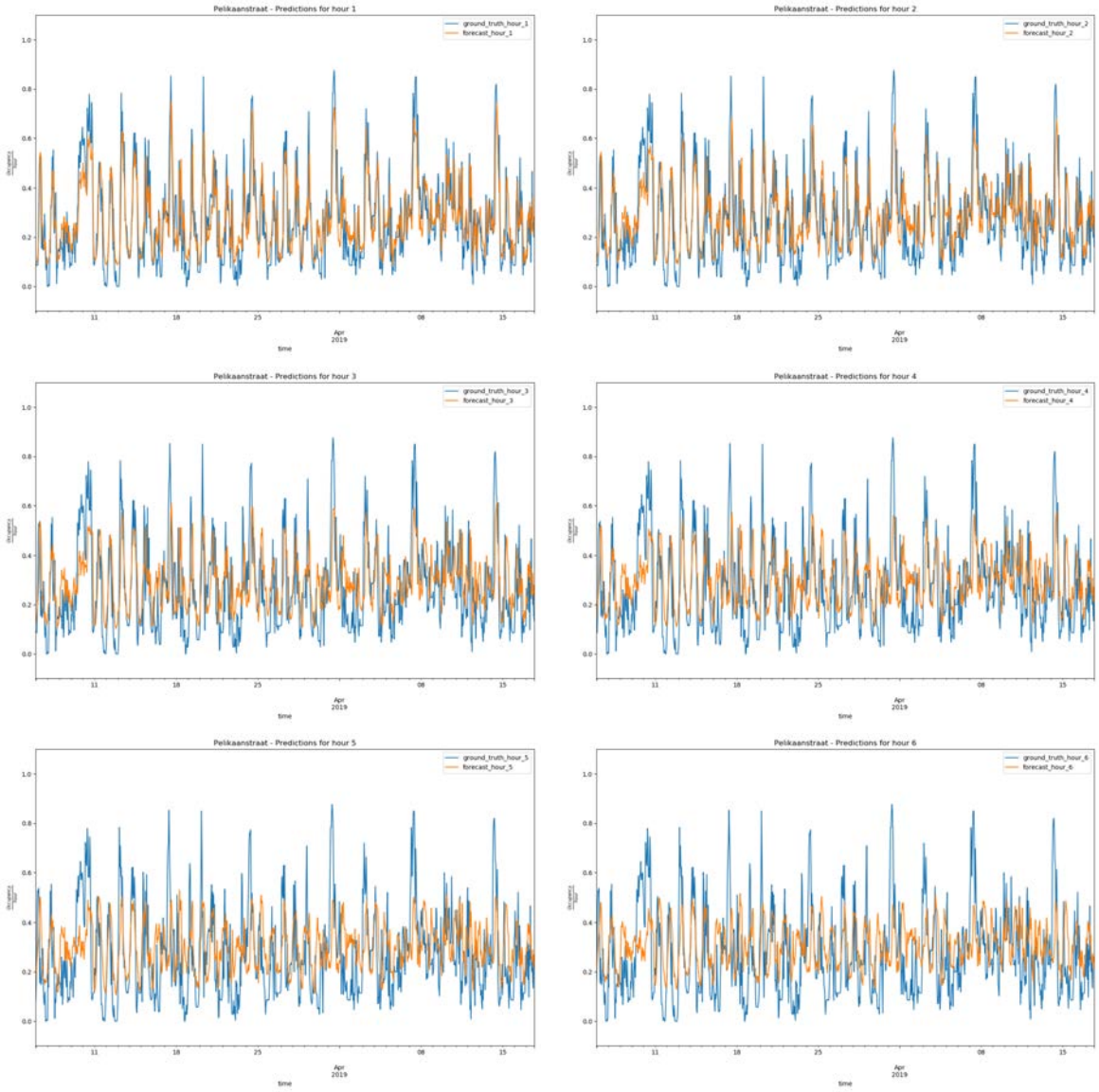


Figure 7.48: Antwerp Pelikaanstraat MLP forecast visualizations. From top left to bottom right are steps one to six

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_endogenous_antwerp_pelikaanstraat	8	1	0.25	0.25	Glorot normal	Sigmoid	64	SGD	0.02151	0.8
LSTM_endogenous_antwerp_vanwesenbekestraat	64	3	0.25	0.25	He normal	Linear	64	SGD	0.55276	0.4
LSTM_endogenous_antwerp_vanwesenbekestraat2	128	2	0.25	0	He normal	Linear	128	SGD	0.01078	0.6
LSTM_endogenous_antwerp_vanSchoonhovestraat1	8	3	0	0.25	He normal	Linear	128	SGD	0.0299967	0.6
LSTM_endogenous_antwerp_vanSchoonhovestraat2	256	3	0.5	0	Glorot normal	Linear	672	SGD	001710	0.9
LSTM_endogenous_antwerp_vanSchoonhovestraat3	32	3	0.25	0	He normal	Linear	672	SGD	0.0279009	0.6

Table 7.34: Antwerp LSTM candidate model characteristics.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_endogenous_antwerp_pelikaanstraat	0.110668	0.137267	0.153259	0.165521	0.174594	0.181250	0.155625
LSTM_endogenous_antwerp_vanwesenbekestraat	0.103780	0.126287	0.137642	0.141567	0.142330	0.144293	0.133610
LSTM_endogenous_antwerp_vanwesenbekestraat2	0.093660	0.121632	0.137392	0.148286	0.154991	0.159997	0.137875
LSTM_endogenous_antwerp_vanSchoonhovestraat1	0.113743	0.132188	0.144037	0.145389	0.153726	0.155251	0.141533
LSTM_endogenous_antwerp_vanSchoonhovestraat2	0.177184	0.214003	0.231087	0.237052	0.254416	0.258750	0.233800
LSTM_endogenous_antwerp_vanSchoonhovestraat3	0.138450	0.157465	0.166710	0.171605	0.175181	0.175716	0.164694

Table 7.35: Antwerp LSTM candidate model RMSE values in the test set.

weekdays and weekends. Similar behaviour can be seen in the other candidate model forecasts. One thing to notice is that, in general, the forecasted values are lower than those of the real occupancy, which is noticeable for those sectors where days have sudden high values for short intervals of time. The only sector in which the visualizations present notable discrepancies between the forecast values and the real occupancy is the sector Van Schoonhovestraat Zone 2. As mentioned in other sections, this sector has a low number of parking places and its behaviour was identified as a random walk after performing the ARIMA analysis.

7.5.2 LSTM

Table 7.34 presents the hyperparameter values of the candidate models from the LSTM methodology in the city of Antwerp. It is of interest that the linear activation function is the most common one, meaning that non-linearities offered by NN methods do not play a role when the LSTM models are applied to this data.

Table 7.35 presents the error scores for the MLP experiments in Antwerp. Similar results have been obtained for all the models with mean RMSE values of around 0.14. The Van Schoonhovestraat 2 model is the one with the highest error rate, but this is not unexpected because it is already known that it behaves like a random walk.

The visualizations for the different horizon steps one to six are presented in Figure 7.50 for the sector Pelikaanstraat in order to observe the behaviour of the forecasts as the forecasting steps increase. Similarly to MLP (see Figure 7.48), the LSTM forecasts the mean as the prediction goes further in time.

Figure 7.51 presents the visualizations for the first forecasting steps from the Antwerp models. The visualizations show that models learn the patterns throughout the days and give accurate forecasts for those sectors with uniform patterns. It is noticeable that for

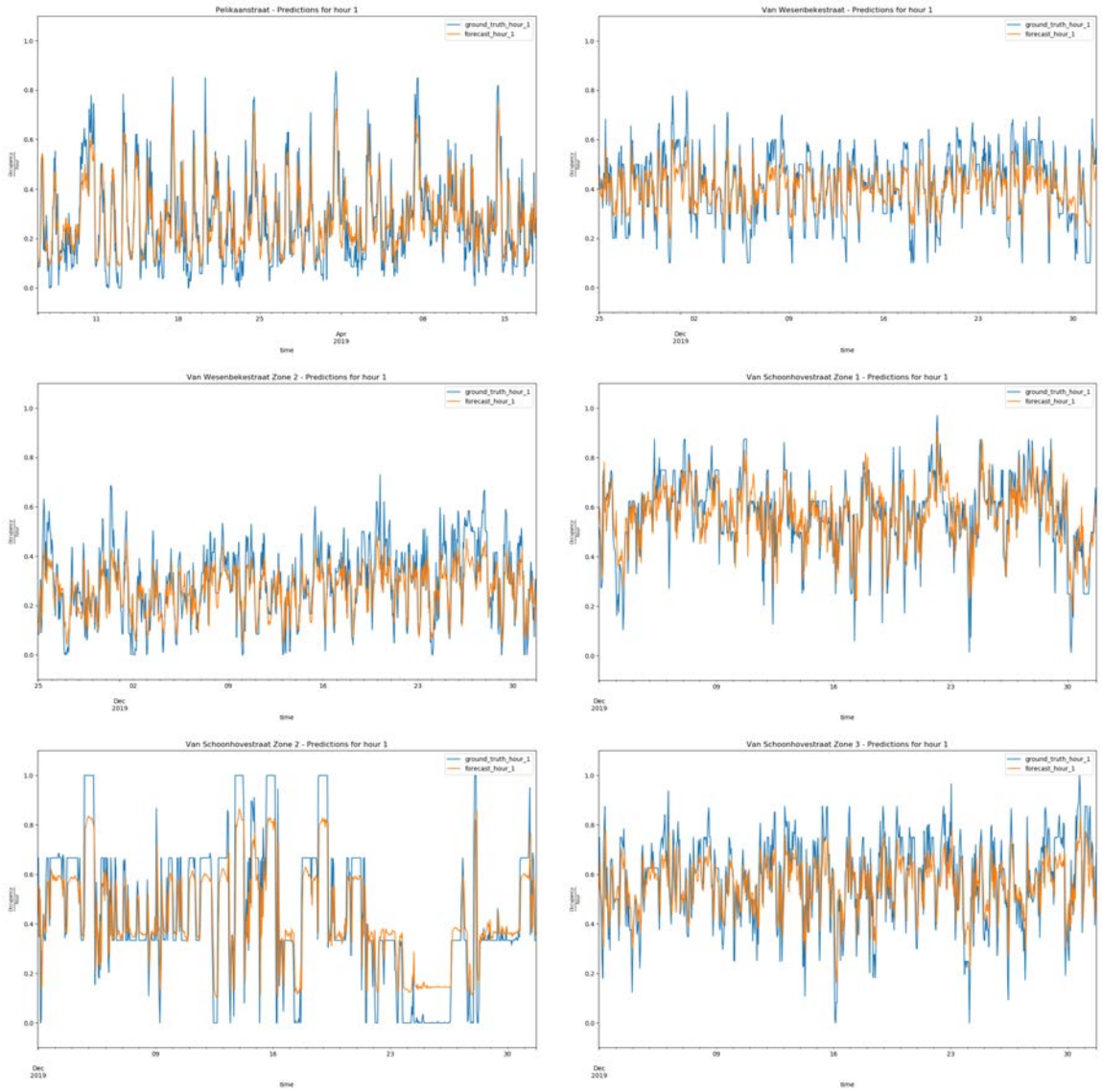


Figure 7.49: Antwerp MLP candidate model first step forecast plots. The first row are the sectors Pelikaanstraat and Van Wesenbekestraat Zone 1. Second row are sectors Van Wesenbekestraat Zone 2 and Van Schoonhovestraat Zone 1. The third row are the sectors Van Schoonhovestraat Zone 2 and Van Schoonhovestraat Zone 3.

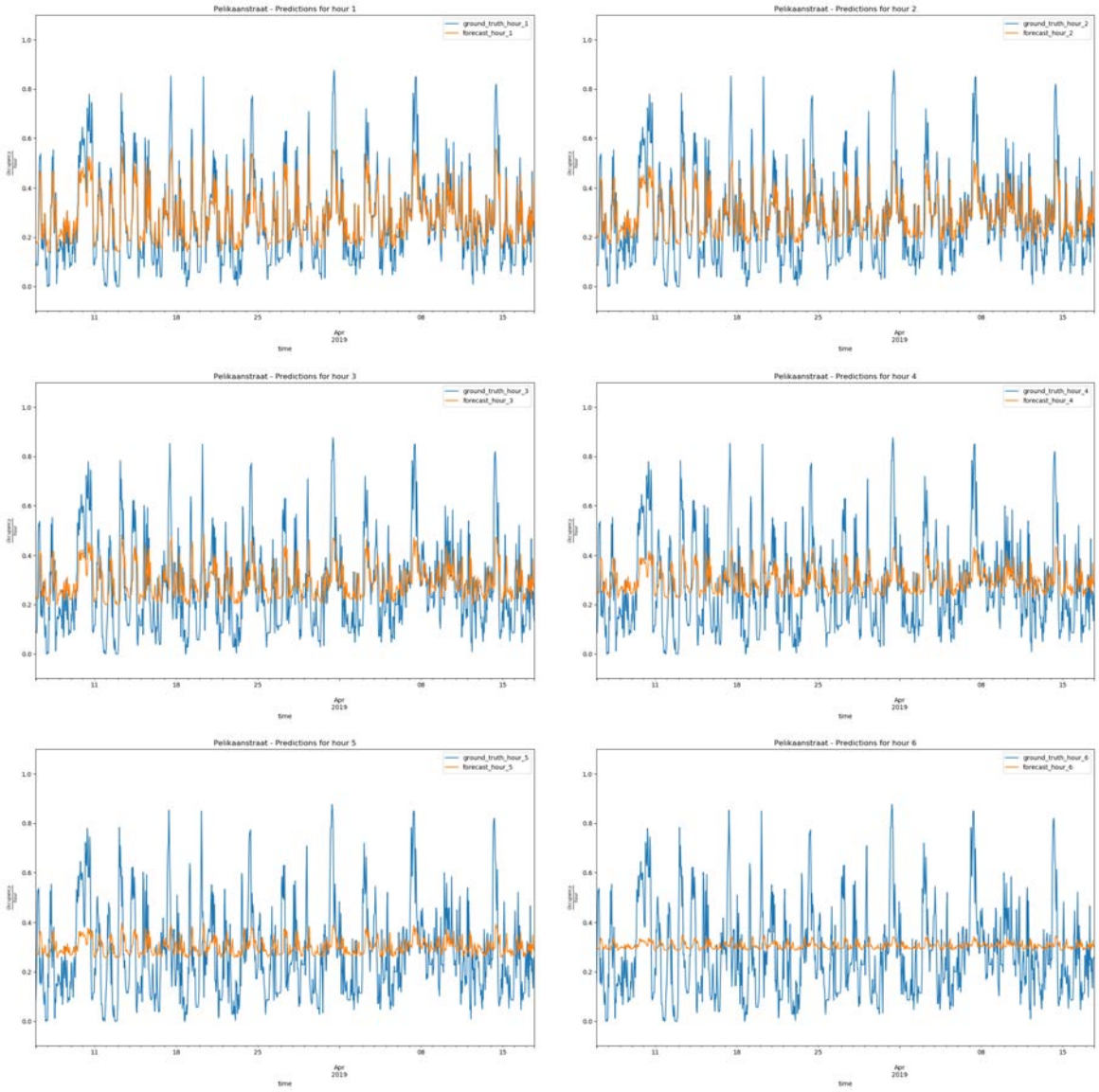


Figure 7.50: Antwerp Pelikaanstraat LSTM forecast visualizations. From top left to bottom right are steps one to six

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_endogenous_antwerp_pelikaanstraat	32	1	0.25	0	He normal	Sigmoid	672	RMSProp	0.00001	0.2
GRU_endogenous_antwerp_vanwesenbekestraat	16	2	0.25	0	He normal	Linear	256	RMSProp	0.0001271	0.9
GRU_endogenous_antwerp_vanwesenbekestraat2	256	3	0.25	0	He normal	Sigmoid	672	RMSProp	0.00100	0.2
GRU_endogenous_antwerp_vanschoonhovestraat1	128	3	0.25	0	He normal	ReLu	128	RMSProp	0.00028981	0.5
GRU_endogenous_antwerp_vanschoonhovestraat2	32	1	0.25	0	Glorot normal	Sigmoid	128	RMSProp	0.0000609171	0.4
GRU_endogenous_antwerp_vanschoonhovestraat3	8	1	0	0.25	Glorot normal	Linear	32	RMSProp	0.00176222	0.7

Table 7.36: Antwerp GRU candidate model characteristics.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_endogenous_antwerp_pelikaanstraat	0.113561	0.135125	0.151318	0.165038	0.173547	0.182441	0.157462
GRU_endogenous_antwerp_vanwesenbekestraat	0.103207	0.124504	0.131194	0.137746	0.142796	0.132778	0.129447
GRU_endogenous_antwerp_vanwesenbekestraat2	0.099607	0.120591	0.128654	0.138555	0.139059	0.141712	0.129501
GRU_endogenous_antwerp_vanschoonhovestraat1	0.119425	0.134792	0.141594	0.144979	0.146056	0.148689	0.139435
GRU_endogenous_antwerp_vanschoonhovestraat2	0.163100	0.201986	0.229196	0.245610	0.258182	0.265528	0.229798
GRU_endogenous_antwerp_vanschoonhovestraat3	0.1280	0.15256	0.162955	0.167865	0.169857	0.171224	0.15943

Table 7.37: Antwerp GRU candidate model RMSE values in the test set.

sectors with more variability, the models cannot predict peaks of high/low occupancy, which can be seen clearly in the Pelikaanstraat visualization.

7.5.3 GRU

Table 7.36 presents the hyperparameter values of the candidate models for the GRU methodology in the city of Antwerp. The most common hyperparameter values are the non-usage of recurrent dropout for nearly all models while using a 25% dropout for the units in the hidden layers. Unlike LSTM, the GRU models for Antwerp make use of non-linear activation functions.

Table 7.37 presents the RMSE for the GRU experiments in Antwerp, from one to six forecast steps and their mean. Notice that, compared to models from previous methodologies, the error increases slowly as the forecasting steps increase. Errors are a bit higher than those from LSTM for the first forecasting steps. Also, similar to the other methods, the model from Van Schoonhovestraat Zone 2 shows the highest errors.

The visualizations for forecasts at the different horizon steps one to six are presented in Figure 7.52 for the Pelikaanstraat GRU model. At first hand, the behaviour is similar to those of the MLP and LSTM models, with decay towards the mean as the forecasting horizon increases. However, GRU shows slower decay when compared to the other methods.

Figure 7.53 presents the visualizations for the first forecasting steps of all GRU models for the sectors in Antwerp . The visualizations are surprising, because the forecasts for some sectors like Pelikaanstraat are very accurate, while the model for others (Van Schoonhovestraat Zone 1) seems to perform exponential smoothing. It is interesting how the model has captured the level changes in the sector Van Schoonhovestraat Zone 2 while maintaining the same occupancy value for successive hours, which is something that did not happen with the MLP and LSTM models for that sector. Forecasts from the

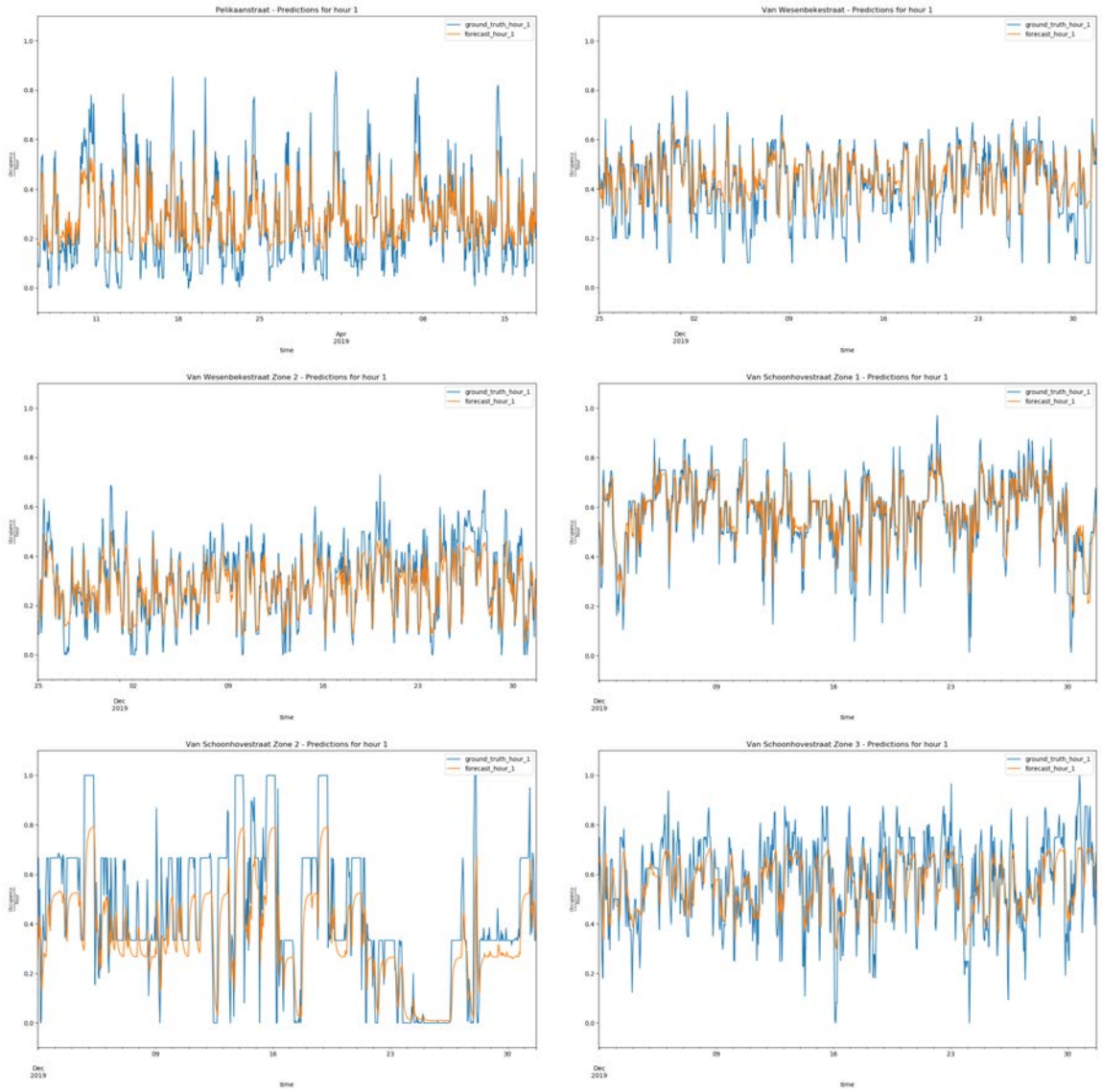


Figure 7.51: Antwerp LSTM candidate model first forecasting step plots. The first row are the sectors Pelikaanstraat and Van Wesenbekestraat Zone 1. Second row are sectors Van Wesenbekestraat Zone 2 and Van Schoonhovestraat Zone 1. The third row are the sectors Van Schoonhovestraat Zone 2 and Van Schoonhovestraat Zone 3.

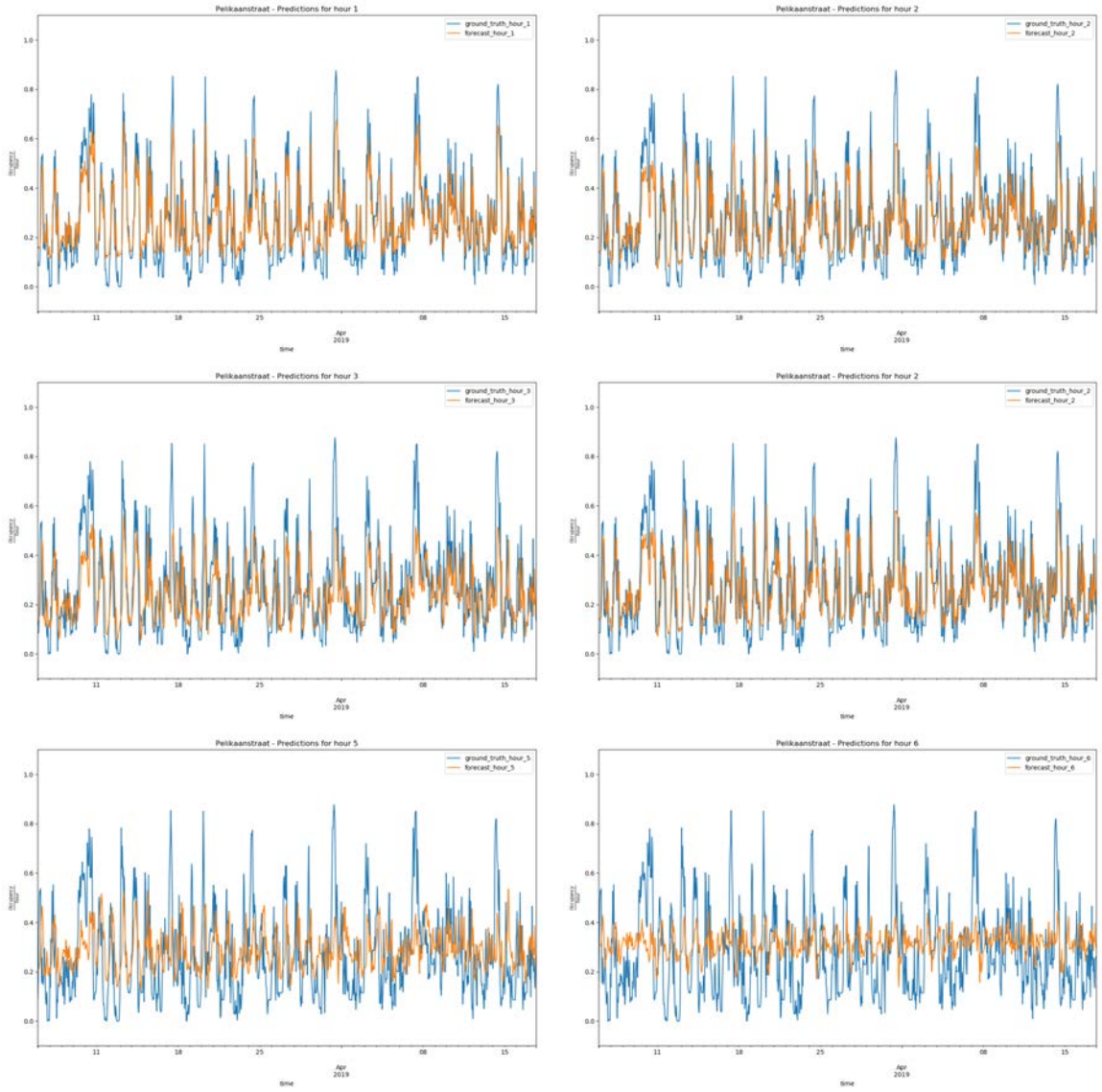


Figure 7.52: Antwerp Pelikaanstraat GRU forecast visualizations. From top left to bottom right are steps one to six

model for the sector Van Schoonhovestraat Zone 3 replicate the most recent occupancy, but with a lower value. Thus, it acts like a naive forecast (the next forecast will be the same as the actual value) but with lower "power", as it does not achieve the same level of peaks and valleys.

7.6 NN experiments - Barcelona

Table 7.38 presents the date range for the training, validation and test sets for each of the experiments on the sectors in the city of Barcelona. With the experiments presented in this section, the goal is to study how well NN models can capture time series data that consists of two well-defined distributions in which one corresponds to the weekdays and the other to the weekends.

7.6.1 MLP

Table 7.39 presents the hyperparameter values for the candidate model for the Regular sector in Barcelona, and Table 7.40 shows the RMSE achieved by the model at forecasts one to six.

The visualizations of the forecasts of the candidate MLP model for the Barcelona sector (Figure 7.54) show that the model has learnt the patterns of the occupancy for the different horizon steps, although it fails at the occupancy values for the weekend and offers lower peak levels at the sixth forecasting step.

7.6.2 LSTM

The Barcelona LSTM candidate model hyperparameter values are presented in Table 7.41. Compared to the MLP values, the model has fewer neurons while using recurrent dropout and the ReLu activation function.

RMSE values for the LSTM model for the Regular sector in Barcelona are shown in Table 7.42. At each forecasting step, the error seems to increase more or less linearly.

The visualizations of the forecasts at different horizon steps one to six are presented in Figure 7.55 for the model trained with the data from the Regular sector in Barcelona. Forecasts show that the model is able to capture the common patterns and gives lower

Sector	Training	Validation	Test
Regular	2018-01-08, 2019-08-09	2019-08-10, 2019-10-20	2019-10-21, 2019-12-31

Table 7.38: Barcelona sector intervals of dates used for the training, validation and test sets.

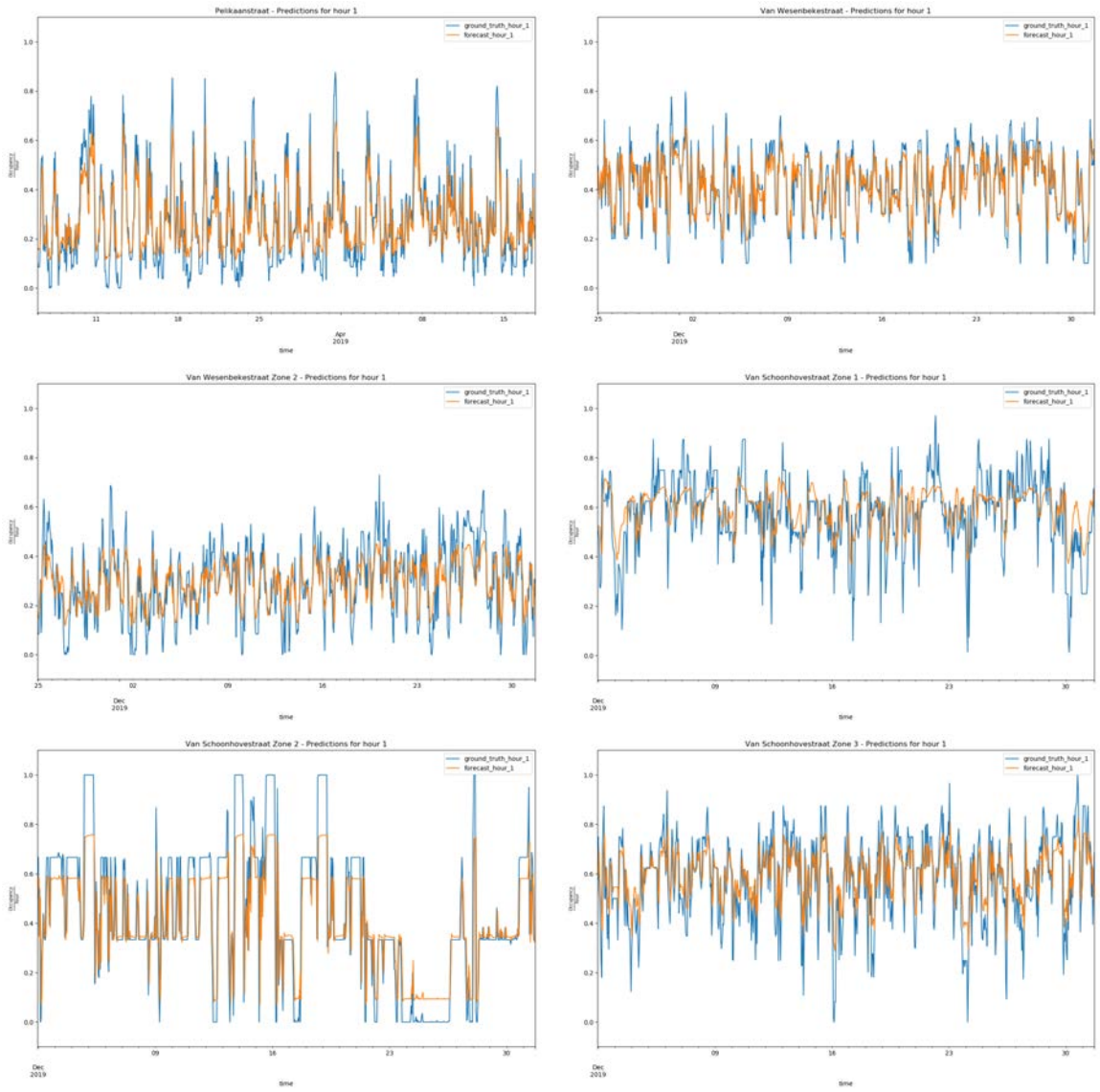


Figure 7.53: Antwerp GRU candidate model first forecasting step plots. The first row are the sectors Pelikaanstraat and Van Wesenbekestraat Zone 1. Second row are the sectors Van Wesenbekestraat Zone 2 and Van Schoonhovestraat Zone 1. The third row are the sectors Van Schoonhovestraat Zone 2 and Van Schoonhovestraat Zone 3.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_endogenous_barcelona_regular	256	4	0.5	He normal	Linear	672	RMSProp	0.00282544

Table 7.39: Barcelona MLP candidate model characteristics.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_endogenous_barcelona_regular	0.092727	0.109607	0.119814	0.125462	0.130421	0.135289	0.119841

Table 7.40: Barcelona MLP candidate model RMSE values in the test set.

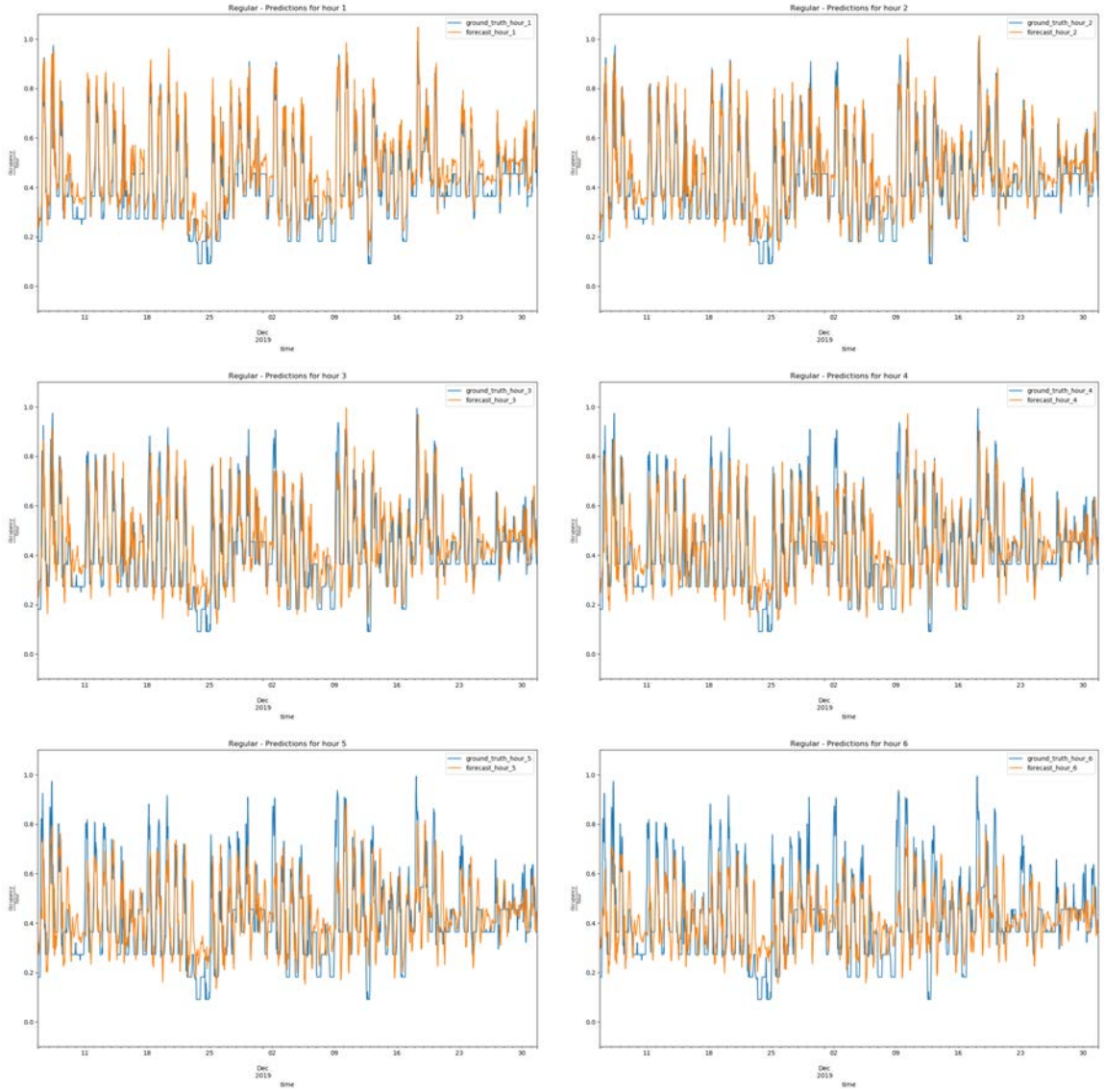


Figure 7.54: Barcelona Regular sector MLP model forecast visualizations. From top left to bottom right are steps one to six

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_endogenous_barcelona_regular	16	2	0	0.5	He normal	ReLu	256	SGD	0.0542547	0.4

Table 7.41: Barcelona LSTM candidate model characteristics for Regular sector.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_endogenous_barcelona_regular	0.100250	0.113873	0.128788	0.146382	0.164747	0.175187	0.141719

Table 7.42: Barcelona LSTM candidate model RMSE values in the test set.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_endogenous_barcelona_regular	256	2	0.25	0	Glorot normal	Sigmoid	64	RMSProp	0.00001291478	0.9

Table 7.43: Barcelona GRU candidate model characteristics.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_endogenous_barcelona_regular	0.096280	0.114129	0.130481	0.142458	0.148959	0.158079	0.133832

Table 7.44: Barcelona GRU candidate model RMSE values in the test set.

occupancy levels for weekends, but at one forecasting step it overestimates the occupancy levels. As the forecasting steps increase, the occupancy levels become the mean value of the series.

7.6.3 GRU

Table 7.43 presents the hyperparameter values for the GRU experiment in the Regular sector of Barcelona. It has the same number of hidden layers as the LSTM model and the same number of neurons as the MLP. Unlike LSTM, it has dropout on the input neurons and uses the sigmoid activation function.

The RMSE values presented in Table 7.44 show that the GRU candidate model scores in between the MLP and LSTM candidate models, as the former has lower values and the latter has higher ones.

The Barcelona GRU model visualizations at the different horizon steps one to six are presented in Figure 7.56. Unlike LSTM, GRU captures the levels of the series better, but it overestimates the occupancy at the weekends. As the prediction goes further, the GRU model offers a smoothed version of the real time series.

7.7 NN experiments - Wattens

Table 7.45 presents the date range for the training, validation and test sets for each of the experiments presented for the sectors in the city of Wattens. The goal of this scenario is to ascertain the difference between the models of two sectors in the same zone with the same usage but that differ in the number of places and (possibly) in their users.

7.7.1 MLP

The characteristics of the MLP candidate models obtained by random search for the Wattens sectors are presented in Table 7.46. Although the time series from both sectors are similar (as shown in 3.5.3), the obtained models are quite different.

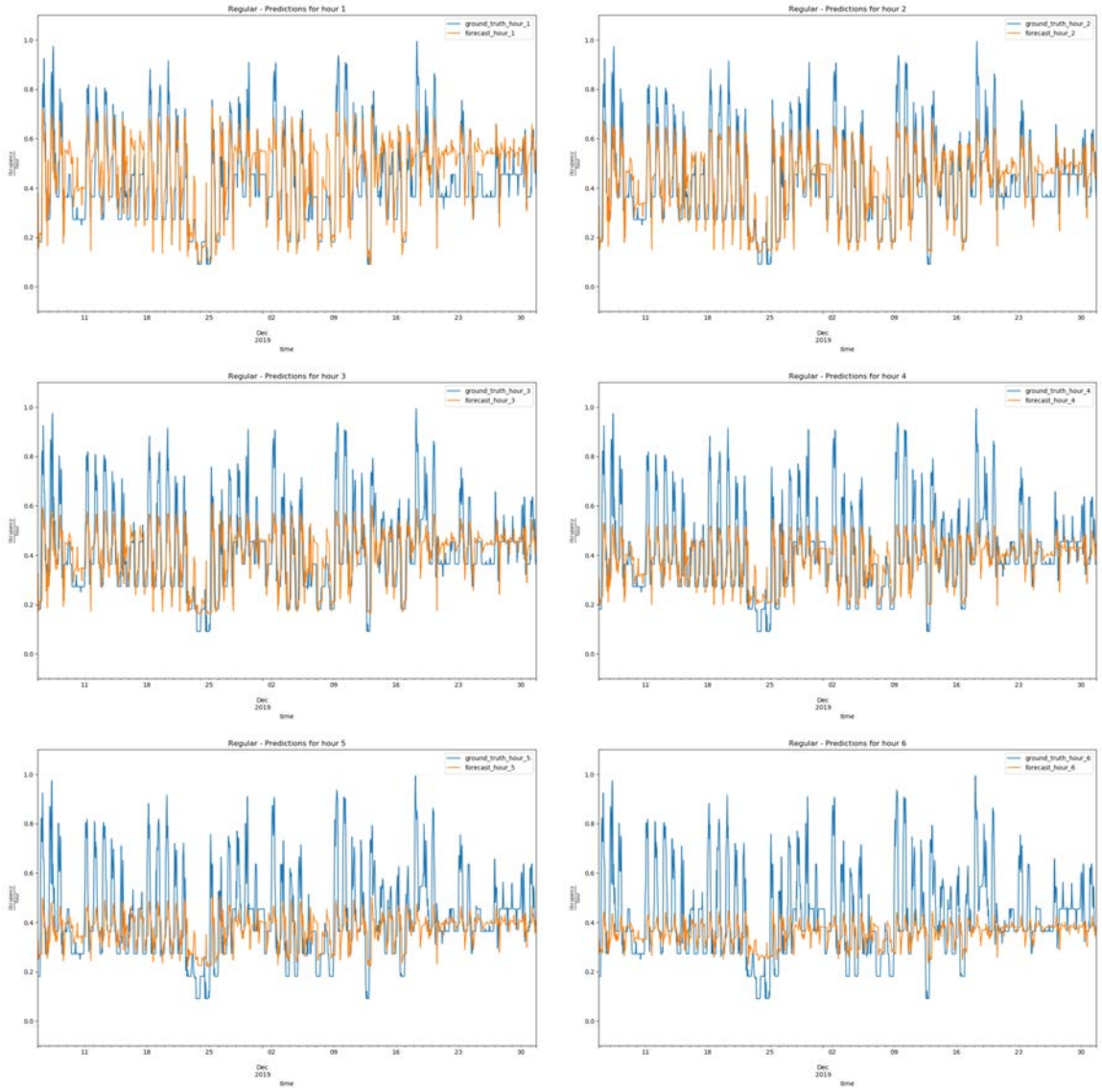


Figure 7.55: Barcelona Regular LSTM forecast visualizations. From top left to bottom right are steps one to six.

Sector	Training	Validation	Test
Inside	2018-03-01, 2019-02-23	2019-02-24, 2019-04-09	2019-04-10, 2019-05-23
Outside	2018-03-01, 2019-02-23	2019-02-24, 2019-04-09	2019-04-10, 2019-05-23

Table 7.45: Wattens sector intervals of dates used for the training, validation and test sets.

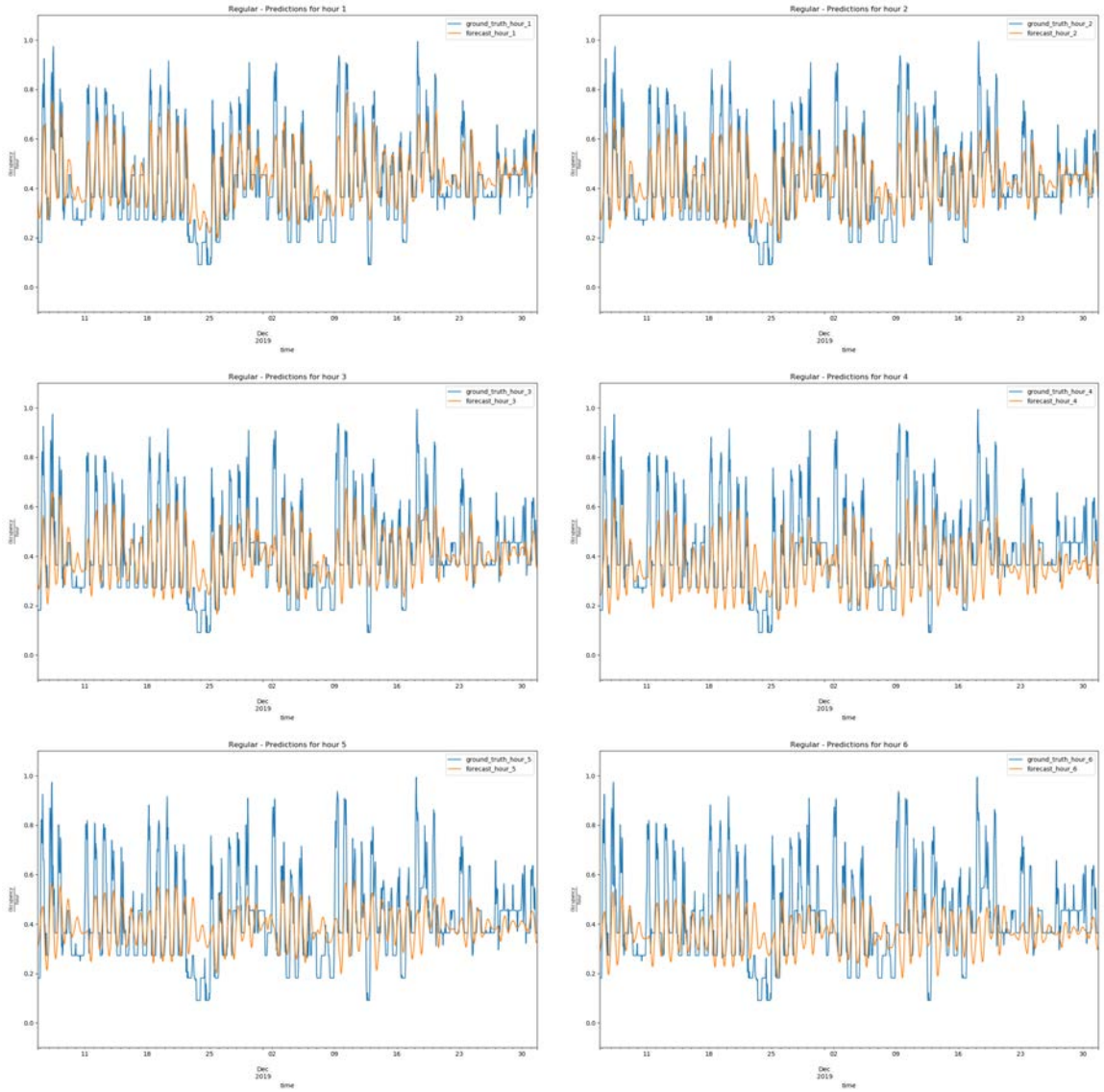


Figure 7.56: Barcelona experiment forecast visualizations. From top left to bottom right are steps one to six

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_endogenous_wattens_inside	128	1	0.25	He normal	ReLu	256	RMSProp	0.00386
MLP_endogenous_wattens_outside	256	5	0.5	He normal	Sigmoid	256	RMSProp	0.004920.00400

Table 7.46: Wattens MLP candidate model characteristics.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_endogenous_wattens_inside	0.056937	0.081595	0.101219	0.116856	0.127939	0.138484	0.107350
MLP_endogenous_wattens_outside	0.167275	0.168808	0.160449	0.161442	0.173651	0.192762	0.173874

Table 7.47: Wattens MLP candidate model RMSE values in the test set.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_endogenous_wattens_inside	16	2	0.25	0	He normal	Linear	256	SGD	0.00483981	0.8
LSTM_endogenous_wattens_outside	32	2	0	0	He normal	Sigmoid	64	SGD	0.39419	0.2

Table 7.48: Wattens LSTM candidate model hyperparameter values.

Table 7.47 presents the scores for the MLP experiments in Wattens. It is surprising how the model for the Outside sector achieved a higher error rate than the candidate model for the Inside sector, although both sectors presented similar time series. It would be interesting to find out if this is repeated in the results from recurrent methods.

Figure 7.57 offers visualizations of the MLP candidate model for the Inside sector in Wattens at different forecasting horizons (one to six). The model is able to capture the different patterns in the data, like weekday and weekend occupancies, although its forecast overestimates the actual weekend occupancy. Unlike some other previous models, the forecast values do not become the mean as the forecasting steps go further, thus indicating that the model is able to output accurate forecasts for the parking occupancy six hours in advance.

Figure 7.58 presents the visualizations for the Outside sector in Wattens for different time horizons (one to six). Similarly to the Inside sector, the visualizations show that the model is able to reproduce the patterns that occur throughout the day and the differences between weekdays and weekends. The visualizations do not provide any clue that explains why the RMSE is far higher in the Outside model when compared to the RMSE obtained from the Inside model.

7.7.2 LSTM

Table 7.48 presents the hyperparameter values for the LSTM candidate models for the Wattens experiments. Both models are based on a low number of neurons and share the same number of layers.

Table 7.49 presents the RMSE values for the LSTM candidate models for the experiments in the Wattens scenario. It can be observed that, as the forecasting steps increase, the differences become smaller. In other words, the difference between the first and second steps of the Inside sector is nearly 0.035, but between the fifth and sixth forecasting steps it is 0.014. The Outside sector presents a huge increase in the error between steps one, two and three.

The visualizations in Figure 7.59 present the forecasts of the LSTM candidate model for the experiment LSTM_endogenous_wattens_inside. It can be observed that the

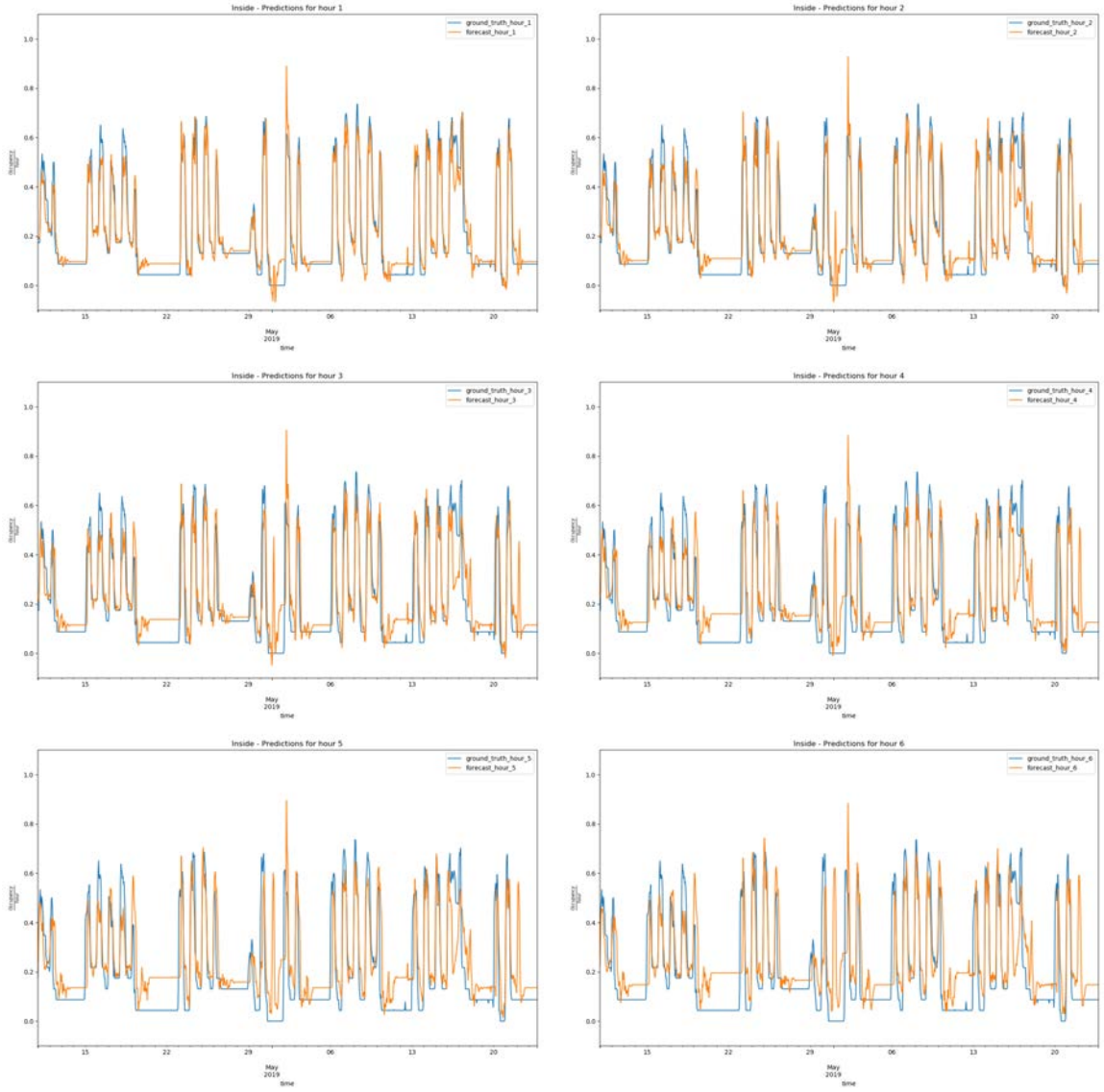


Figure 7.57: Wattens Inside sector MLP candidate model forecast visualizations. From top left to bottom right are steps one to six

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_endogenous_wattens_inside	0.063618	0.098741	0.126547	0.149175	0.162757	0.176160	0.134990
LSTM_endogenous_wattens_outside	0.095564	0.132638	0.162885	0.186857	0.191105	0.220024	0.169323

Table 7.49: Wattens LSTM candidate model RMSE values in the test set.

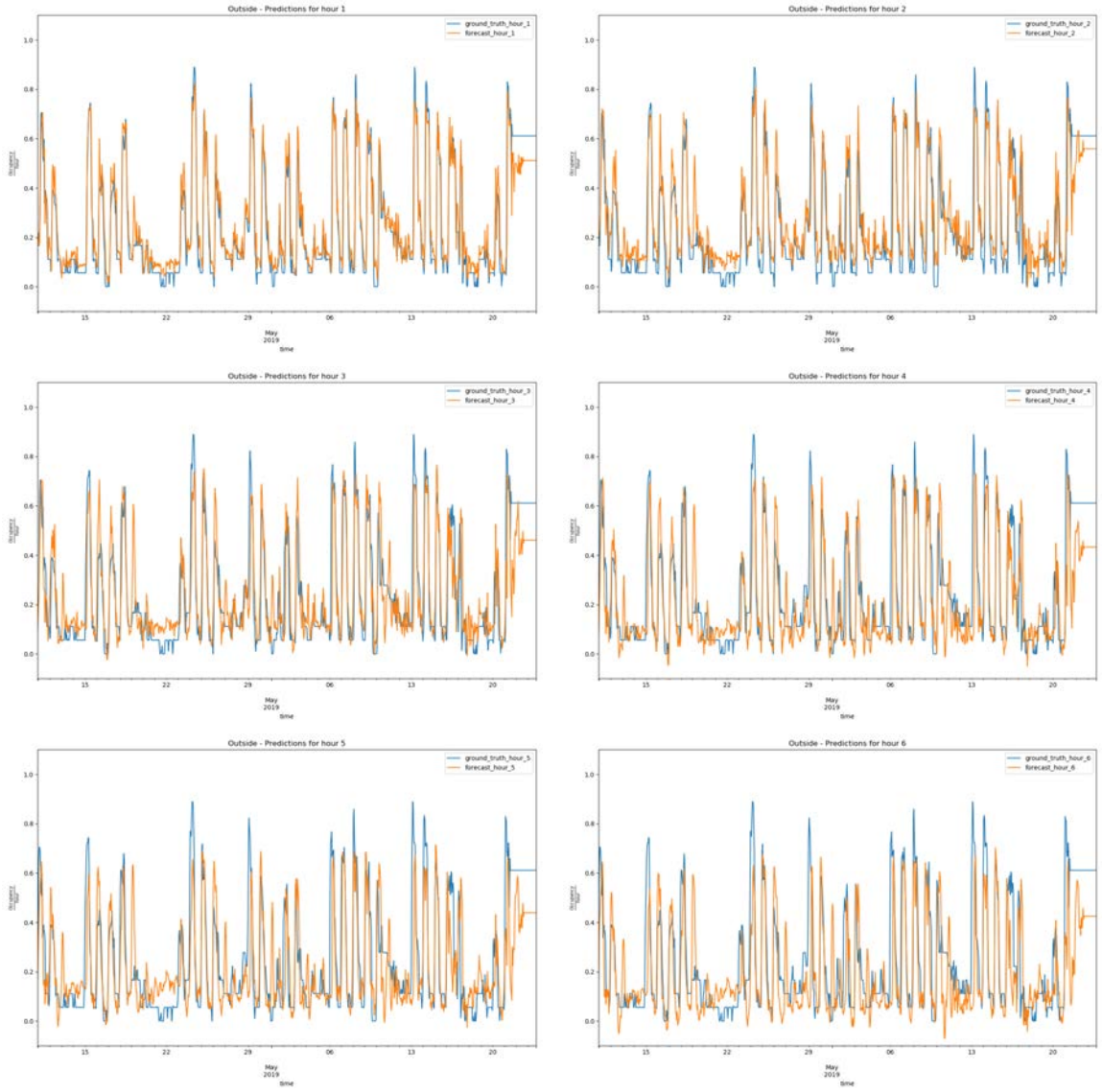


Figure 7.58: Wattens MLP candidate model for the Outside sector forecasts from one to six.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_endogenous_wattens_inside	32	2	0.25	0	He normal	Sigmoid	672	RMSProp	0.000373002	0.6
GRU_endogenous_wattens_outside	64	2	0.25	0	He normal	Sigmoid	128	RMSProp	0.00370607	0.8

Table 7.50: Wattens GRU candidate model hyperparameter values.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_endogenous_wattens_inside	0.056622	0.079935	0.100578	0.116191	0.127633	0.139902	0.107230
GRU_endogenous_wattens_outside	0.083591	0.116279	0.139536	0.156237	0.165543	0.174135	0.142176

Table 7.51: Wattens GRU candidate model RMSE values in the test set.

model provides good forecasts, especially in the first steps; but at higher horizon steps the model overestimates the parking occupancy at the weekends, giving the impression (at the sixth forecasting step) that it outputs occupancy for another day of the weekday instead of for a weekend day.

Figure 7.60 presents the visualizations for the considered time horizons (one to six) for the LSTM candidate model trained on the data from the Outside sector in Wattens. The visualizations for the first forecasting step from the Outside sector show that the model has been able to learn the patterns across the weeks, but it fails by underestimating the occupancy, as it forecasts occupancies below 0. As the forecasting steps increase, the effect of having occupancies below zero continues to happen, with the exception of time step five.

7.7.3 GRU

The GRU candidate model hyperparameter values are presented in Table 7.50 for the Wattens experiments. Both models are similar in that they use a small number of neurons, two layers and the same activation function.

Table 7.51 shows the candidate model RMSE values for the sectors in Wattens. Similarly to the LSTM candidate models, the candidate model for the Outside sector presents a higher increase in the RMSE values between the first and later forecasting steps. At later prediction steps, the error does not increase as much as in the LSTM candidate model.

The visualizations of the forecasts at different horizon steps one to six are presented in Figure 7.61 for the Inside sector. The visualizations show that the candidate model offers accurate forecasts for nearly all the forecasting steps. Similarly to LSTM, weekend occupancies are overestimated, but to a lower degree when compared to LSTM – and it is only notable for forecasts at the sixth step.

Figure 7.62 contains the visualizations at different forecasting steps of the occupancy values provided by the GRU candidate model for the Outside sector in Wattens. Predictions are quite similar between the visualizations but noticeable differences can be observed on the weekends, where the model predicts a higher occupancy.

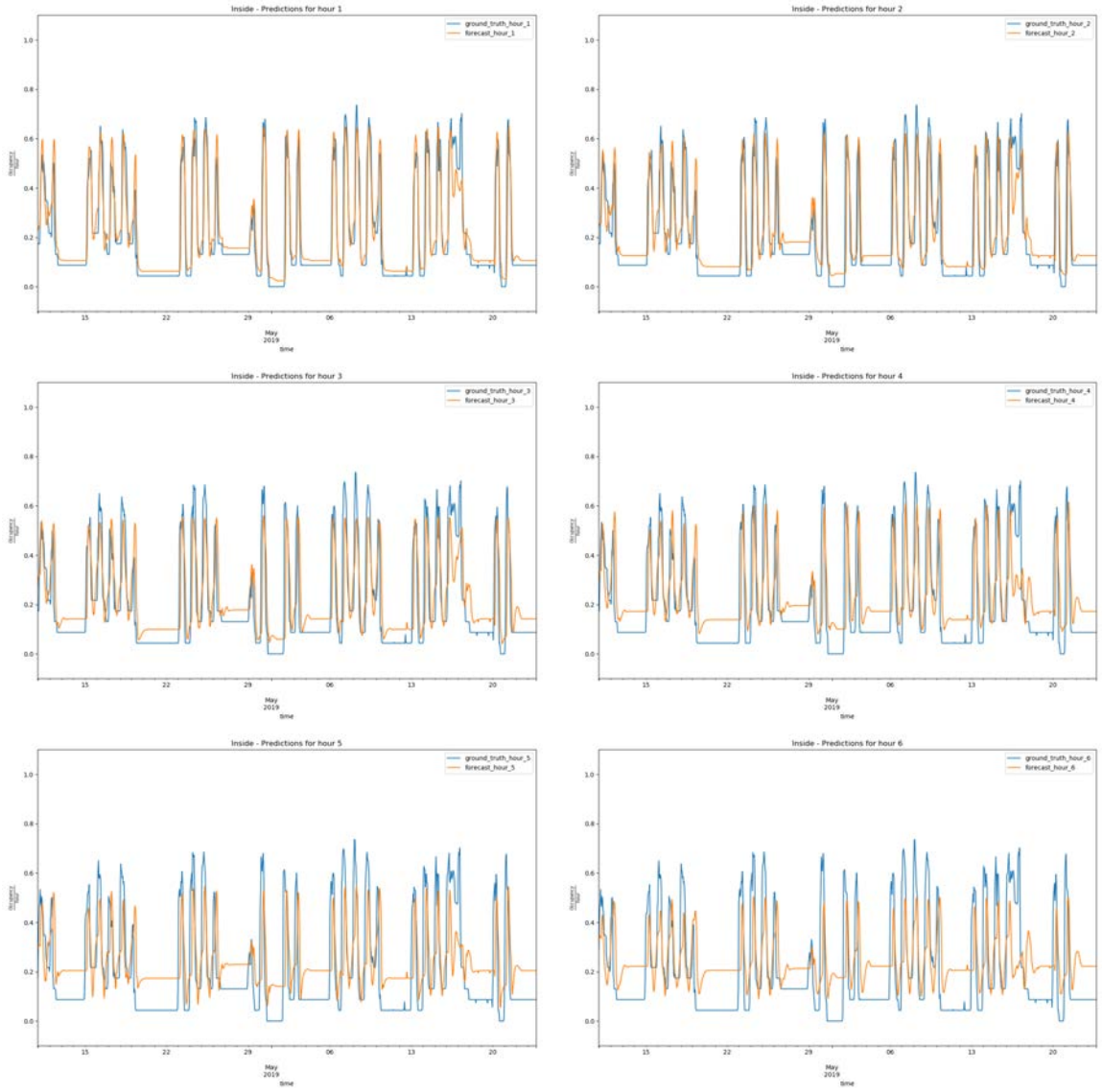


Figure 7.59: Wattens Inside LSTM forecast visualizations. From top left to bottom right are forecasting steps one to six.

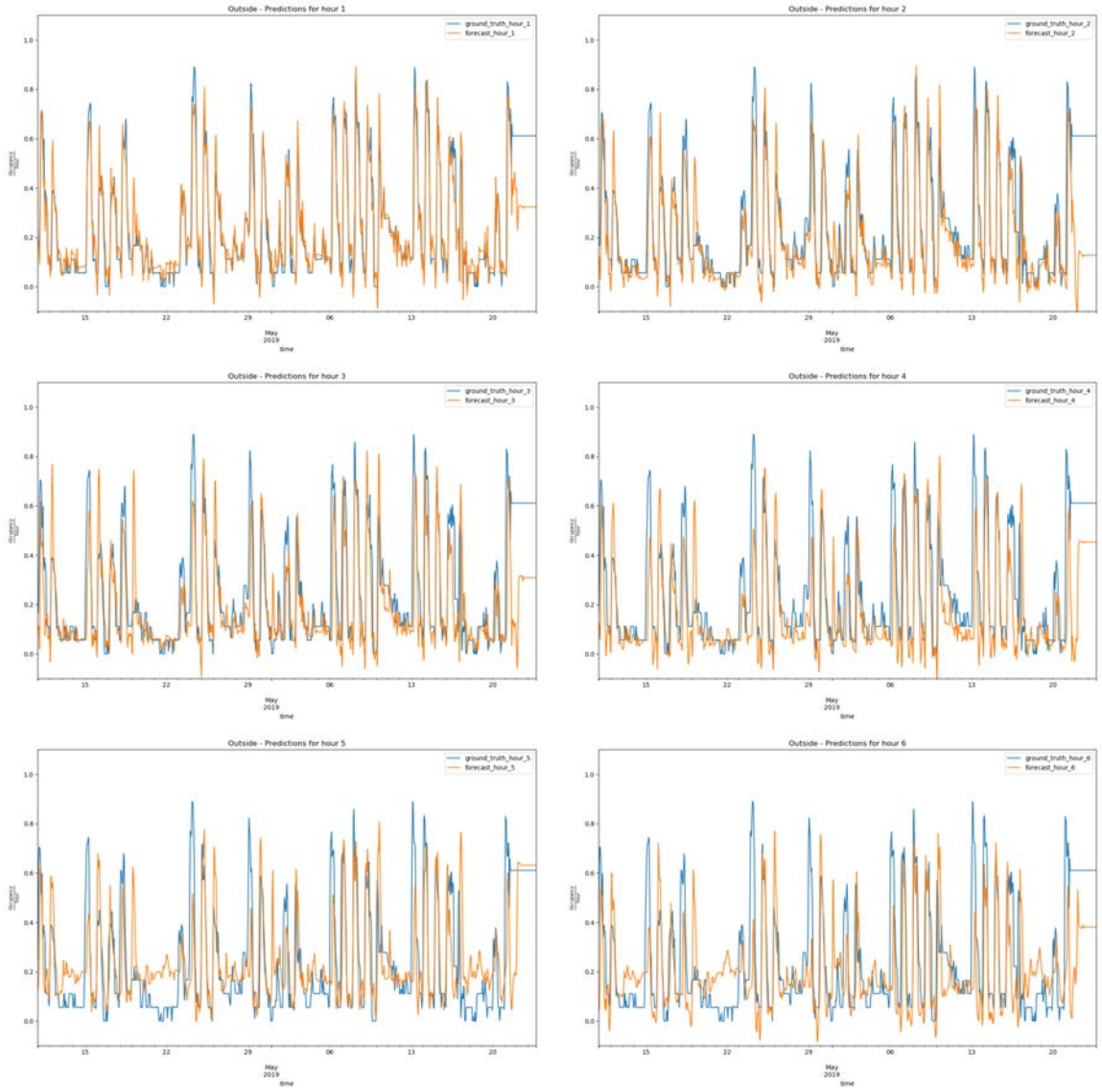


Figure 7.60: Wattens Outside LSTM forecast visualizations. From top left to bottom right are forecasting steps one to six.

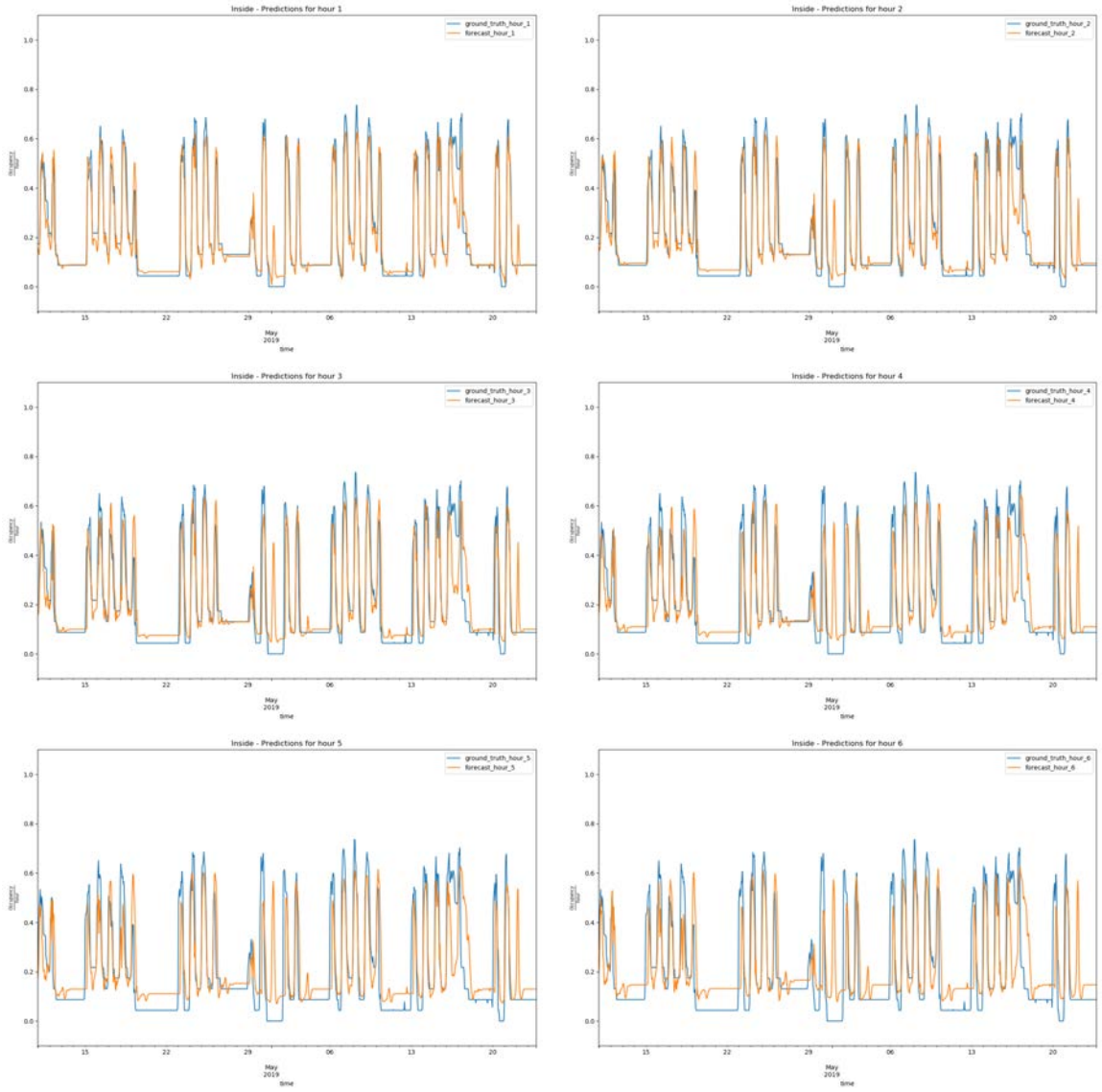


Figure 7.61: Wattens Inside GRU forecast visualizations. From top left to bottom right are forecasting steps one to six.

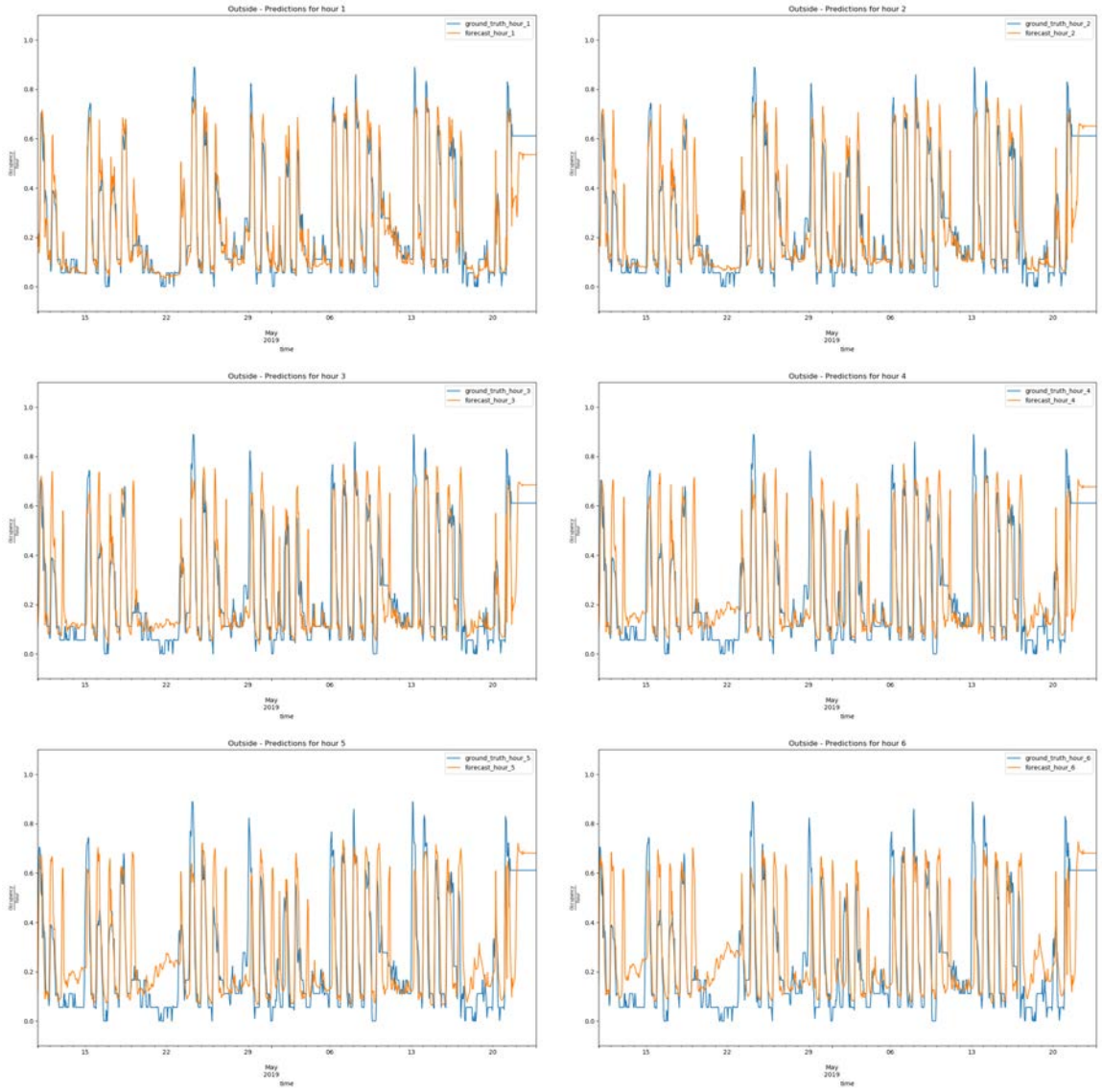


Figure 7.62: Wattens Outside GRU forecast visualizations. From top left to bottom right are forecasting steps one to six.

Sector	Training	Validation	Test
Standard	2018-04-01, 2019-07-01	2019-07-02, 2019-08-27	2019-08-28, 2019-10-23
Permit	2018-04-01, 2019-07-01	2019-07-02, 2019-08-27	2019-08-28, 2019-10-23
ZipCar	2018-04-01, 2019-07-01	2019-07-02, 2019-08-27	2019-08-28, 2019-10-23
Disabled	2018-04-01, 2019-07-01	2019-07-02, 2019-08-27	2019-08-28, 2019-10-23

Table 7.52: Los Angeles sector intervals of dates used for the training, validation and test sets.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_endogenous_LA_Standard	1024	4	0.5	He normal	ReLu	128	RMSProp	0.00279
MLP_endogenous_LA_Permit	64	4	0.25	Glorot normal	Linear	672	RMSProp	0.00725562
MLP_endogenous_LA_ZipCar	64	2	0.25	Glorot normal	Sigmoid	672	RMSProp	0.00137
MLP_endogenous_LA_Disabled	256	5	0.5	Glorot normal	ReLu	256	RMSProp	0.00277038

Table 7.53: Los Angeles MLP candidate model hyperparameter values.

7.8 NN experiments - Los Angeles

Table 7.52 presents the date range for the training, validation and test sets for each of the experiments presented for the sectors in the city of Los Angeles. The main concerns being investigated in this scenario are: whether the models are able to overcome a change in distribution in the realizations of the time series, as happens in the Standard and Permit sectors; how the models overcome sector idiosyncrasies that are characterized by their users (in other words, how different is the Permit sector model of reserved parking spots from that of the Disabled sector for people with disabilities?); and how the proximity of the sectors is presented in the models.

7.8.1 MLP

Table 7.53 shows the values of the hyperparameters obtained by random search from the MLP experiments in the city of Los Angeles. There are no common patterns in the values of the hyperparameters that can be linked to their usage or proximity.

Table 7.54 presents the RMSE values from the MLP candidate models for Los Angeles. The low error levels achieved by some of them are due to the fact that the test data included some parking places that were closed and thus the occupancy levels became low, especially in the Permit sector. Despite the low occupancy levels, the patterns across the hours of the days remain, and not all the sectors were affected by the effect of works in the zone.

Figure 7.63 shows the forecasts for six different forecasting steps from the candidate model of the Standard sector. The model outputs different occupancy levels for weekdays

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_endogenous_LA_Standard	0.083213	0.091379	0.096799	0.102012	0.105445	0.112373	0.099285
MLP_endogenous_LA_Permit	0.026132	0.027593	0.028446	0.029413	0.030845	0.032627	0.029377
MLP_endogenous_LA_ZipCar	0.065272	0.088367	0.099961	0.106587	0.111648	0.115079	0.099335
MLP_endogenous_LA_Disabled	0.068539	0.070388	0.073355	0.076586	0.078879	0.079243	0.074189

Table 7.54: Los Angeles MLP candidate model RMSE values in the test set, by hour and on average.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_endogenous_LA_Standard	128	2	0.5	0	He normal	ReLu	128	SGD	0.0285305	0.9
LSTM_endogenous_LA_Permit	512	1	0	0.25	Glorot normal	Linear	32	SGD	0.77142755	0.6
LSTM_endogenous_LA_ZipCar	512	1	0.5	0	Glorot normal	Linear	672	SGD	0.0022204800	0.8
LSTM_endogenous_LA_Disabled	128	1	0.25	0.5	He normal	Linear	672	SGD	0.0028594766	0.8

Table 7.55: Los Angeles LSTM candidate model hyperparameter values.

and weekends, although the minimum level is a bit above zero. Unlike observations from other experiments, the levels of the predictions do not shrink as the forecasting steps go further.

The first forecasting steps presented in Figure 7.64 show that the models are able to overcome the change in distribution of the test set. Furthermore, we can see that they provide adequate forecasts of the occupancy levels, which have become much lower than the levels for the training data. One can observe that, although the model responses seem adequate, their occupancy peaks are always a bit lower.

7.8.2 LSTM

LSTM hyperparameter values are presented in Table 7.55 for each of the candidate models from the Los Angeles sectors. The values of the models do not change a lot, as they have either 128 or 512 neurons and linear or ReLu activation functions (unbounded activation functions, like in Antwerp). Only the candidate model for the Standard sector has two hidden layers.

Table 7.56 shows that the LSTM candidate model for the Standard sector has a far greater error than the MLP candidate model, indicating that recurrent NN struggles with the distribution changes in the time series and are not able to adapt to them. Despite that, the scores in the other sectors are similar to those from the corresponding MLP ones. Although the Permit sector also has a distribution change, it surprisingly has a similar RMSE score as the MLP candidate model.

The visualizations of forecasts at different forecasting horizons (one to six hours) are presented in Figure 7.65. They show that, although the model provides occupancies below zero, the forecasted occupancies follow the shape of the time series with adequate levels throughout the days and weeks.

Figure 7.66 presents the visualizations for the first forecasting steps for the Los Angeles sectors. They show accurate forecasts at the first forecasting step, but it decaysas further

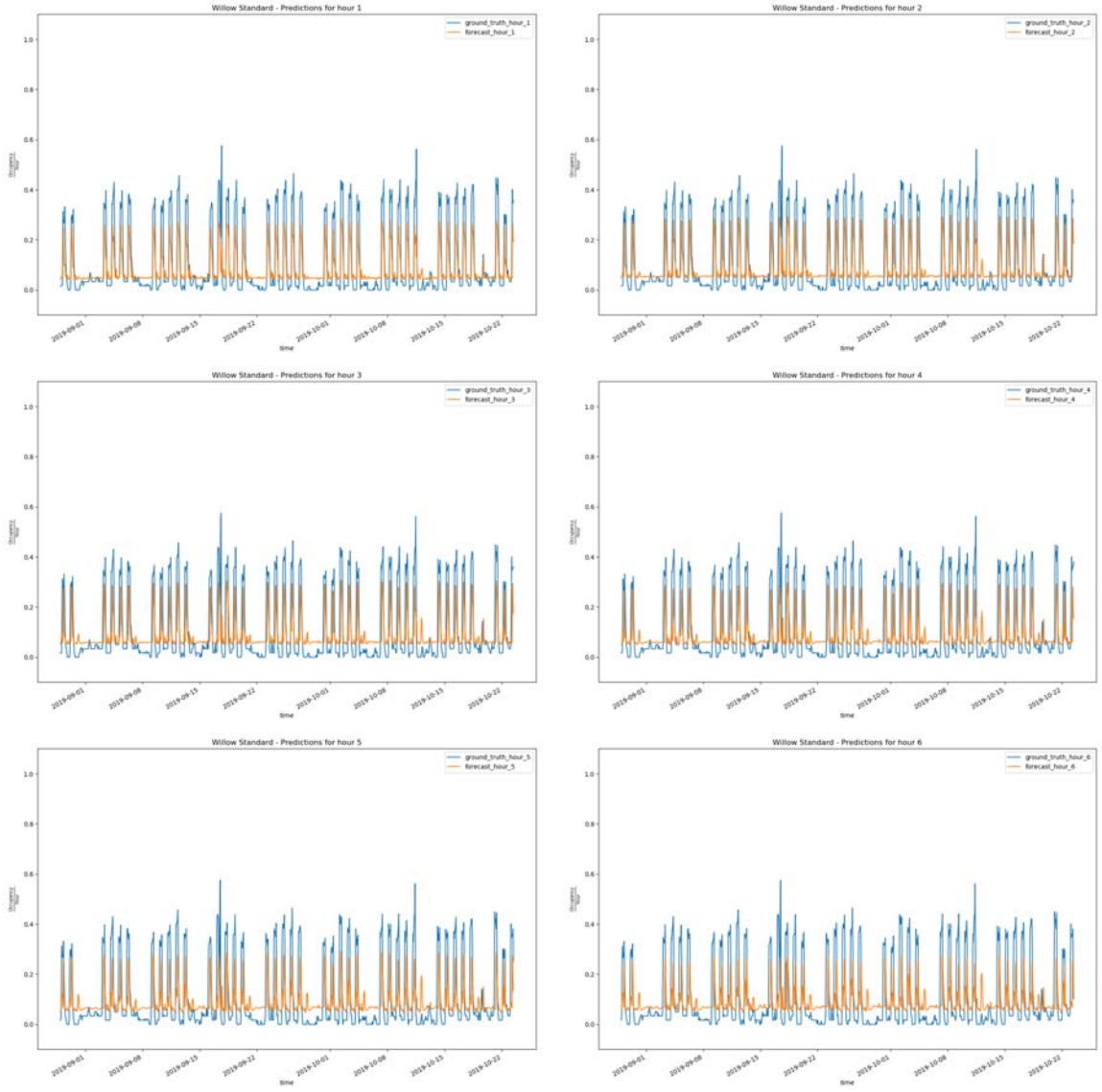


Figure 7.63: Los Angeles Standard MLP forecast visualizations from the candidate model. From top left to bottom right are steps one to six

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_endogenous_LA_Standard	0.157765	0.170322	0.187332	0.202759	0.221877	0.232236	0.198609
LSTM_endogenous_LA_Permit	0.023564	0.026408	0.028256	0.030966	0.0344788	0.029721	0.030148
LSTM_endogenous_LA_ZipCar	0.065059	0.088938	0.101202	0.108231	0.1166095	0.119523	0.101638
LSTM_endogenous_LA_Disabled	0.055212	0.079277	0.087534	0.119898	0.143644	0.192689	0.126183

Table 7.56: Los Angeles LSTM candidate model RMSE values in the test set.

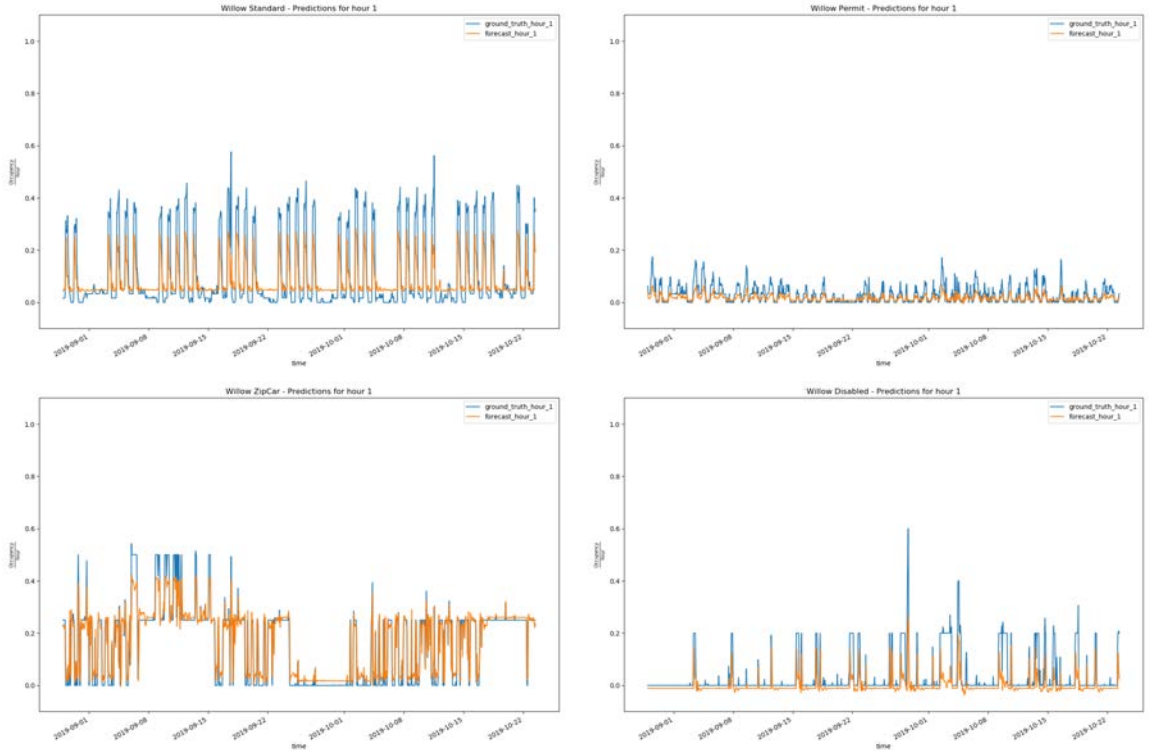


Figure 7.64: Los Angeles MLP candidate model first forecasting step plots. The first row are the Standard and Permit sectors. The sectors in the second row are ZipCar and Disabled.

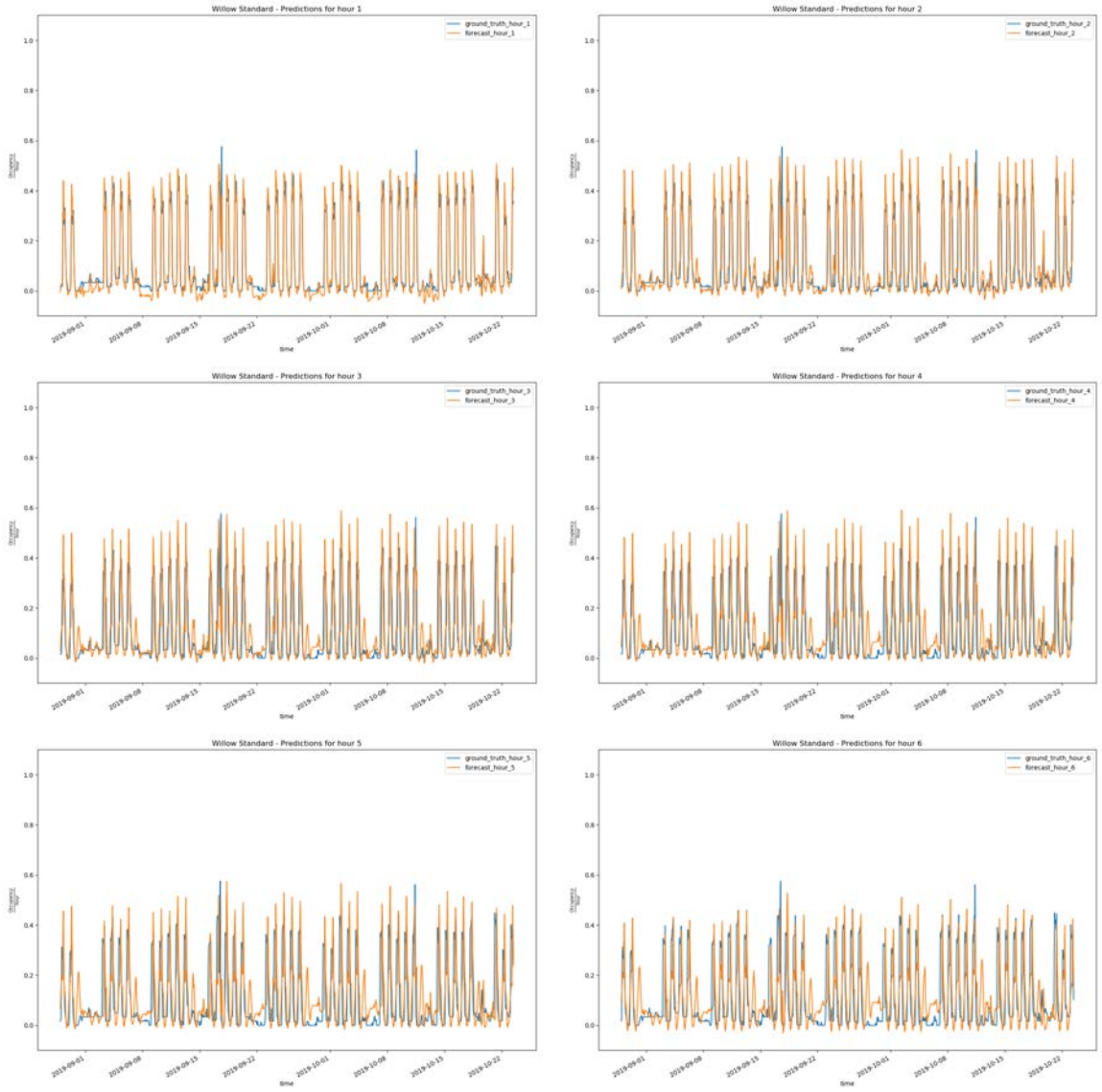


Figure 7.65: Standard sector from the Los Angeles LSTM forecast visualizations. From top left to bottom right are forecasting steps one to six.

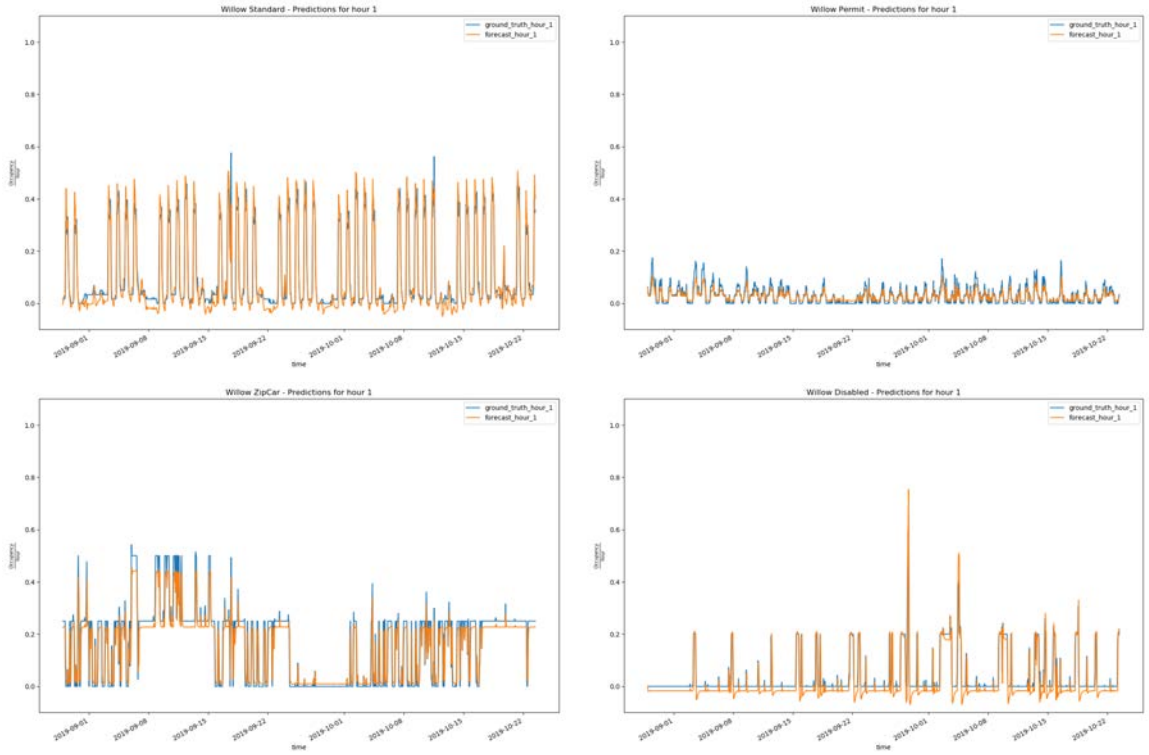


Figure 7.66: Los Angeles LSTM candidate model step one forecasts. The first row corresponds to the sectors Standard and Permit. The visualizations in the bottom row are from the ZipCar and Disabled sectors.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_endogenous_LA_Standard	64	2	0.25	0.5	Glorot normal	Linear	256	RMSProp	0.0001738	0.5
GRU_endogenous_LA_Permit	8	2	0.25	0	He normal	Linear	32	RMSProp	0.008523	0.3
GRU_endogenous_LA_ZipCar	64	2	0.5	0	Glorot normal	Sigmoid	32	RMSProp	0.00002339	0.2
GRU_endogenous_LA_Disabled	16	3	0	0.25	Glorot normal	ReLu	672	RMSProp	0.00193683	0.2

Table 7.57: Los Angeles GRU candidate model hyperparameter values.

are the step from the current time. Forecasts from ZipCar and Disabled sectors present an small offset with respect to the flat level of the series.

7.8.3 GRU

The hyperparameter values obtained for the GRU candidate models for the Los Angeles sectors are presented in 7.57. In contrast to LSTM, the candidate GRU models make use of a higher number of hidden layers (two and three). However, on the other hand, they also make use of unbounded activation functions (except the ZipCar sector).

Table 7.58 presents the scores for the GRU experiments in the city of Los Angeles. Notice that the different candidate models present very similar RMSE values across the different forecasting horizons. The reason for this behaviour is explained later.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_endogenous_LA_Standard	0.145115	0.144743	0.144704	0.143890	0.144326	0.142233	0.143721
GRU_endogenous_LA_Permit	0.032748	0.029283	0.031485	0.033495	0.034186	0.037622	0.032948
GRU_endogenous_LA_ZipCar	0.119933	0.130471	0.136913	0.140708	0.145512	0.145386	0.137795
GRU_endogenous_LA_Disabled	0.081012	0.086450	0.085712	0.092816	0.084150	0.079378	0.092937

Table 7.58: Los Angeles GRU candidate model RMSE values in the test set.

Figure 7.67 shows that the GRU model has converged towards the mean of the series and is not able to learn the time patterns in the Standard scenario, thus presenting a flat shape. Possible explanations for these results are: the regularization is too high in the model; it is not able to generalize data that is not used in training; or the model is incapable of facing a change in the distribution of data realizations.

The first task was to relax the recurrent regularization value in order to check if the model is able to learn the daily patterns. Figure 7.68 presents the realizations of the candidate model with no recurrent dropout. This time, the model has learnt the daily patterns of the sector but has signs of suffering from changes in the distribution, as it is not able to adapt to the lower values of the series and also provides negative parking occupancy values. It is interesting to note that the model forecasts low parking occupancy for the three-day weekend at the end of August. The most likely explanation for this is that the model predicts low values if the previous hours had low occupancy values (which is the case, because the previous day was Sunday). As the forecasting steps increase, the predictions at the weekends is slightly higher than the ones expected.

Another point that is worth exploring is the behaviour of the model with the data before street works happened in the sector. The visualizations in Figure 7.69 present the model trained with data mainly from 2018. Note that the model has been able to learn the patterns because they are more stable, despite the recurrent dropout value used (0.25). This shows that a high dropout value can make a model unable to learn the recurrent patterns, especially if the distribution of the data changes during the training set – something that is prone to happen in time series.Also, the practioner needs to be aware that, if the changes of distribution are not in the training set data used for the random search, a model with high error rates (as in our case) can result from using the same hyperparameters in the training and validation sets to train a model and in the test set to evaluate it.

Figure 7.70 presents the visualizations for the first forecasting steps for all sectors in Los Angeles. For the Permit sector, there was no need to tune the recurrent dropout, as the random search selected a model with no recurrent dropout and the visualization shows that the model has been able to adapt to the change in distribution. What is more, it predicts low occupancy levels that are in concordance with the available number of parking places in the sector during the street works. The candidate models obtained with random search for the sectors ZipCar and Disable offer forecast values that coincide with the mean, and this indicates that the models are subject to the same issue that occurred in the Standard sector. An individual exploration of the hyperparameters is

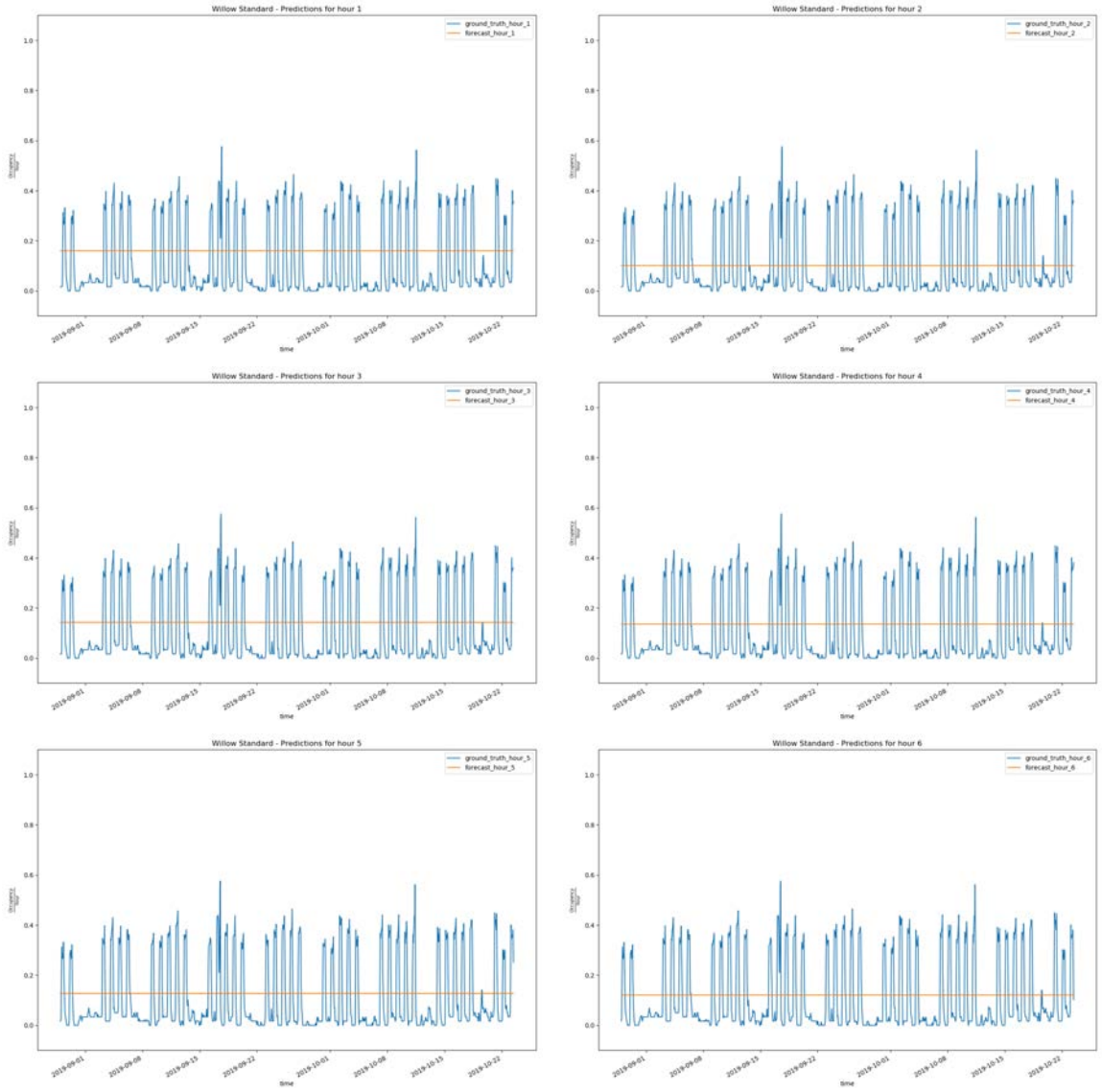


Figure 7.67: The Los Angeles Standard sector GRU forecast visualizations at different horizon steps . From top left to bottom right are forecasting steps one to six.

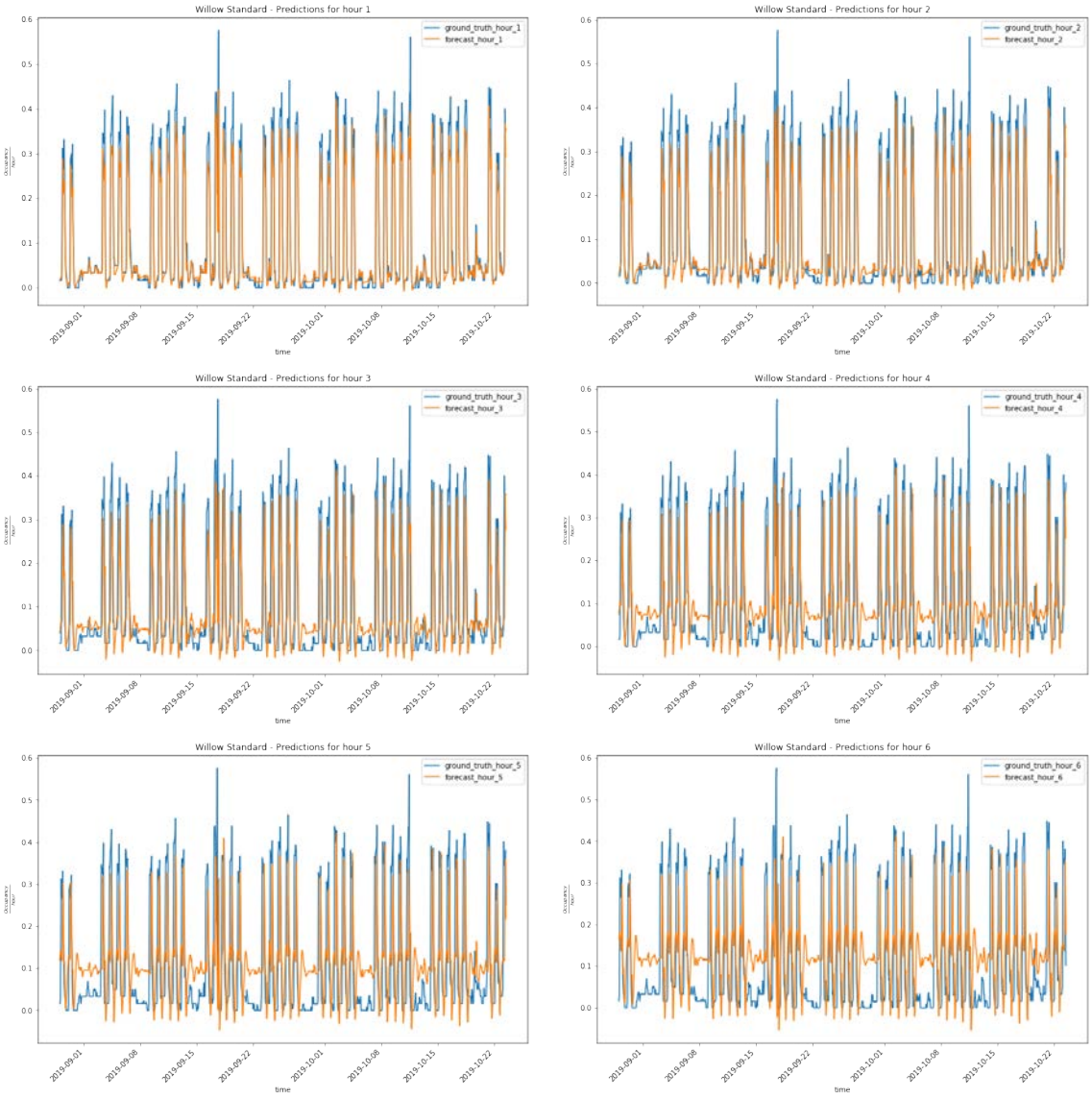


Figure 7.68: Los Angeles Standard sector GRU forecast visualizations at different horizon steps using the candidate model with a less recurrent dropout value. From top left to bottom right are forecasting steps one to six.

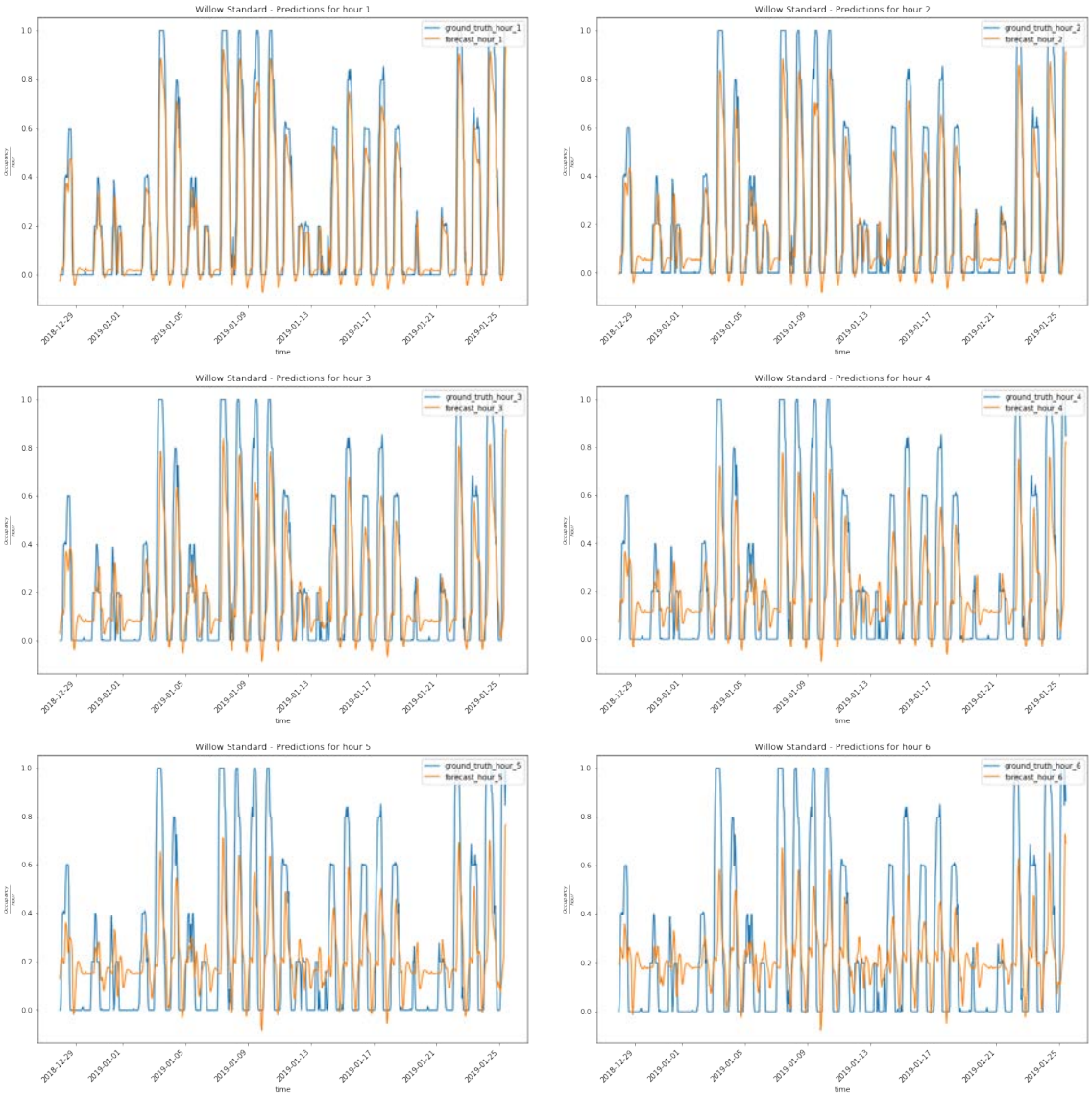


Figure 7.69: Los Angeles Standard sector GRU forecast visualizations at different horizon steps with data before the street works in the sector. From top left to bottom right are forecasting steps one to six.

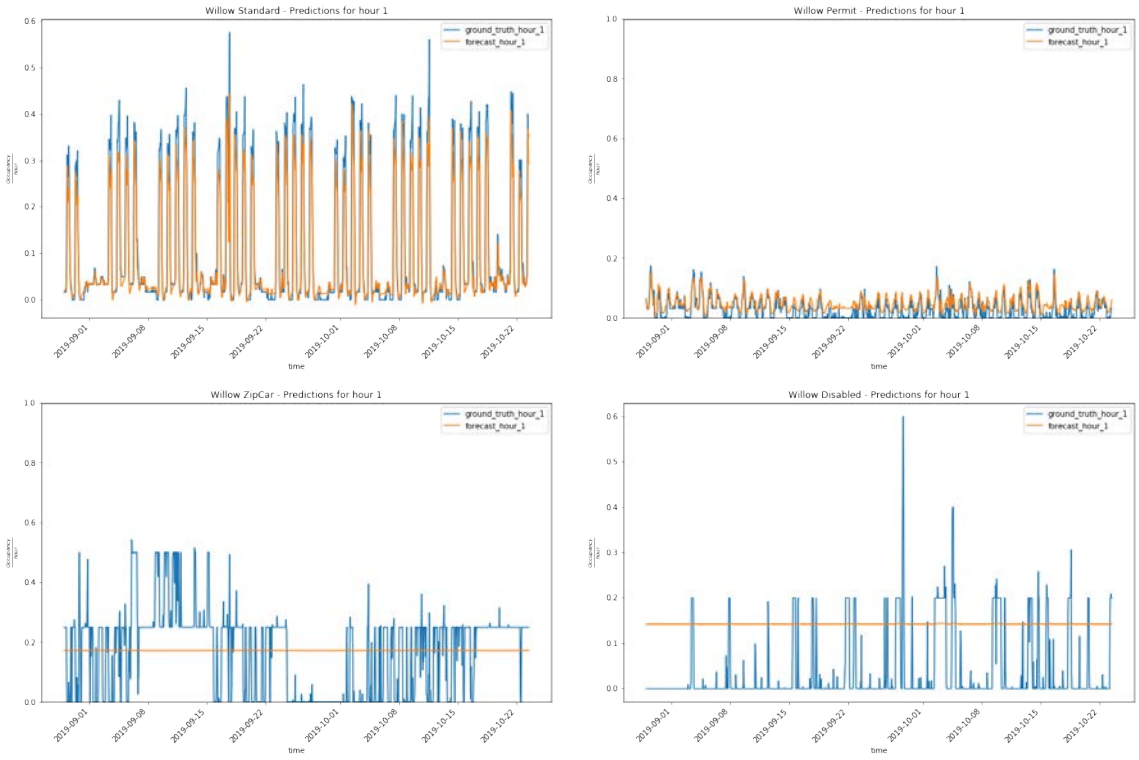


Figure 7.70: Los Angeles GRU candidate model step one forecasts. The first row corresponds to the sectors Standard and Permit. The visualizations in the bottom row are from the ZipCar and Disabled sectors.

offered next, in order to obtain better candidate models.

As the candidate model for the ZipCar sector forecasts the mean and the recurrent dropout is already 0, the next step would be to reduce the dropout to 0 and check if the model is able to learn recurrent patterns. The visualization on the left in Figure 7.71 shows the first forecasting step for the ZipCar candidate model without dropout regularization. The results continue to be the same as those from the previous version of the model (although it started to "jiggle"), so the next step was to reduce the model complexity by using one hidden layer instead of two. As can be seen in the visualization at the right, the model forecasts are more aligned with the ground truth values. This could indicate that random walk processes can be modeled using an RNN with one hidden layer and no regularization (or a high number of hidden units with dropout, like in the LSTM candidate model for ZipCar).

For the Disabled sector, the first step was to reduce the recurrent dropout. This did not improve the visualizations, so the next step was to reduce the model complexity by reducing the number of hidden layers to two. With this, the forecasts began to become more like the ground truth. The next step was to increase the number of units in the hidden layers in order to increase model capacity. This last change made the model able to provide good enough forecasts, but a close-up visualization indicates that, in fact, the

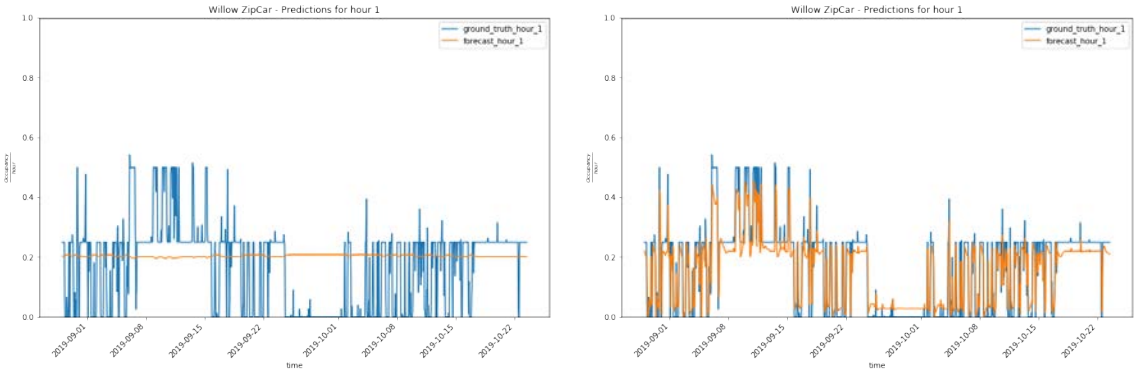


Figure 7.71: Los Angeles GRU candidate model for the ZipCar sector with no regularization. Left is the model with two hidden layers and right the model with only one.

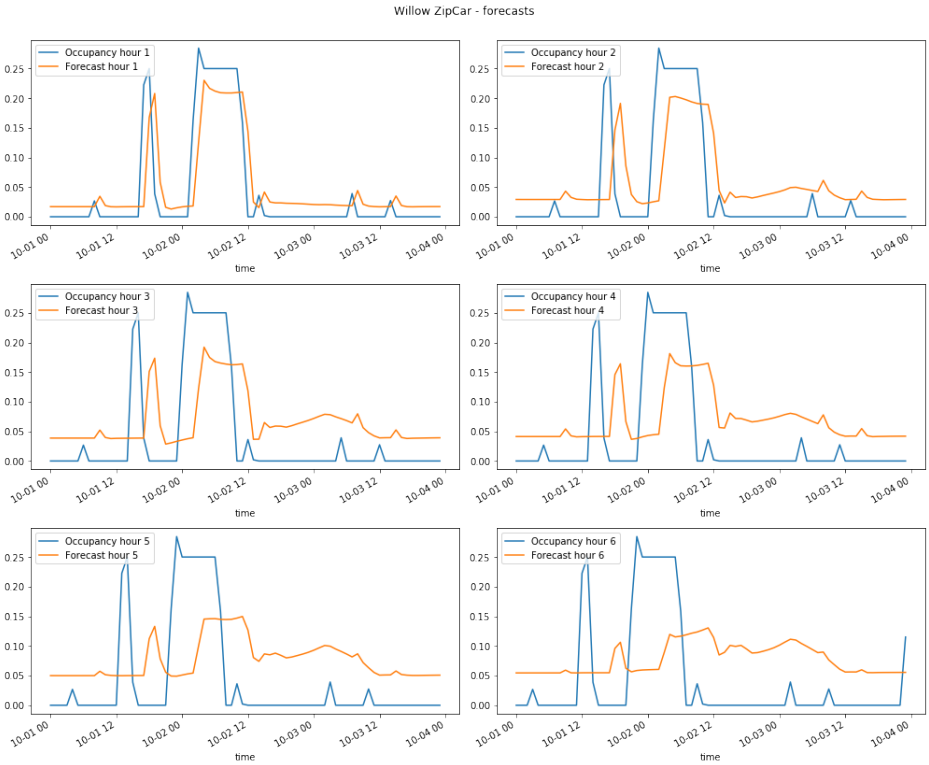


Figure 7.72: The visualizations from the outputs of Los Angeles GRU modified candidate model for the ZipCar sector.

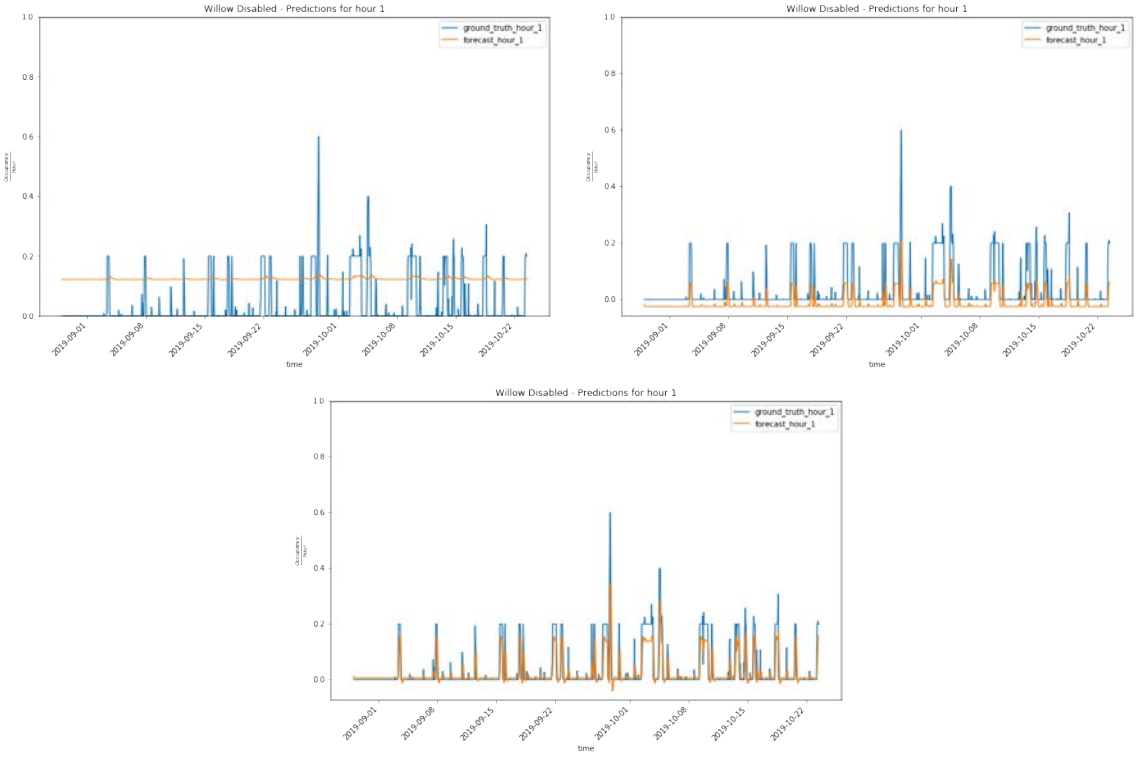


Figure 7.73: Los Angeles GRU candidate model for the Disabled sector. The first visualization corresponds to the model with no dropout, the second is the model with two hidden layers, and the third is the model with 64 units in the hidden layers.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_endogenous_LA_Standard	64	2	0.25	0.	Glorot normal	Linear	256	RMSProp	0.0001738	0.5
GRU_endogenous_LA_Permit	8	2	0.25	0	He normal	Linear	32	RMSProp	0.008523	0.3
GRU_endogenous_LA_ZipCar	64	1	0.5	0	Glorot normal	Sigmoid	32	RMSProp	0.00002339	0.2
GRU_endogenous_LA_Disabled	64	2	0	0.	Glorot normal	ReLu	672	RMSProp	0.00193683	0.2

Table 7.59: Los Angeles GRU candidate model hyperparameter values.

model is repeating the last occupancy value as the next one, as can be shown in Figure 7.73.

Finally, the model architectures and the RMSE scores following the manual changes are presented in Tables 7.59 and 7.60.

Table 7.60 presents the scores for the GRU experiments in the city of Los Angeles. Notice that the different candidate models present very similar RMSE values across the different forecasting horizons. The reason for this behaviour is explained later.

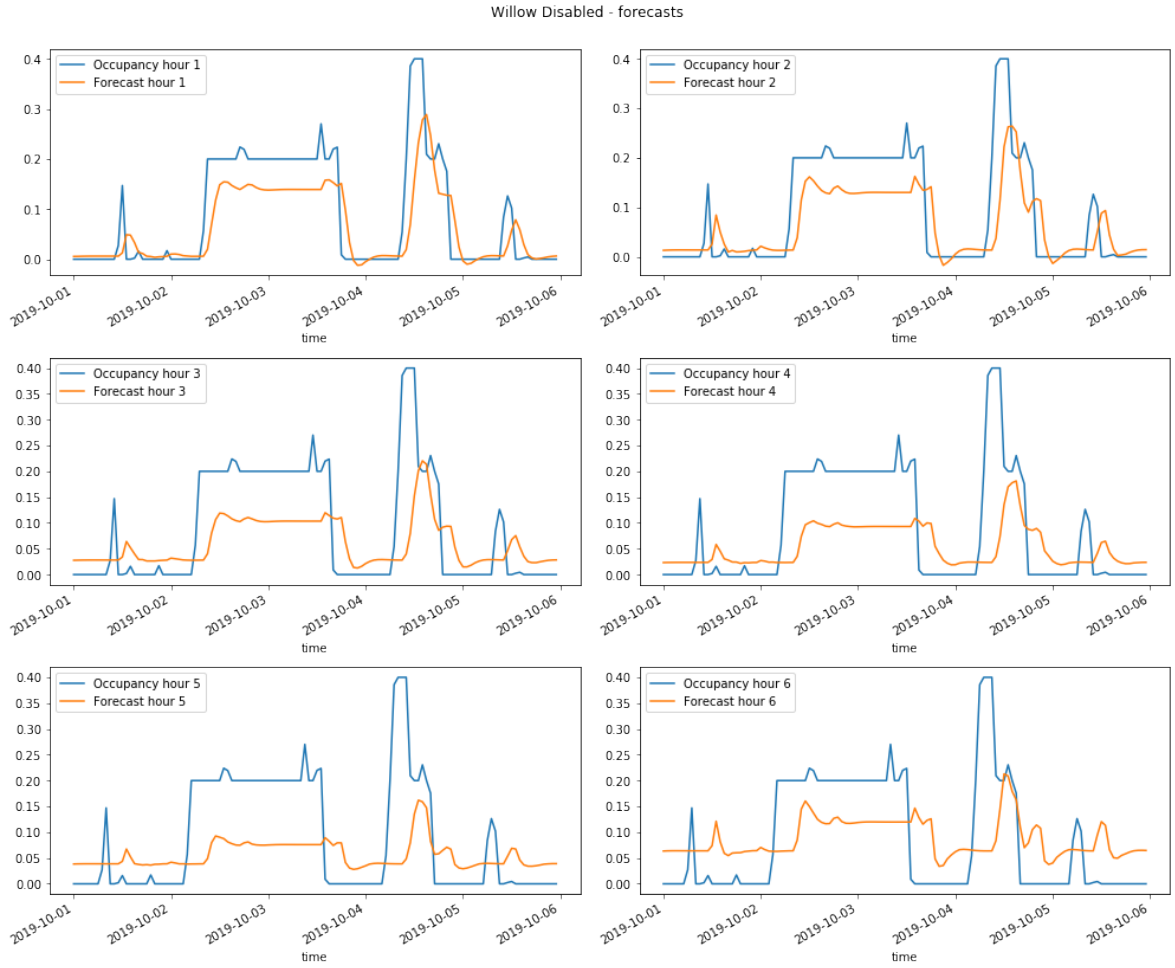


Figure 7.74: The visualizations from the outputs of the Los Angeles GRU modified candidate model for the Disabled sector.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_endogenous_LA_Standard	0.07801	0.09512	0.10619	0.11393	0.12064	0.12356	0.107401
GRU_endogenous_LA_Permit	0.032748	0.029283	0.031485	0.033495	0.034186	0.037622	0.032948
GRU_endogenous_LA_ZipCar	0.07025	0.09021	0.10057	0.10946	0.11207	0.11485	0.10076
GRU_endogenous_LA_Disabled	0.0476	0.05673	0.06614	0.06944	0.07601	0.08876	0.06872

Table 7.60: Los Angeles GRU candidate model RMSE values in the test set.

Sector	Training	Validation	Test
All sectors	2018-09-01, 2019-08-01	2019-08-02, 2019-09-12	2019-09-13, 2019-10-24

Table 7.61: Riyadh sector date intervals for the training, validation and test sets.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_endogenous_riyadh_sector1	256	4	0.25	He normal	ReLu	256	RMSProp	0.00022
MLP_endogenous_riyadh_sector10	1024	3	0	Glorot normal	ReLu	256	RMSProp	0.00017

Table 7.62: Riyadh MLP candidate model characteristics for Sector-1 and Sector-10.

7.9 NN experiments - Riyadh

Table 7.61 presents the date range for the training, validation and test sets for each of the experiments on the sectors in the city of Riyadh.Similarly to what was done in Chapter 6, only the results from Sector-1 and Sector-10 are presented.

7.9.1 MLP

The characteristics of the MLP candidate models obtained for the Riyadh sectors are presented in Table 7.62. We can observe that both models make use of the ReLu activation function and some considerable depth, four layers for Sector-1 and three layer for Sector-10.

Table 7.63 presents the RMSE scores for the MLP experiments in Riyadh. The most remarkable point is that low RMSE scores have been obtained from both models (less than 0.1 RMSE). As a particularity, Sector-1 presents that the error for successive hours increases faster than in Sector-10, although the latter has more error in the first five hours.

Figure 7.75 shows the forecasts of the candidate MLP model for Sector-1 in Riyadh. The model shows nearly perfect accuracy across the month of data used in the test set for one to six forecasting steps.

Figure 7.76 presents the visualizations of the candidate models for Sector-1 and Sector-10 in Riyadh. Predictions from the candidate model for Sector-10 capture the patterns that are present in the data, but they overestimate the occupancy at peak hours by predicting occupancies above 1.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_endogenous_riyadh_sector1	0.047095	0.060557	0.073404	0.083346	0.093408	0.103209	0.079634
MLP_endogenous_riyadh_sector10	0.071116	0.081256	0.080439	0.086139	0.097126	0.096646	0.087643

Table 7.63: Riyadh MLP candidate model RMSE values in the test set for Sector-1 and Sector-10.

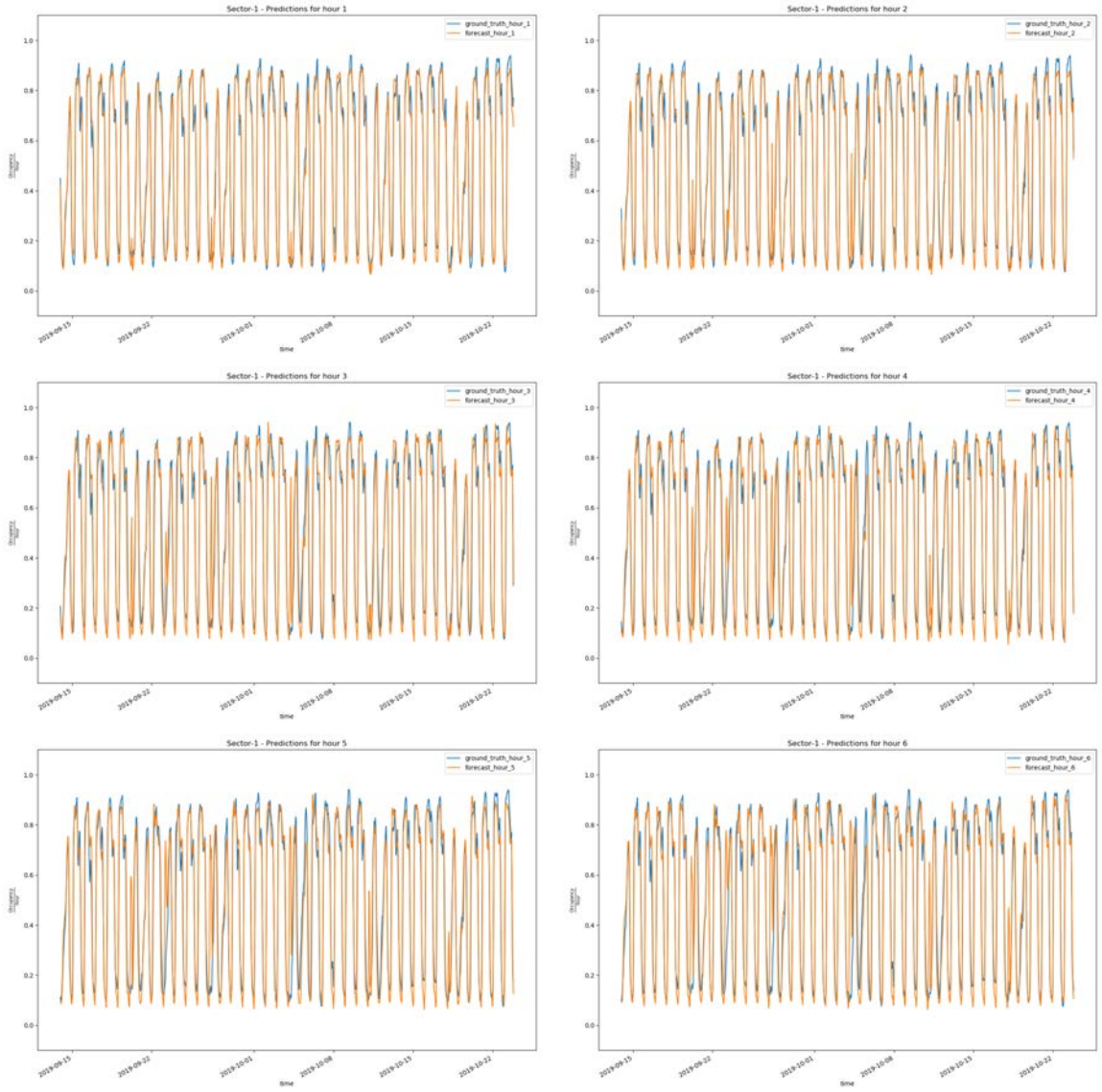


Figure 7.75: Riyadh Sector-1 MLP candidate model forecast visualizations. From top left to bottom right are steps one to six

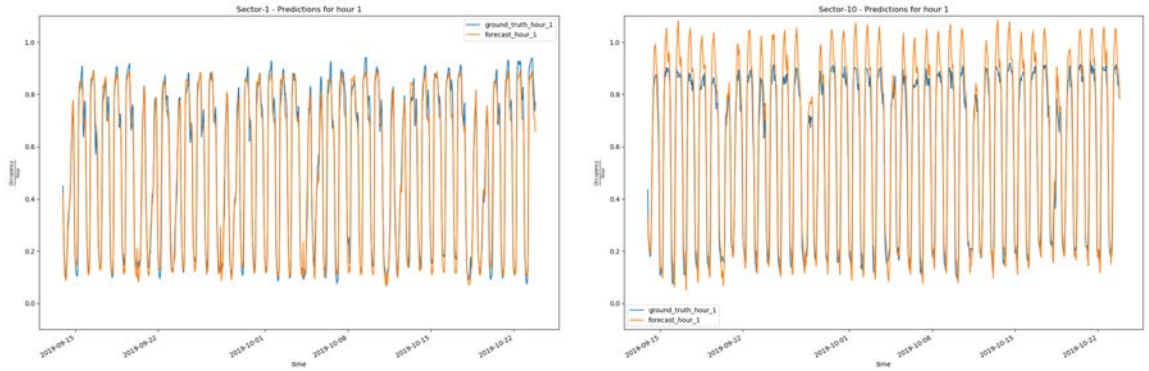


Figure 7.76: Riyadh MLP candidate model first forecasting step plots. Left is Sector-1 and right is Sector-10.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_endogenous_riyadh_sector1	512	2	0.5	0.25	He normal	ReLu	64	SGD	0.195172	0.7
LSTM_endogenous_riyadh_sector10	256	3	0.25	0.5	He normal	ReLu	128	SGD	0.966438	0.4

Table 7.64: Riyadh LSTM candidate model hyperparameter values.

7.9.2 LSTM

The hyperparameter values for the LSTM candidate models for Riyadh are presented in Table 7.64. Both models use the activation function ReLu and are not very different between each other.

Table 7.65 presents the RMSE values for the different forecasting steps (from one to six) for the considered LSTM experiments in the city of Riyadh. Both models achieve similar RMSE values, but the model for Sector-1 presents fewer errors, especially in the first forecasting steps.

The visualizations of the forecasts at different horizon steps one to six are presented in Figure 7.77 for the Sector-1. The visualizations show that the model outputs accurate forecasts for the first forecasting steps, but the predictions fall short at higher time steps.

The first forecasting steps from the Riyadh candidate LSTM models are presented in Figure 7.78. Both visualizations present accurate forecasts for a one-month span of time, although those from Sector-1 offer lower values than expected. On the other hand, those from Sector-10 offer higher values at the peaks than expected.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_endogenous_riyadh_sector1	0.051068	0.069476	0.080824	0.085531	0.093426	0.111499	0.084376
LSTM_endogenous_riyadh_sector10	0.078077	0.090971	0.098520	0.108535	0.118905	0.128008	0.104397

Table 7.65: Riyadh LSTM candidate model RMSE values in the test set.

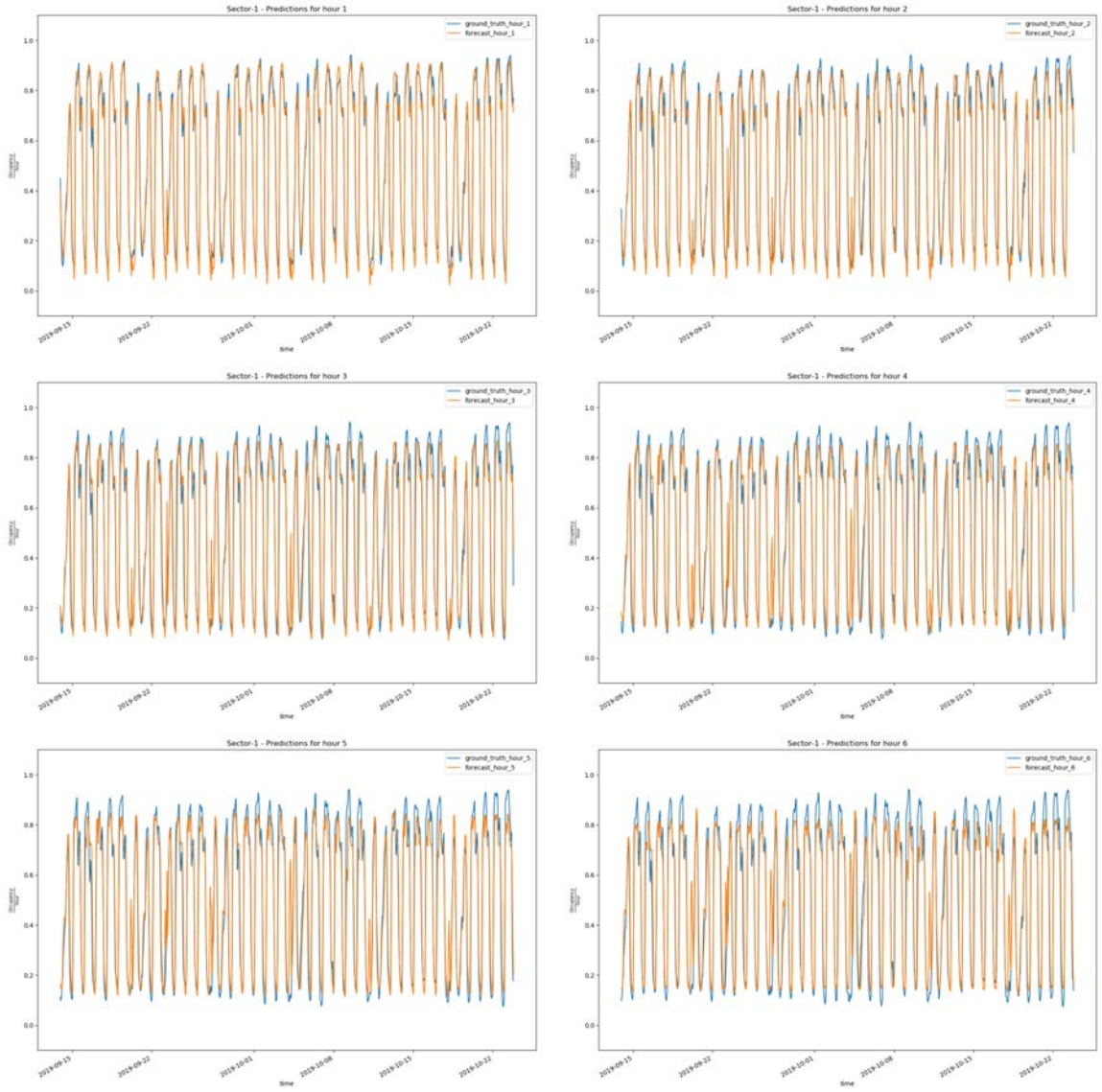


Figure 7.77: Riyadh Sector-1 LSTM forecast visualizations. From top left to bottom right are forecasting steps one to six.

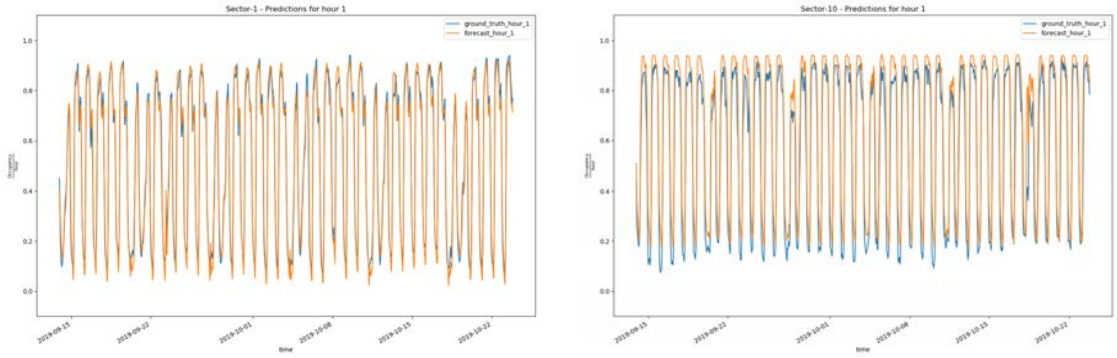


Figure 7.78: Riyadh LSTM candidate model step one forecasts. Top for Sector-1 and bottom for Sector-10.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_endogenous_riyadh_sector1	32	2	0.25	0.	He normal	Linear	256	RMSProp	0.0009239435	0.3
GRU_endogenous_riyadh_sector10	64	2	0.25	0.	He normal	Sigmoid	672	RMSProp	0.002696859	0.5

Table 7.66: Riyadh GRU candidate model hyperparameter values.

7.9.3 GRU

The candidate model hyperparameter values from the Riyadh experiments using the GRU method are presented in Table 7.66. Both models are similar in the number of hidden layers, and both also have a low number of neurons. Another aspect shared by both models is the values of the regularization technique used, as both have the value 0.25 for dropout in the hidden layer and no dropout in the hidden states.

Table 7.67 presents the scores for the GRU experiments in the city of Riyadh. Both models show a similar mean RMSE score where model for Sector-1 has lower error in the first hours while the model for Sector-10 has low error in the last hours.

Forecasts at different horizon steps are presented in Figure 7.79 for the Sector-1 candidate model. Although the model provides accurate forecasts for the six steps, it falls short in the values of the peak occupancies.

When checking the first forecasting steps of both models (Figure 7.80), the candidate model for Sector-10 offers better forecasted values for the next hour than those from Sector-1.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_endogenous_riyadh_sector1	0.059842	0.076781	0.086742	0.092680	0.096025	0.106015	0.087838
GRU_endogenous_riyadh_sector10	0.063591	0.072662	0.081042	0.087759	0.091248	0.097092	0.083619

Table 7.67: Riyadh GRU candidate model RMSE values in the test set.

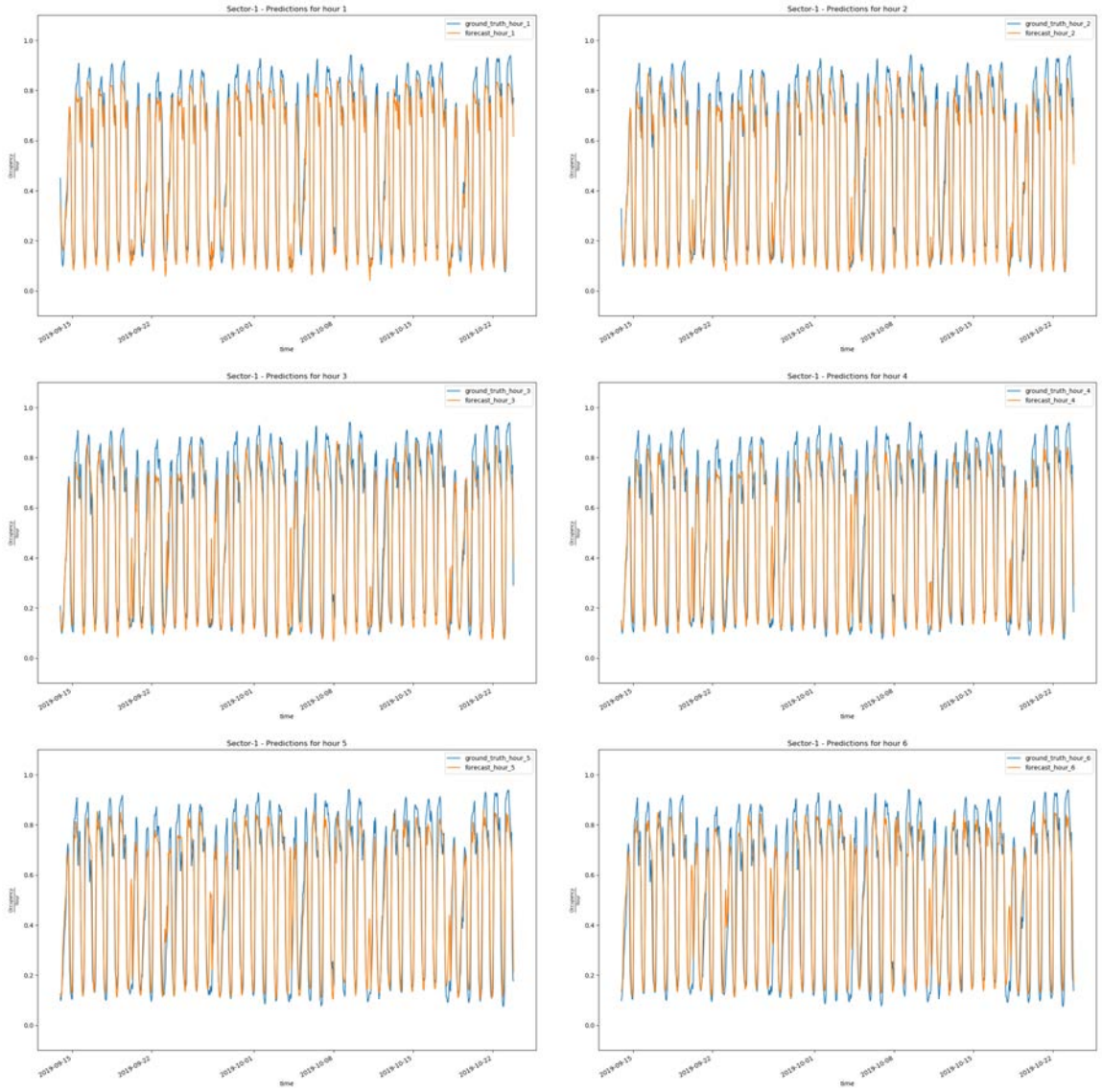


Figure 7.79: Riyadh Sector-1 GRU forecasts step one visualizations. From top left to bottom right are forecasting steps one to six.

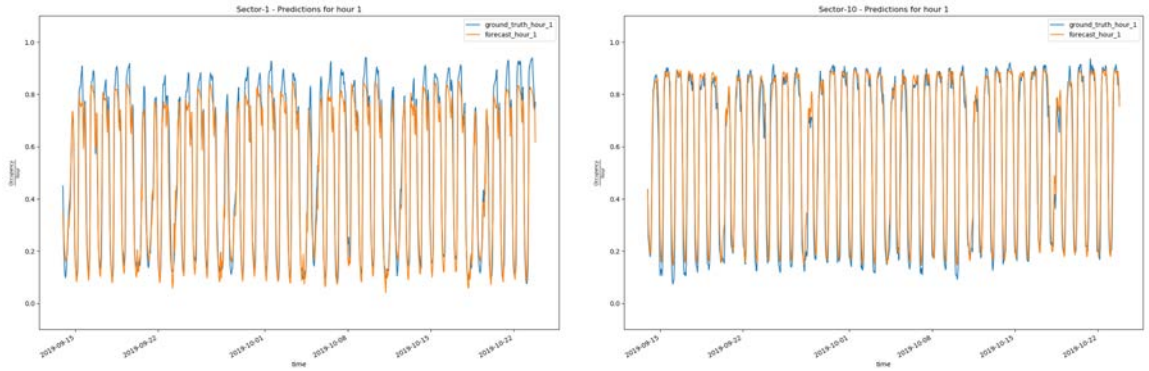


Figure 7.80: Riyadh GRU candidate model step one forecasts. Top for Sector-1 and bottom for Sector-10.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_endogenous_antwerp_pelikaanstraat	64	3	0.25	He normal	ReLu	672	RMSProp	0.00028786
MLP_endogenous_antwerp_vanwesenbekestraat	512	4	0.25	Glorot normal	Linear	64	RMSProp	0.00436647
MLP_endogenous_antwerp_vanwesenbekestraat2	128	3	0.25	Glorot normal	ReLu	128	RMSProp	0.00265265
MLP_endogenous_antwerp_vanSchoonhovestraat1	64	3	0	He normal	ReLu	256	RMSProp	0.00363
MLP_endogenous_antwerp_vanSchoonhovestraat2	64	4	0.25	Glorot normal	Linear	64	RMSProp	0.000418
MLP_endogenous_antwerp_vanSchoonhovestraat3	64	5	0.5	He normal	Linear	64	RMSProp	0.0002242
MLP_endogenous_barcelona_regular	256	4	0.5	He normal	Linear	672	RMSProp	0.00282544
MLP_endogenous_wattens_inside	128	2	0.25	He normal	ReLu	256	RMSProp	0.00386
MLP_endogenous_wattens_outside	256	5	0.5	He normal	Sigmoid	256	RMSProp	0.00400
MLP_endogenous_LA_Standard	1024	4	0.5	He normal	ReLu	128	RMSProp	0.00279
MLP_endogenous_LA_Permit	64	4	0.25	Glorot normal	Linear	672	RMSProp	0.00725562
MLP_endogenous_LA_ZipCar	64	2	0.25	Glorot normal	Sigmoid	672	RMSProp	0.00137
MLP_endogenous_LA_Disabled	256	5	0.5	Glorot normal	ReLu	256	RMSProp	0.00277038
MLP_endogenous_riyadh_sector1	256	4	0.25	He normal	ReLu	256	RMSProp	0.00022
MLP_endogenous_riyadh_sector10	1024	3	0	Glorot normal	ReLu	256	RMSProp	0.00017

Table 7.68: MLP candidate model architectures.

7.10 Results and conclusions

Throughout this chapter, different experiments have been realized in order to model parking occupancy data with NN methods. Next, the architecture and RMSE scores of the models developed from each of the experiments are presented, along with the comparison of the methods and the conclusions.

7.10.1 MLP

Table 7.68 presents the architectures of the candidate MLP models for each of the experiments considered.

A more specific quantification of the usage number for each of the MLP method's hyperparameters is given in Figure 7.81. The visualization shows clear tendencies for some of the hyperparameter values considered, and this could help practitioners reduce the search space of their random search experiments when using NN methods for parking

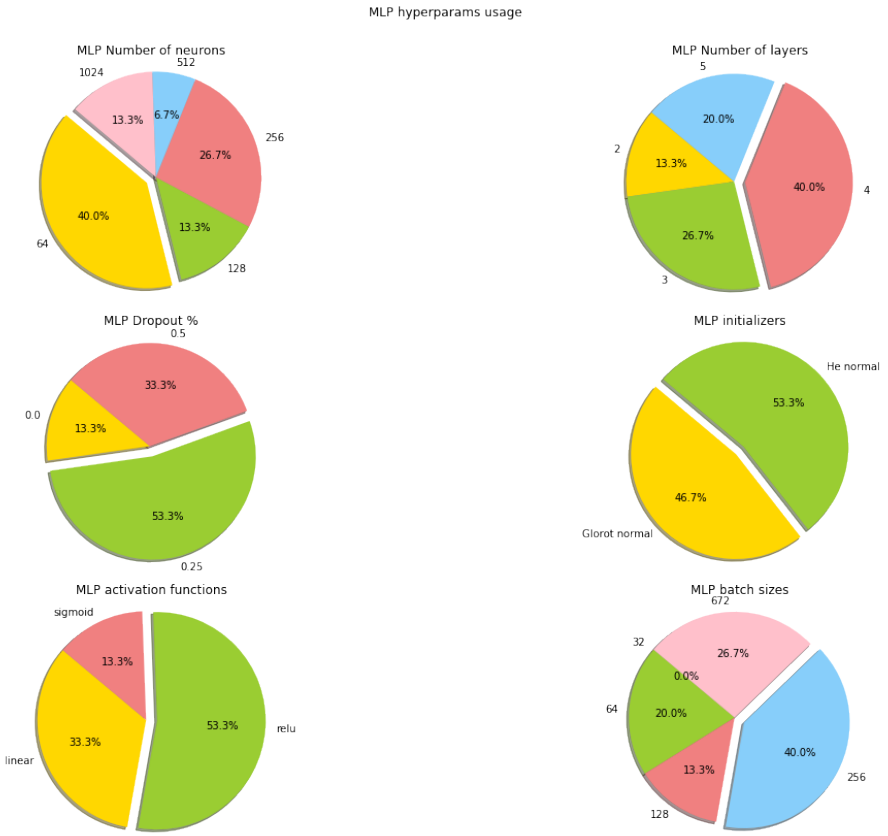


Figure 7.81: MLP hyperparameter usage across experiments.

time series forecasting modeling. Specifically, we found that, for our candidate MLP models, 64 is the preferred number of neurons per hidden layer while using four hidden layers. A dropout usage of 25% has been used for more than half of the experiments considered. There is no consensus regarding which weight initializer to use; but, because ReLU is the most used activation function, the He activation function would be the way to go. A batch size of 256 has been the most used, thus invalidating the intuition that time series could benefit the most from larger batch sizes.

Table 7.69 presents all the RMSE scores obtained with the MLP candidate models from all the experiments.

Figure 7.82 presents a matrix representation of the table of RMSE scores, which allows identifying the behaviour of the candidate models in the considered experiments. The results obtained are similar to all the candidate models, but it is clear that those models trained using data with smooth occupancy changes (the sectors with more parking places) achieved fewer errors. Considering each scenario in detail, it can be observed that:

- The Antwerp experiments are the ones with higher error rates, which may be caused by their noisy patterns (mid- to small-sized sectors). The sector with the highest error is Van Schoonhovestraat 2, as it has proved to be a random walk

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_endogenous_antwerp_pelikaanstraat	0.110130	0.133542	0.150384	0.162346	0.172676	0.179238	0.153562
MLP_endogenous_antwerp_vanwesenbekestraat	0.095566	0.116724	0.125885	0.131969	0.133297	0.133658	0.124019
MLP_endogenous_antwerp_vanwesenbekestraat2	0.101069	0.124634	0.135584	0.143275	0.146959	0.148462	0.134506
MLP_endogenous_antwerp_vanSchoonhovestraat1	0.123471	0.139036	0.148395	0.154933	0.158351	0.161390	0.149402
MLP_endogenous_antwerp_vanSchoonhovestraat2	0.157954	0.204540	0.235772	0.255512	0.268585	0.280565	0.237706
MLP_endogenous_antwerp_vanSchoonhovestraat3	0.13	0.156846	0.166445	0.173264	0.176913	0.177004	0.164277
MLP_endogenous_barcelona_regular	0.092727	0.109607	0.119814	0.125462	0.130421	0.135289	0.119841
MLP_endogenous_wattens_inside	0.056937	0.081595	0.101219	0.116856	0.127939	0.138484	0.107350
MLP_endogenous_wattens_outside	0.167275	0.168808	0.160449	0.161442	0.173651	0.192762	0.173874
MLP_endogenous_LA_Standard	0.083213	0.091379	0.096799	0.102012	0.105445	0.112373	0.099285
MLP_endogenous_LA_Permit	0.026132	0.027593	0.028446	0.029413	0.030845	0.032627	0.029377
MLP_endogenous_LA_ZipCar	0.065272	0.088367	0.099961	0.106587	0.111648	0.115079	0.099335
MLP_endogenous_LA_Disabled	0.068539	0.070388	0.073355	0.076586	0.078879	0.079243	0.074189
MLP_endogenous_riyadh_sector1	0.047095	0.060557	0.073404	0.083346	0.093408	0.103209	0.079634
MLP_endogenous_riyadh_sector10	0.071116	0.081256	0.080439	0.086139	0.097126	0.096646	0.087643

Table 7.69: MLP candidate models RMSE on test set.

process. Van Schoonhobestraat 1 and 3 scored similarly, because both sectors are very similar in nature due to their size and proximity. The sector Van Wesenbekestraat 1 shows little increase in the RMSE, as the forecasting steps increase in comparison to the other candidate models of Antwerp.

- Barcelona RMSE scores are as expected, remaining within the regular patterns throughout the weekdays and weekends. The model has demonstrated adaptation to the low occupancy that occurs on the weekends.
- For Wattens, the scores from the Outside sector are surprisingbecause they are the second-highest, despite its regular patterns and similarity to the Inside sector.
- All sectors in Los Angeles scored low error values. This shows that the model trained with data before and during works is able to provide forecasts with data after street works.
- Both Riyadh sectors scored similarly, although the Sector-1 error increases more with the time steps when compared to Sector-10. The regular patterns in the data make it easy for the MLP models to acquire an RMSE value below 0.10, which can be interpreted as missing in the occupancy status when there are fewer than ten parking places in sectors with over one-hundred parking spots.

7.10.2 LSTM

Table 7.70 presents the values of all the architectures from the LSTM candidate models obtained from the different experiments.

From Figure 7.83, it can be observed that the LSTM models do not have a clear winner in terms of the number of units per layer, although it is not too high. For the number of hidden layers, two is preferred. For regularization, we find that low dropout values

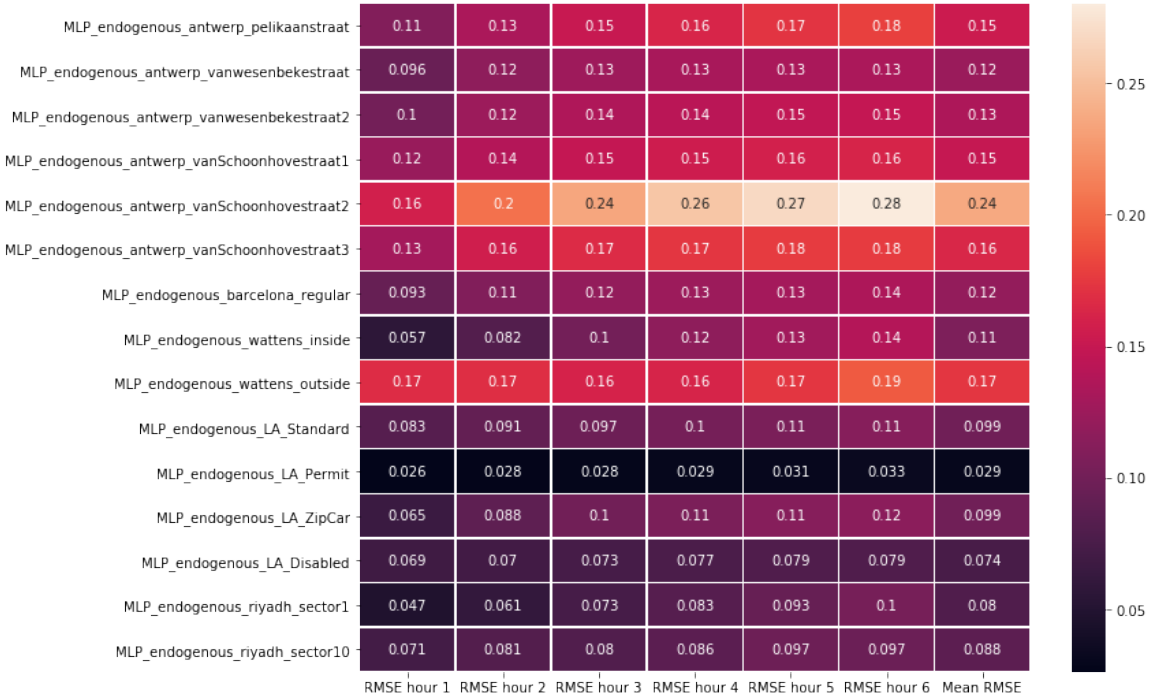


Figure 7.82: MLP RMSE heatmap.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_endogenous_antwerp_pelikaanstraat	8	1	0.25	0.25	Glorot normal	Sigmoid	64	SGD	0.02151	0.8
LSTM_endogenous_antwerp_vanwesenbekestraat	64	3	0.25	0.25	He normal	Linear	64	SGD	0.55276	0.4
LSTM_endogenous_antwerp_vanwesenbekestraat2	128	2	0.25	0	He normal	Linear	128	SGD	0.01078	0.6
LSTM_endogenous_antwerp_vanSchoonhovestraat1	8	3	0	0.25	He normal	Linear	128	SGD	0.0299967	0.6
LSTM_endogenous_antwerp_vanSchoonhovestraat2	256	3	0.5	0	Glorot normal	Linear	672	SGD	001710	0.9
LSTM_endogenous_antwerp_vanSchoonhovestraat3	32	3	0.25	0	He normal	Linear	672	SGD	0.0279009	0.6
LSTM_endogenous_barcelona_regular	16	2	0	0.5	He normal	ReLu	256	SGD	0.0542547	0.4
LSTM_endogenous_wattens_inside	16	2	0.25	0	He normal	Linear	256	SGD	0.00483981	0.8
LSTM_endogenous_wattens_outside	32	2	0	0	He normal	Sigmoid	64	SGD	0.39419	0.2
LSTM_endogenous_LA_Standard	128	2	0.5	0	He normal	ReLu	128	SGD	0.0285305	0.9
LSTM_endogenous_LA_Permit	512	1	0	0.25	Glorot normal	Linear	32	SGD	0.77142755	0.6
LSTM_endogenous_LA_ZipCar	512	1	0.5	0	Glorot normal	Linear	672	SGD	0.0022204800	0.8
LSTM_endogenous_LA_Disabled	128	1	0.25	0.5	He normal	Linear	672	SGD	0.0028594766	0.8
LSTM_endogenous_riyadh_sector1	512	2	0.5	0.25	He normal	ReLu	64	SGD	0.195172	0.7
LSTM_endogenous_riyadh_sector10	256	3	0.25	0.5	He normal	ReLu	128	SGD	0.966438	0.4

Table 7.70: LSTM candidate models architectures.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_endogenous_antwerp_pelikaanstraat	0.110668	0.137267	0.153259	0.165521	0.174594	0.181250	0.155625
LSTM_endogenous_antwerp_vanwesenbekestraat	0.103780	0.126287	0.137642	0.141567	0.142330	0.144293	0.133610
LSTM_endogenous_antwerp_vanwesenbekestraat2	0.093660	0.121632	0.137392	0.148286	0.154991	0.159997	0.137875
LSTM_endogenous_antwerp_vanSchoonhovestraat1	0.113743	0.132188	0.144037	0.145389	0.153726	0.155251	0.141533
LSTM_endogenous_antwerp_vanSchoonhovestraat2	0.177184	0.214003	0.231087	0.237052	0.254416	0.258750	0.233800
LSTM_endogenous_antwerp_vanSchoonhovestraat3	0.138450	0.157465	0.166710	0.171605	0.175181	0.175716	0.164694
LSTM_endogenous_barcelona_regular	0.100250	0.113873	0.128788	0.146382	0.164747	0.175187	0.141719
LSTM_endogenous_wattens_inside	0.063618	0.098741	0.126547	0.149175	0.162757	0.176160	0.134990
LSTM_endogenous_wattens_outside	0.095564	0.132638	0.162885	0.186857	0.191105	0.220024	0.169323
LSTM_endogenous_LA_Standard	0.157765	0.170322	0.187332	0.202759	0.221877	0.232236	0.198609
LSTM_endogenous_LA_Permit	0.023564	0.026408	0.028256	0.030966	0.0344788	0.029721	0.030148
LSTM_endogenous_LA_ZipCar	0.065059	0.088938	0.101202	0.108231	0.1166095	0.119523	0.101638
LSTM_endogenous_LA_Disabled	0.055212	0.079277	0.087534	0.119898	0.143644	0.192689	0.126183
LSTM_endogenous_riyadh_sector1	0.051068	0.069476	0.080824	0.085531	0.093426	0.111499	0.084376
LSTM_endogenous_riyadh_sector10	0.078077	0.090971	0.098520	0.108535	0.118905	0.128008	0.104397

Table 7.71: LSTM candidate models RMSE on test set.

are preferred for both the units and the hidden state (recurrent dropout), although no dropout is used. Remember that layer normalization is applied to all the candidate models after the RNN layers, and this introduces regularization into the model. For initializers, He normal is the clear winner, which seems paradoxical because ReLU is the least used activation function. The most used activation function is linear, and this indicates either that parking occupancy can be modeled with linear models or that the dying ReLU problem has a more severe effect on the models considered. This would be something to explore further in the future. For the batch sizes, there is no a clear winner among the different sizes considered. Lastly, from the bar chart figure (represented in this way because of the number of factors being considered), shows that the models benefited the most from clip norm values of 0.6 and 0.8.

Table 7.71 shows the RMSE scores from all the experiments using the LSTM method.

A better interpretation can be extracted by visualizing the data in a matrix heatmap, as presented in Figure 7.84. Overall, the scores follow similar tendencies as in the MLP experiments. More specifically, for each scenario considered, it can be observed that:

- The Antwerp scenario presents RMSE values of around 0.15, with the exception of the sector Van Schoonhovestraat 2, which scored a larger error rate. Despite the similarities between the Van Schoonhovestraat sector 1 and 3, the latter has larger error rate than the former. Overall, the values are similar to those from the MLP experiments.
- The scores for the candidate model of the Barcelona scenario are similar to those of the Pelikaanstraat candidate model, although the former's patterns are more regular.
- The RMSE scores for the Wattens candidate models have a large increase in the error rate as the forecasting steps increase. Although the patterns in the scenario are regular from Monday to Thursday, the models cannot handle the forecasting

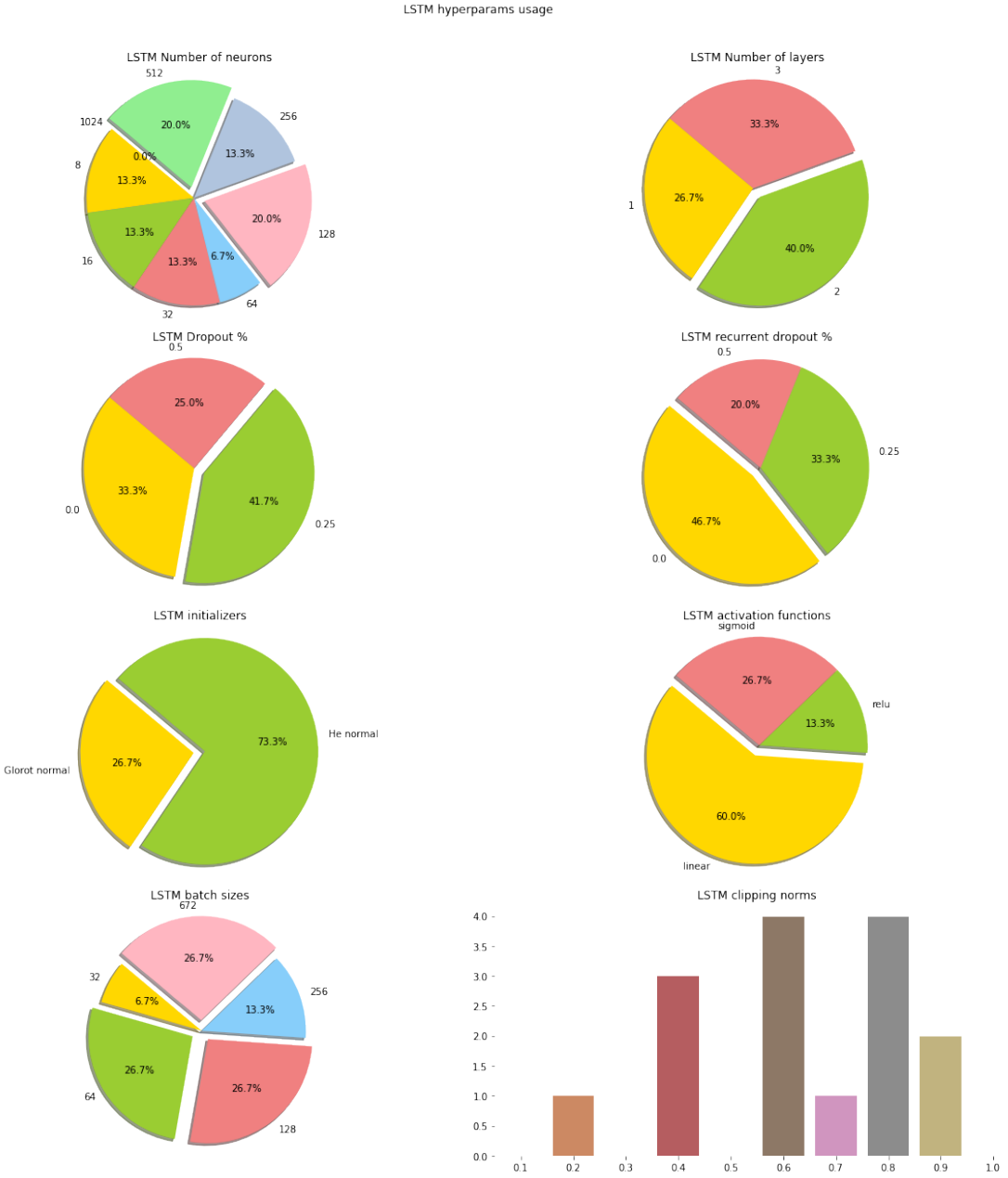


Figure 7.83: LSTM hyperparameter usage across experiments.

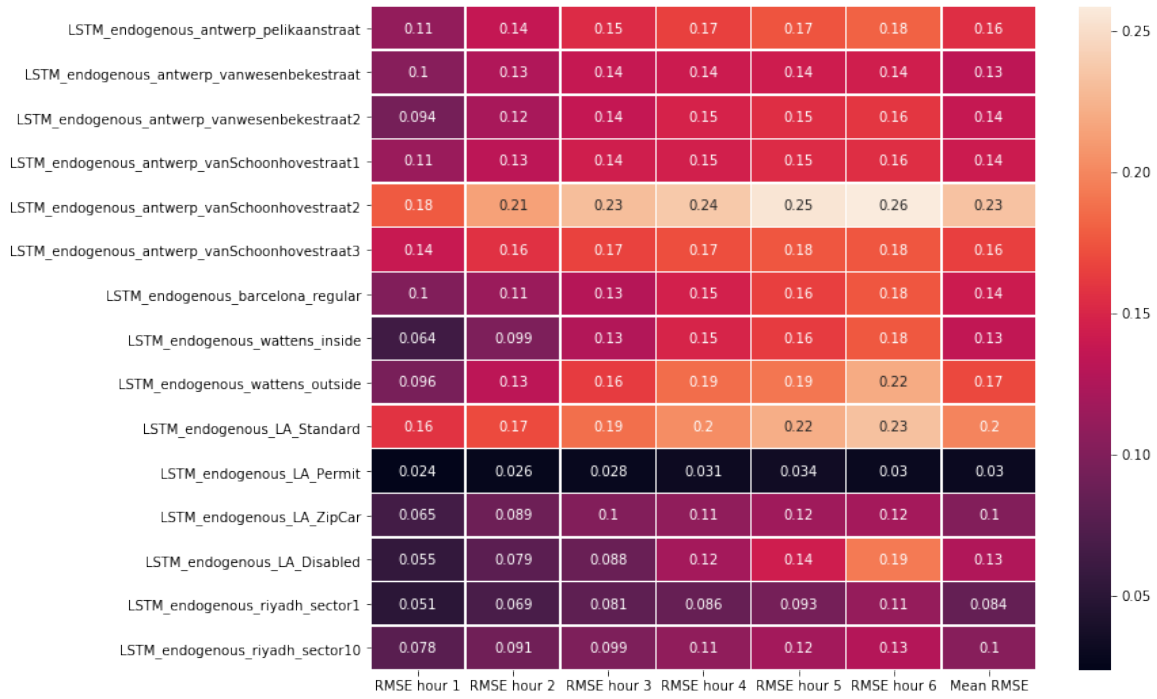


Figure 7.84: LSTM RMSE heatmap.

well beyond four hours.

- In the Los Angeles scenario, it can be observed that the second-highest RMSE comes from the Standard sector. This could be interpreted as the model being unable to adapt to the distribution changes due to the zone's street works during the considered span of time. On the other hand, the model for the Permit sector shows adaptation to the changes in distribution, which could seem counter-intuitive when compared to the Standard sector. However, the explanation for this is that the test data for the Standard sector includes occupancy levels similar to those prior to the street works, meaning that, after summer 2019, the street works ended for that sector and the number of available places was restored; whereas, for the Permit sector, the data levels were still the same as those of the street works. The Disabled sector shows a huge increase in the error rate for the last forecasting step (hour six), but a lower RMSE overall.
- The RMSE scores from the Riyadh sectors are similar to those of the MLP models.

7.10.3 GRU

All GRU candidate model hyperparameter values are presented in Table 7.72.

The visualization in Figure 7.85 presents the most used hyperparameter values from the GRU candidate models. In contrast to LSTM, the GRU models have a tendency to use 64 and 32 units in the hidden layers. Notice that no model uses more than 256. For

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_endogenous_antwerp_pelikaanstraat	32	1	0.25	0	He normal	Sigmoid	672	RMSProp	0.00001	0.2
GRU_endogenous_antwerp_vanwesenbekestraat	16	2	0.25	0	He normal	Linear	256	RMSProp	0.0001271	0.9
GRU_endogenous_antwerp_vanwesenbekestraat2	256	3	0.25	0	He normal	Sigmoid	672	RMSProp	0.00100	0.2
GRU_endogenous_antwerp_vanschoonhovestraat1	128	3	0.25	0	He normal	ReLu	128	RMSProp	0.00028981	0.5
GRU_endogenous_antwerp_vanschoonhovestraat2	32	1	0.25	0	Glorot normal	Sigmoid	128	RMSProp	0.0000609171	0.4
GRU_endogenous_antwerp_vanschoonhovestraat3	8	1	0	0.25	Glorot normal	Linear	32	RMSProp	0.00176222	0.7
GRU_endogenous_barcelona_regular	256	2	0.25	0	Glorot normal	Sigmoid	64	RMSProp	0.00001291478	0.9
GRU_endogenous_wattens_inside	32	2	0.25	0	He normal	Sigmoid	672	RMSProp	0.000373002	0.6
GRU_endogenous_wattens_outside	64	2	0.25	0	He normal	Sigmoid	128	RMSProp	0.00370607	0.8
GRU_endogenous_LA_Standard	64	2	0.25	0.	Glorot normal	Linear	256	RMSProp	0.0001738	0.5
GRU_endogenous_LA_Permit	8	2	0.25	0	He normal	Linear	32	RMSProp	0.008523	0.3
GRU_endogenous_LA_ZipCar	64	1	0.5	0	Glorot normal	Sigmoid	32	RMSProp	0.00002339	0.2
GRU_endogenous_LA_Disabled	64	2	0	0.	Glorot normal	ReLu	672	RMSProp	0.00193683	0.2
GRU_endogenous_riyadh_sector1	32	2	0.25	0.	He normal	Linear	256	RMSProp	0.0009239435	0.3
GRU_endogenous_riyadh_sector10	64	2	0.25	0.	He normal	Sigmoid	672	RMSProp	0.002696859	0.5

Table 7.72: GRU candidate models architectures.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_endogenous_antwerp_pelikaanstraat	0.113561	0.135125	0.151318	0.165038	0.173547	0.182441	0.157462
GRU_endogenous_antwerp_vanwesenbekestraat	0.103207	0.124504	0.131194	0.137746	0.142796	0.132778	0.129447
GRU_endogenous_antwerp_vanwesenbekestraat2	0.099607	0.120591	0.128654	0.138555	0.139059	0.141712	0.129501
GRU_endogenous_antwerp_vanschoonhovestraat1	0.119425	0.134792	0.141594	0.144979	0.146056	0.148689	0.139435
GRU_endogenous_antwerp_vanschoonhovestraat2	0.163100	0.201986	0.229196	0.245610	0.258182	0.265528	0.229798
GRU_endogenous_antwerp_vanschoonhovestraat3	0.128000	0.15256	0.162955	0.167865	0.169857	0.171224	0.15943
GRU_endogenous_barcelona_regular	0.096280	0.114129	0.130481	0.142458	0.148959	0.158079	0.133832
GRU_endogenous_wattens_inside	0.056622	0.079935	0.100578	0.116191	0.127633	0.139902	0.107230
GRU_endogenous_wattens_outside	0.083591	0.116279	0.139536	0.156237	0.165543	0.174135	0.142176
GRU_endogenous_LA_Standard	0.07801	0.09512	0.10619	0.11393	0.12064	0.12356	0.107401
GRU_endogenous_LA_Permit	0.032748	0.029283	0.031485	0.033495	0.034186	0.037622	0.032948
GRU_endogenous_LA_ZipCar	0.07025	0.09021	0.10057	0.10946	0.11207	0.11485	0.10076
GRU_endogenous_LA_Disabled	0.0476	0.05673	0.06614	0.06944	0.07601	0.08876	0.06872
GRU_endogenous_riyadh_sector1	0.059842	0.076781	0.086742	0.092680	0.096025	0.106015	0.087838
GRU_endogenous_riyadh_sector10	0.063591	0.072662	0.081042	0.087759	0.091248	0.097092	0.083619

Table 7.73: GRU candidate models RMSE on test set.

the number of hidden layers considered, two is the most represented one among 60% of the models. Regarding regularization, the most chosen option is 25% dropout and 0% dropout for the hidden state, which concurs with the LSTM results. The initializers are also like those in LSTM, namely in that the most used one is He normal while the least used activation function is ReLU. Thus, the comment on LSTM also applies here. The sigmoid activation function is the one used by most of the models. There is no clear consensus about the batch sizes, but a larger batch size may be preferred. Lastly, for the clipping norm value, the values are more or less spread among the different values considered.

All RMSE scores obtained in the different experiments with the GRU candidate models are presented in Table 7.73

A heatmap of the RMSE values obtained from the GRU candidate models is presented in Figure 7.86. Overall, it can be seen that the RMSE values at each forecasting step increase less than those from the other methods. The experiments with higher error rates are the ones from the Van Schoonhovestraat 2 and Pelikaanstraat sectors. For each scenario, it can be observed for the GRU models that:

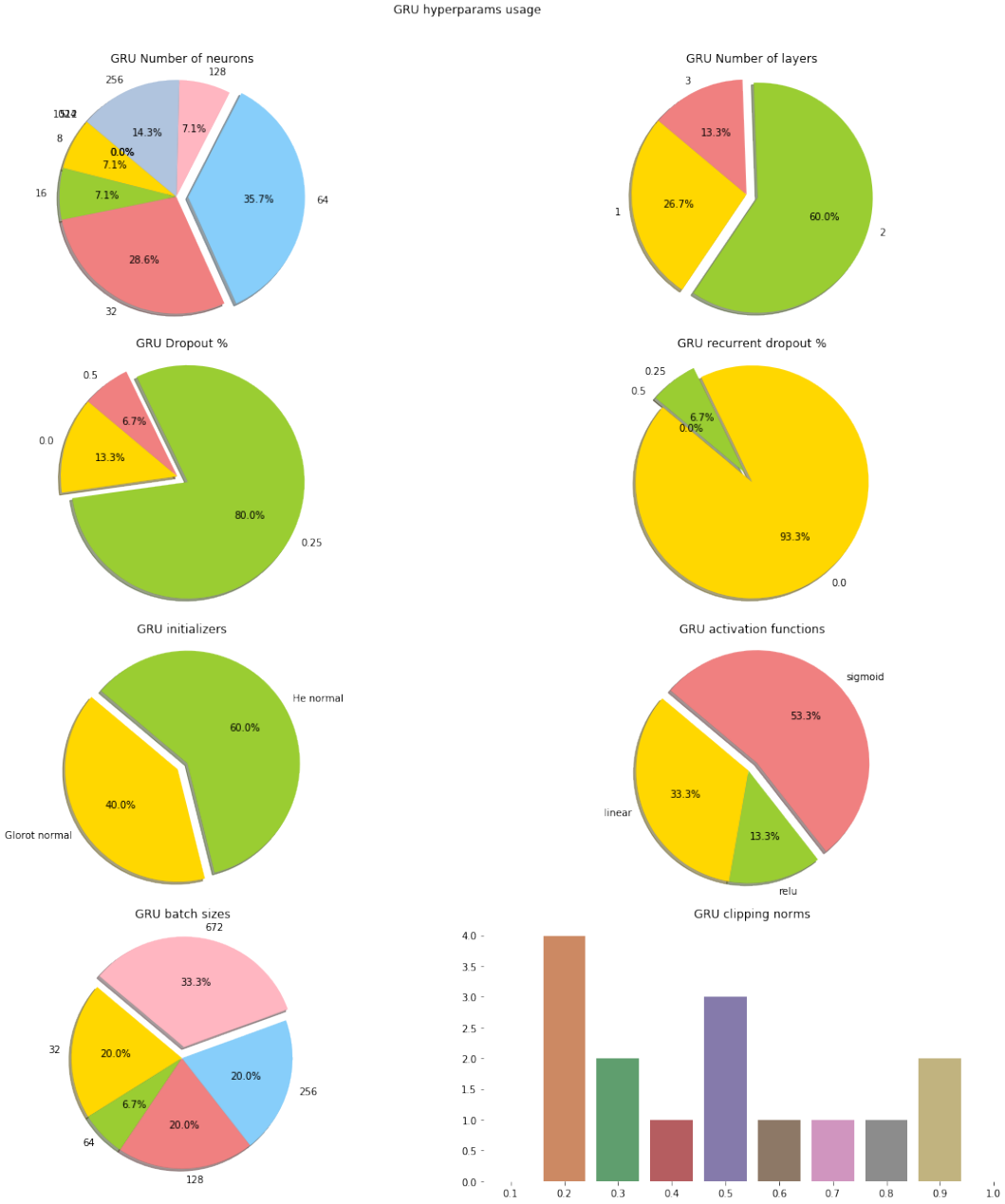


Figure 7.85: GRU hyperparameter usage across experiments.

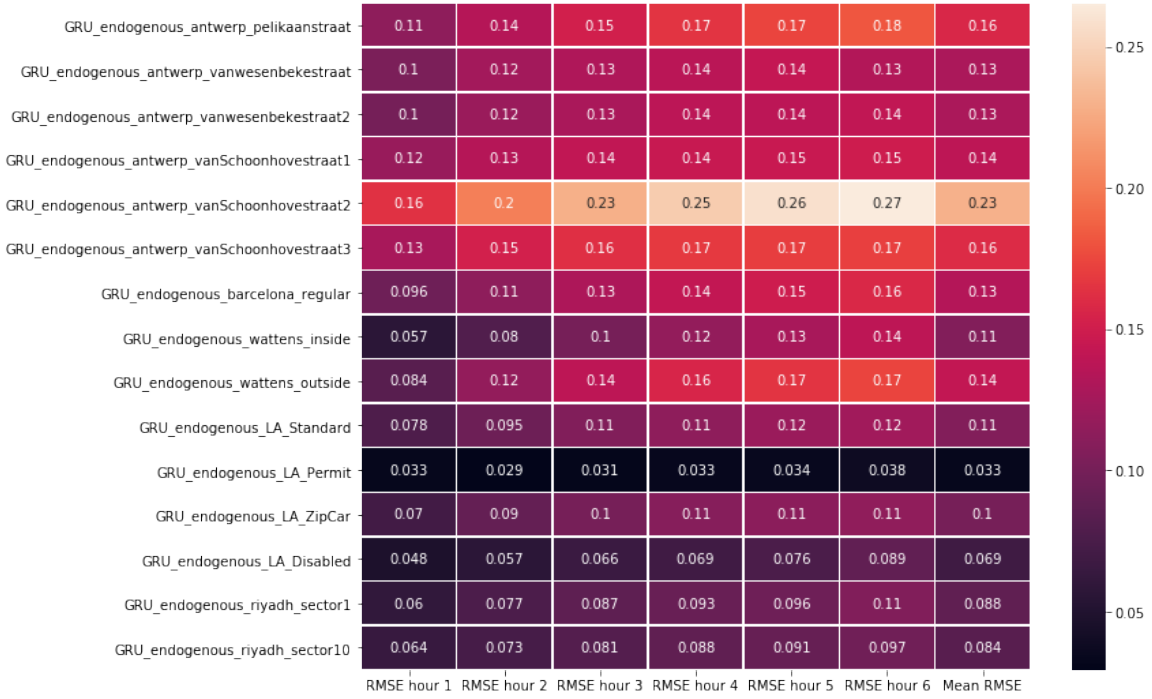


Figure 7.86: GRU RMSE heatmap.

- Similarly to previous methods, the Antwerp scenario achieves the worst RMSE in the Van Schoonhovestraat 2 sector. Notice that the RMSE for the Vanwesenbekestraat sectors increases very little as the forecasting horizon increases.
- The candidate model for the Barcelona sector achieved a similar error rate as the LSTM and MLP models, with a mean error of 0.13.
- The RMSE score of the Outside sector from the Wattens scenario is lower than those presented by the LSTM and MLP models, while the Inside sector is similar to those.
- The experiments for Los Angeles achieved low error rates, but remember that it was necessary to hand-tune them because those proposed by the random search were not able to generalize the test data and they provided forecasts that coincided with the mean value of the time series.
- Both Riyadh sectors achieved low error rates that increase very little as the forecasting steps increase.

7.10.4 Method comparison

Figure 7.87 presents a comparison of the RMSE scores achieved in each of the experiments for the Antwerp scenario. For each sector, observe that:

- For the Pelikaanstraat sector, all three methods behaved very similarly, with a larger error gap between forecasting steps one and two, something that also happens in other sectors.
- For the sector Van Wesenbekestraat Zone 1, the MLP model achieved lower RMSE values than those of the RNN models; but at forecasting hour six, GRU drastically reduces its error rate and matches the error rate of MLP.
- In the sector Van Wesenbekestraat Zone 2, LSTM scores the lowest RMSE for forecasting hour one, but its error rate increases more than with the other methods. At forecasting hour six, it has the highest error rate of all three models. On the other hand, GRU and MLP began with similar RMSE scores for hour one, but at the last forecasting hour GRU is the method with less error.
- In Van Schoonhovestraat Zone 1, the LSTM model starts with the lowest error at the first forecasting hour, followed by GRU and MLP. At forecasting hour three, the RNN methods score similarly; but, from there on, GRU scores lower than LSTM. MLP achieves the highest RMSE values in all the forecasting steps.
- Van Schoonhovestraat Zone 2 has proven to be the sector where the models achieved the worst errors of all the experiments. MLP has the lowest error rate at forecasting step one, but it degrades faster than in the other methods because, at forecasting hour three, it has than highest error rate than any of the other methods. On the other hand, LSTM started as the worst at the first forecasting step, but ended up being the one with lowest error rate at the last forecasting hour.
- Lastly, for the Van Schoonhovestraat Zone 3 sector, GRU is the model that scores the lowest error rates in all the forecasting steps. While MLP started with the second-lowest at the first forecasting step, it ended up close behind LSTM.

Next, the RMSE errors of the candidate models for the Barcelona sector are presented in Figure 7.88, where the RNN methods started with similar scores in the first four forecasting steps, but then GRU started to score lower than LSTM. MLP presents fewer errors in all the forecasting steps.

The visualizations of the RMSE scores for the Wattens sectors are presented in Figure 7.89. For each of the sectors, it can be observed that:

- In the Inside sector, MLP and GRU scored nearly equal in all the forecasting steps, while LSTM performed worse as the forecasting steps increased.
- In the Outside sector, MLP presents an abnormal behaviour, as its starting error decreases a little in forecasting steps three and four, something that has not been observed in any other experiment. The GRU model achieved the lowest RMSE values in all the forecasting steps. While LSTM started at the first forecasting hour with a similar RMSE value as the GRU model, as the forecasting steps increased, its error increased faster than the other two methods.

For the Los Angeles scenario, the RMSE visualizations for method comparison are presented in Figure 7.90. For each sector, it can be observed that:

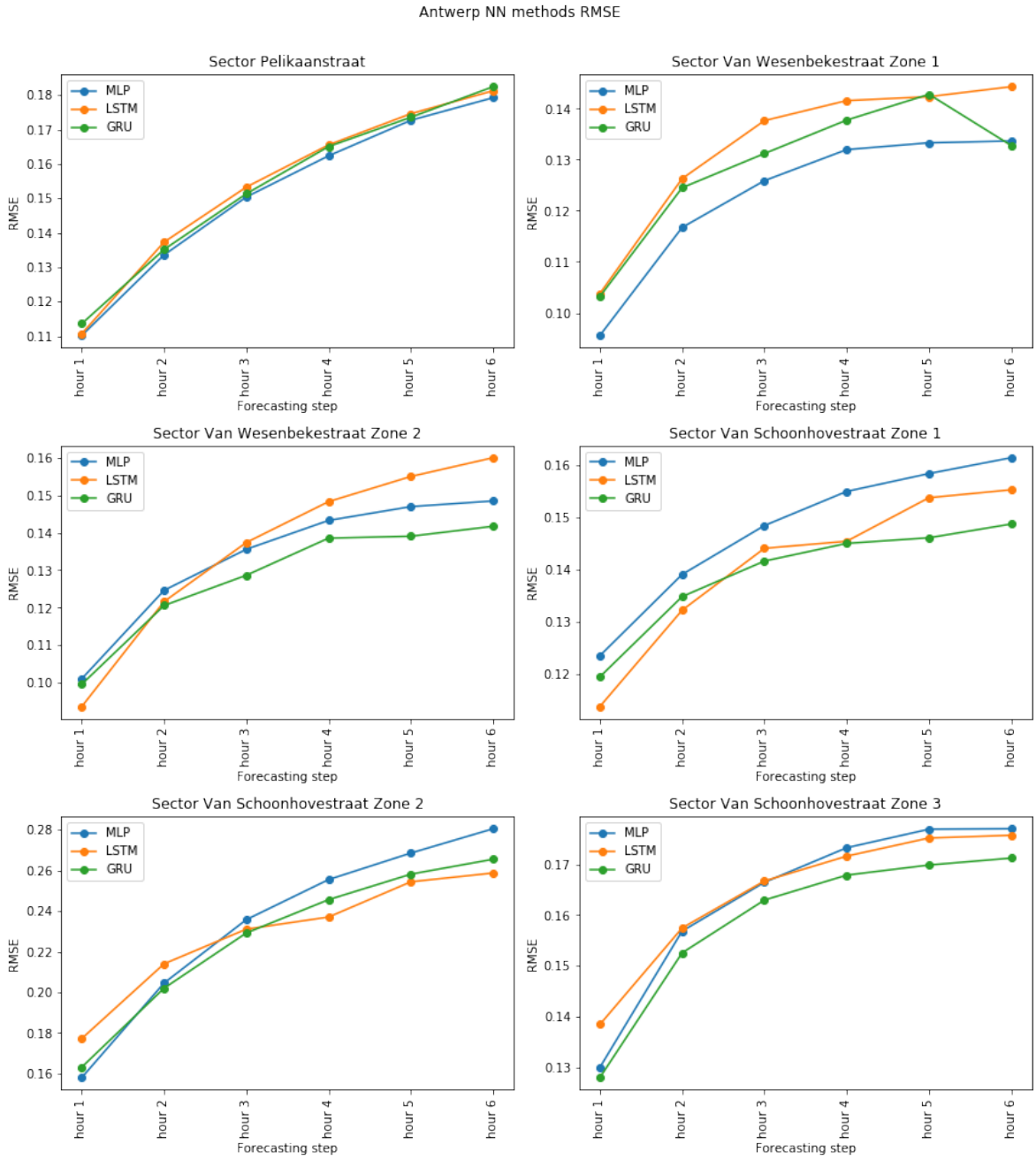


Figure 7.87: Antwerp candidate model NN RMSE.

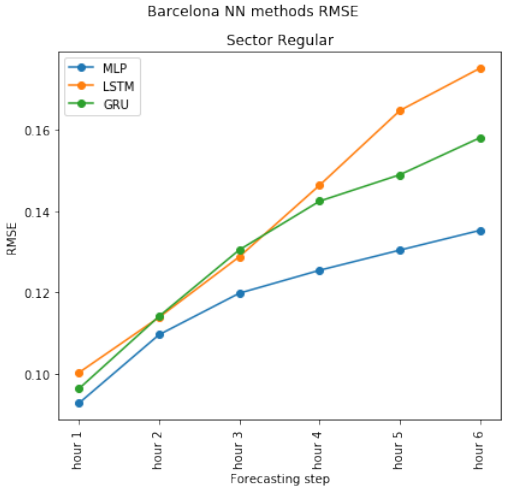


Figure 7.88: Barcelona candidate model NN RMSE.

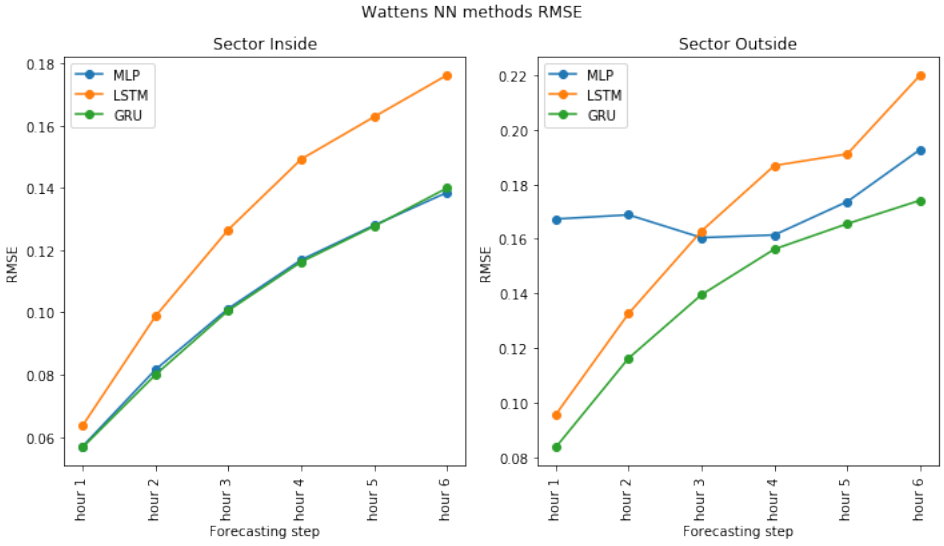


Figure 7.89: Wattens candidate models NN RMSE.

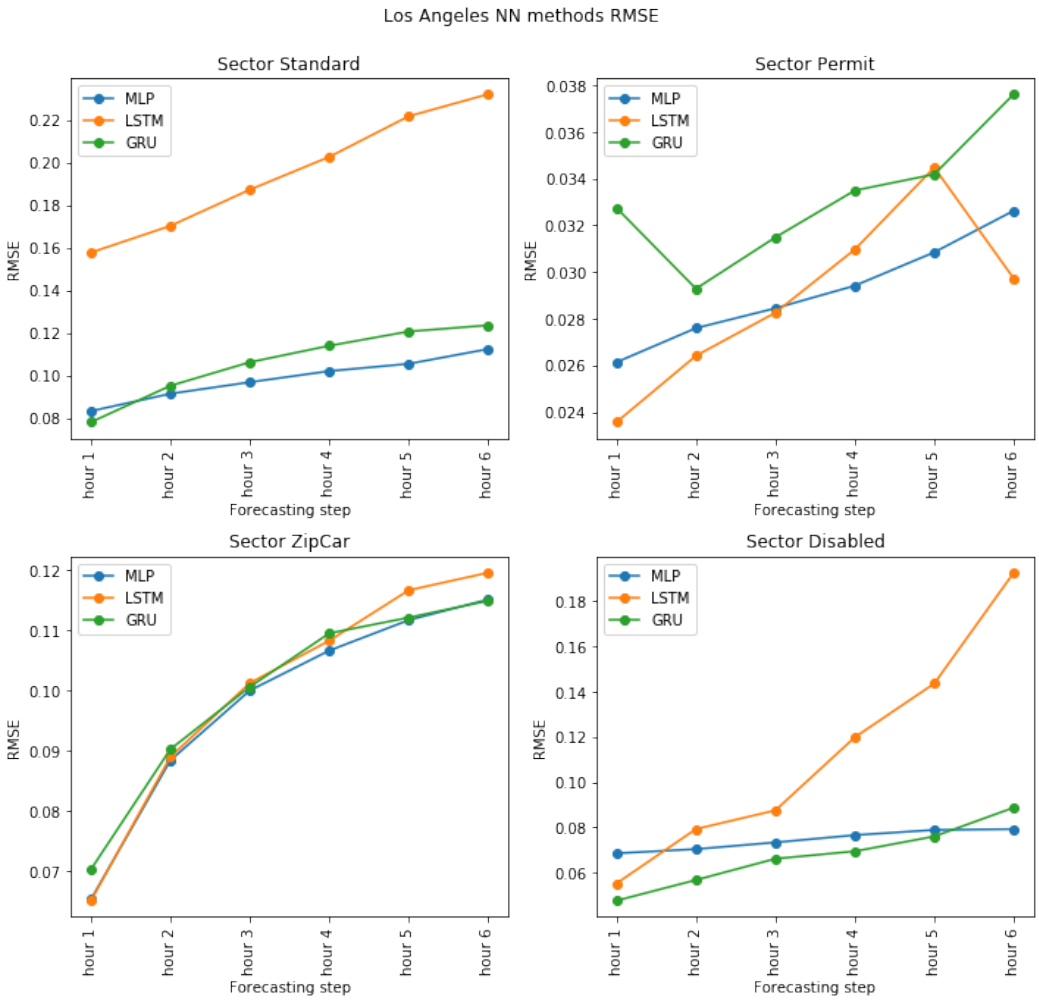


Figure 7.90: Los Angeles candidate model NN RMSE.

- In the Standard sector, MLP and GRU behaved similarly while LSTM presents very bad results.
- In the Permit sector, the RMSE varies little between the models because the street works in the zone cause the occupancy to vary little in time. The differences between the lowest and highest RMSE values are of 0.008.
- In the ZipCar sector, all models behaved similarly, with LSTM achieving the lowest RMSE for the first forecasting hour. However, at the sixth forecasting hour, MLP and GRU scored the lowest.
- For the Disabled sector, the differences are notable between LSTM and the other methods. MLP presents a smaller increase in the error rate throughout the forecasting steps, while GRU started with the lowest RMSE for the first step and ended with a bit higher error rate than MLP at the sixth forecasting step.

Lastly, Figure 7.91 presents the RMSE obtained by the candidate models for the exper-

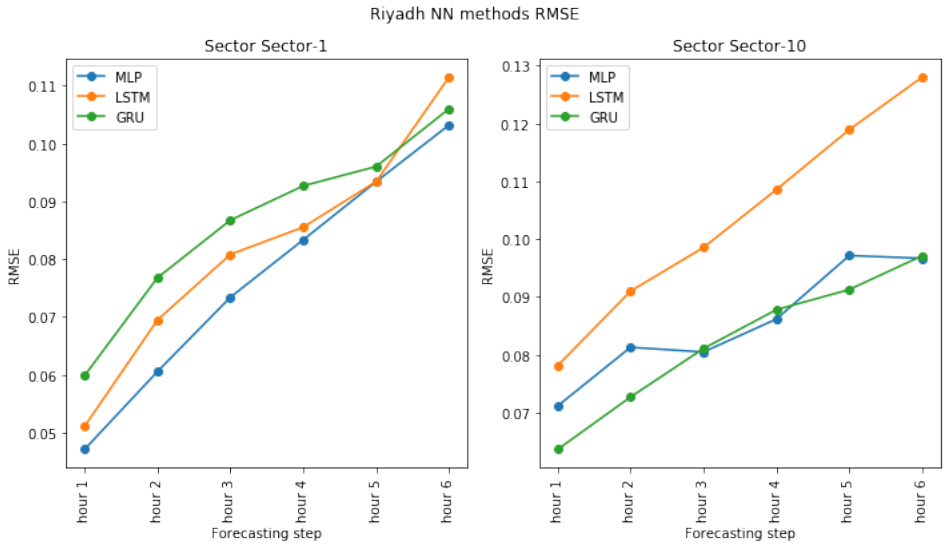


Figure 7.91: Riyadh candidate model NN RMSE.

iments in the Riyadh scenario. The following can be observed:

- In Sector-1, MLP scores the lowest error between the forecasting steps (being matched by LSTM at the fifth hour), thus presenting a nearly linear increase in the error at each step. GRU started with a higher error rate than the other methods in the first forecasting hour, but it ends up scoring second at the sixth hour.
- In Sector-10, GRU presents the lowest error, but it matches MLP at steps three, four and six. LSTM presents the worst scores across all forecasting steps.

Through the different experiments presented throughout this chapter, the following has been observed:

- Due to the nature of NN methods (random initialization, local optima convergence, a huger number of hyperparameters, etc.), it is difficult to infer the effect of the parking sector's behaviour (usage, regular patterns, noise, number of places, etc.) and the obtained model architectures. However, exploring the training losses and the forecast visualizations provides information on how to modify the architecture in order to achieve better scores. An example of this is what happened in the experiments with GRU in Los Angeles.
- There is no clear winner regarding which is the best NN method for parking occupancy forecasting, as MLP and GRU presented very similar RMSE across all the experiments. LSTM is the method that presented the worst scores in nearly all the experiments.
- NN methods presented a high error rate in those sectors with random walk behaviour, such as the sector Van Schoonhovestraat Zone 2.
- Although the models were trained to output six forecasting hours at once, a large

number of experiments has demonstrated that the forecasts come closer to the mean as the forecasting steps increase. Thus, the error rate could be the same as that obtained with a multi-step strategy, which is used by ARIMA models.

- For sectors with distribution changes in the training data, it has been observed that NN methods can overcome this problem.
- In sectors with a high number of sensors that present smooth patterns in their realizations (for example, the Riyadh sectors), a low error rate has been obtained. This indicates that NN models of huge areas with sensors can be perfect candidates for modeling the parking occupancy process.

7.10.5 Conclusions

Now, we can answer the questions that were formulated at the start of this chapter:

- *How different are the sector models based on their idiosyncrasies?*
  It has been shown that model hyperparameters are greatly influenced by the data realizations of the sectors. This can be observed in Riyadh sectors for example, where similar hyperparameters are obtained for both models (Sector-1 and Sector-10) as their data realizations have similar characteristics (levels and regular patterns). So, although the inherent stochasticity is part of the NN methods, if the processess modeled are similar, the obtained hyperparameters for the models are similar.
- *How different are the sector models based on the NN methods' idiosyncrasies?*
  In general, it has been observed that hyperparameters like the optimizer, the number of layers or the number of neurons are different in the models just by their method and this is independent of the sector being modeled.
- *Is there a configuration of hyperparameters that provides better results for the parking occupancy time series?*
  It has been shown that certain values of the hyperparameters have been obtained more times from the random search. These values are constrained to the NN method used.
  For MLP, the optimizer used has been RMSProp and the most used values for the hyperparameters are 64 neurons, 4 hidden layers, 25% dropout, ReLu activation function and batch size of 256.
  For LSTM, the optimizer used has been SGD with momentum and the most used values for the hyperparameters have been 128 and 512 units, 2 hidden layers, 25% dropout, 0 recurrent dropout, He normal as initializer, linear activation function and batch size of value 64 or higher.
  For GRU, the optimizer used has been RMSProp and the most used values for the hyperparameters have been 64 units, 2 hidden layers, 25% dropout, 0% recurrent dropout, He normal initializer, sigmoid activation function and 672 as batch size.

- *Is it possible for an NN model to handle multiple seasonalities?*
  The models have been trained using a windowed approach of twenty four hours and it has been shown that they can adapt to daily and weekly seasonalities.
- *Do nearby sectors share the same model?*
  Because NN stochasticity, nearby sectors do not present the same model but similar ones.
- *Can NN models overcome temporal changes?*
  Yes, it has been observed in Los Angeles sectors that the models can overcome temporal changes if those are present in the training data.
- *Is a model really necessary for each sector?*
  Due to sectors idosyncrasies and NN stochasticity, the best way to model parking occupancy data is with a model for each sector.
- *Which of the NN methods achieved better results overall?*
  MLP and GRU presented the lowest errors in the experiments performed while LSTM scored, in general, worse.

7.11 Lessons learnt

Together with the conclusions, this section also presents the lessons learnt from the empirical experiments of applying novel techniques designed for Deep Learning models. One of the drawbacks when using such novel techniques is that not all of them are useful for all the methods considered, as they were designed for a specific NN method or problem. For example, activation functions like ReLu and techniques like batch normalization were designed for convolutional neural networks; thus, there is little evidence that those will work with other types of methods, such as recurrent ones.

7.11.1 The phases of an RNN model when training on time series

All RNN models trained in this work have shown similar behaviour during training:

1. Learn the mean.
2. Learn recurring patterns.
3. Adjust peaks and valleys.
4. In case of noise or overfitting, the RNN model becomes a naive model by forecasting the most recent input as the most probable for the near future. This also happens if the modeled process is a random walk.

These phases can help the practitioner identify whether the model is underfitting or overfitting the data and how to correct for these. For example, a model that underfits

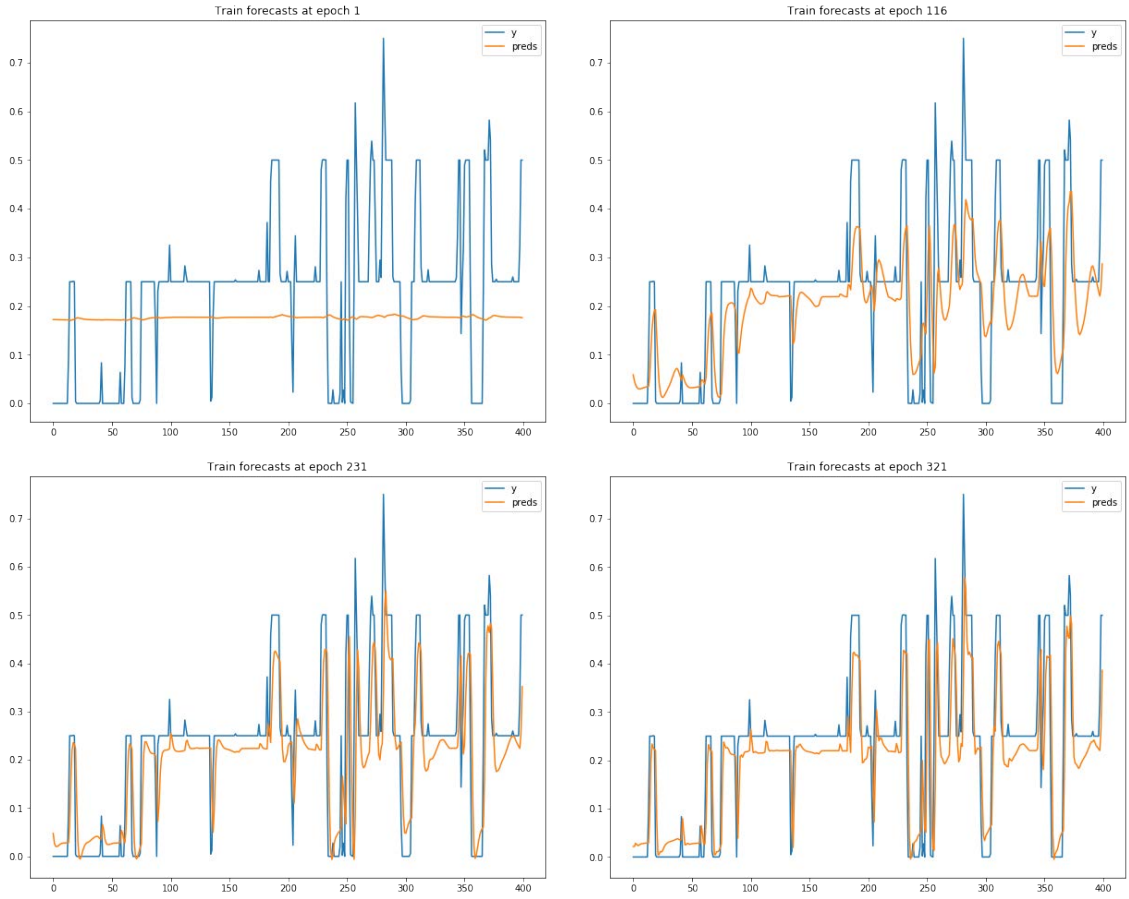


Figure 7.92: RNN model training phases.

the data will only forecast the mean, which can happen if the model complexity is too low or the regularization is too high.

Figure 7.92 presents the visualization of the training phases for a GRU model trained with parking occupancy from the ZipCar sector in Los Angeles. As the epochs increase, the model passes through each of the previously described phases. The overfit (the plot at the bottom right) is noticeable, as the model presents a less smooth visualization when compared to the previous ones.

7.11.2 Undesired local optimum convergence

One major problem that was found is the convergence of the model towards a bad local optima. When the model always predicts the same outcome, it is usually the mean value of the series and indicates that the model is not capable of learning the patterns that govern the generation of the data. This could be because the model is not complex enough (more hidden layers or units) or that the dropout used is too high (usually in the recurrent dropout of the RNN methods). Sometimes the model learns to copy the

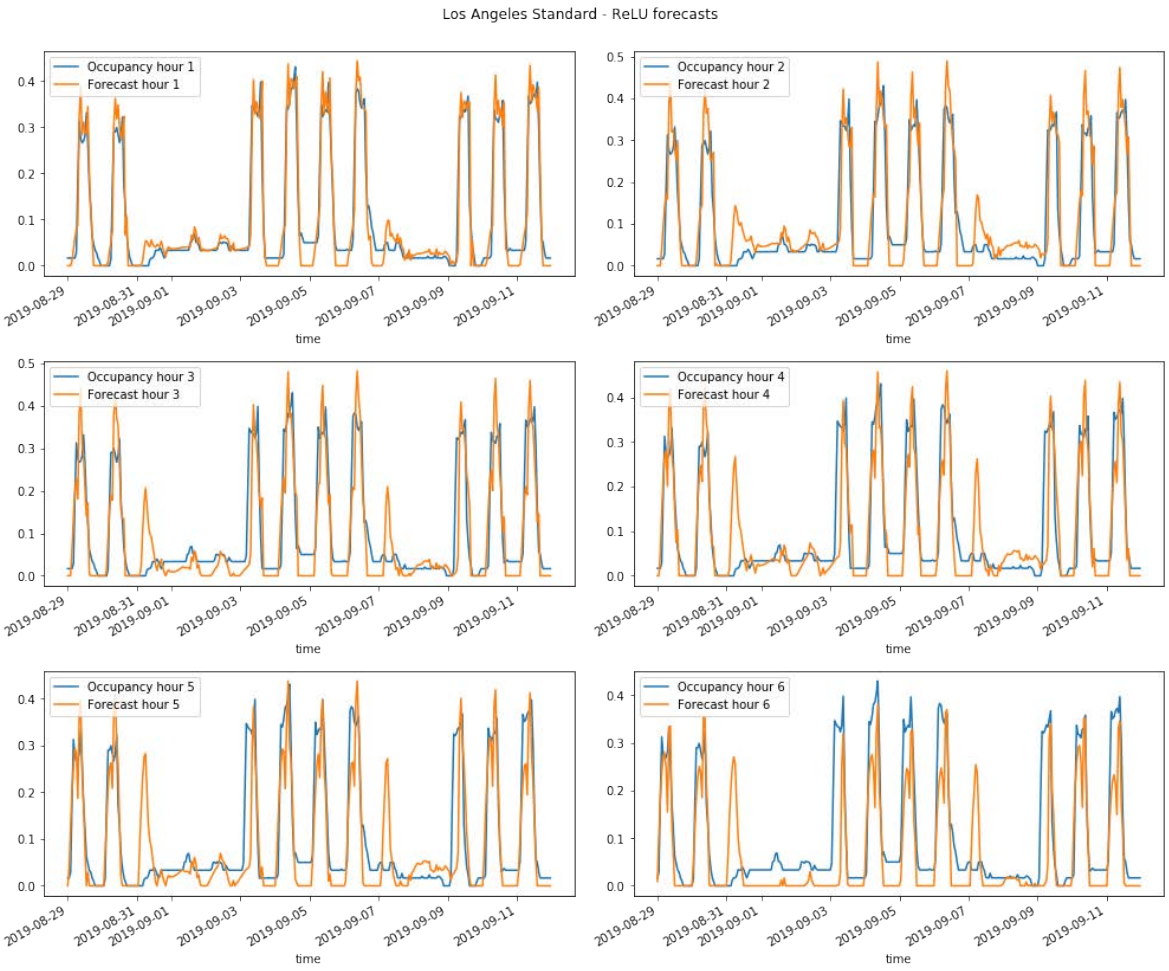


Figure 7.93: Candidate model for the Los Angeles Standard sector with ReLU activation function at the output layer.

last input of the series and thus acts as a naive model. Usually, this indicates that the underlying process that generates the data consists of a random walk or that the model has overfitted the input data and needs regularization.

7.11.3 Negative forecasts

Another eventuality that has been observed is negative forecasts that are not possible in the framework of the problem. In order to overcome such an issue, it is possible to use an activation function in the output layer that is bounded at zero : for example, the ReLU function. The visualizations in Figure 7.93 show the behaviour of the candidate model proposed in Section 7.8.3 when using a ReLU activation function instead of none.

A ReLU activation function forces the outputs to be greater than or equal to zero, but it is necessary to take into account the dying ReLU problem because a bad initialization

could result in a model that is incapable of learning, as the gradient of the ReLU function is zero for the negative numbers. A solution to this could be using a parameterized ReLU (proposed in the work He et al. [2015]), which modifies the ReLU function with a learnable parameter that in turn modifies the domain of the function so that it is no longer bounded at zero, like in the Leaky ReLU.

7.11.4 One-step and multi-step forecasting

One advantage that can be intuitively imposed on a Deep Learning model for time series is a dense output layer (which is used in MLP models), which has as many neurons as it has desired forecasting steps (so the output is a vector in which each element is a forecasting step). This assumption should be taken with a pinch of salt, because there are major factors that each play a role in the success of this strategy. Namely, these are: initialization of the parameters for the output layer; loss function; and data structure. Although taking care of only one of these elements can help achieve good results, all three must be addressed in order to increase the probability of success. The solutions that can be used are:

- Use the same initial parameters for each of the neurons at the output layer instead of making them all different and random. The reason for this is that similar starting values for the parameters allow the model to focus on learning the differences between consecutive values of the series instead of learning the pattern for each neuron.
- Loss functions such as MAE or MSE average the loss and make all neurons at the output layer change their values, although the output of some of them is exactly the expected output. Other loss functions like dynamic time warping or DILATE (Guen and Thome [2019]) can be used to overcome this problem, as those loss functions have been designed with time series in mind, which is in contrast to a generic solution for regression, such as MAE or MSE.
- A proper data structure (time window length, overlap in the time window, number of outputs) is crucial for overcoming the problem. In this work, a time window length of 24 time steps has been used, as the ACF and PACF plots detected higher correlations with the actual parking occupancy during that span of time.

7.11.5 The importance of hyperparameters

NN models in general are difficult to tune, as the effect of changing the hyperparameters and their dependencies is still not well understood. As explained previously, the study of the training loss curves and visualizing the fitting of the model can help improve the models and correct some of the possible problems.

During this work, we found that the following hyperparameters have the greatest effects on the model forecasts (from the most to least important):

1. The optimizer and its learning rate. It has been shown how the methods behaved differently across the three different optimizers (SGD with momentum, Adam and RMSProp) while using a range of learning rate values based on the data of our problem.
2. The number of layers has an effect on the computation time and the learning of the patterns in the model.
3. Dropout affects the training, especially the recurrent dropout used in RNN. As seen with the latter, we found that dropout values above 0 and below 0.25 work well across all models.
4. The number of neurons must not be very low or very high. In this work, hidden layers with neurons of between 32 and 256 worked well.
5. The effect of the activation function is not clear, but ReLU is one of the least used. This could be because of the dying ReLU effect.
6. Normalization should be controlled. It has been shown that batch normalization hurts learning in RNN methods while layer normalization seems to provide better results, but using no normalization does not hurt at all.

7.11.6 Noise in the data

RNNs can be difficult to train, as they can overfit training data and provide the last input as the next forecast, especially when the data is noisy. This work proposes one way to overcome this phenomenon by using an autoencoder with forecasting capabilites. The autoencoder will reconstruct the daily pattern of the data while reducing the noise (similarly to exponential smoothing); and while the model learns to reconstruct the input data (global patterns), a second part can provide predictions for the next forecasting steps by learning local patterns.

The visualizations in Figure 7.94 show the outputs of the autoencoder throughout the training, which demonstrates how it is able to capture recurrent patterns and filter the noise.

The visualization in Figure 7.95 presents the step one forecasts of the autoencoder. It shows that the model is able to forecast the rise and fall of occupancy while maintaining a smooth shape, which indicates that the model has not learned noise. In addition, the autoencoders' reconstruction of the input can be used for anomaly detection. The autoencoder results seem to be promising and would be of interest for further investigation.

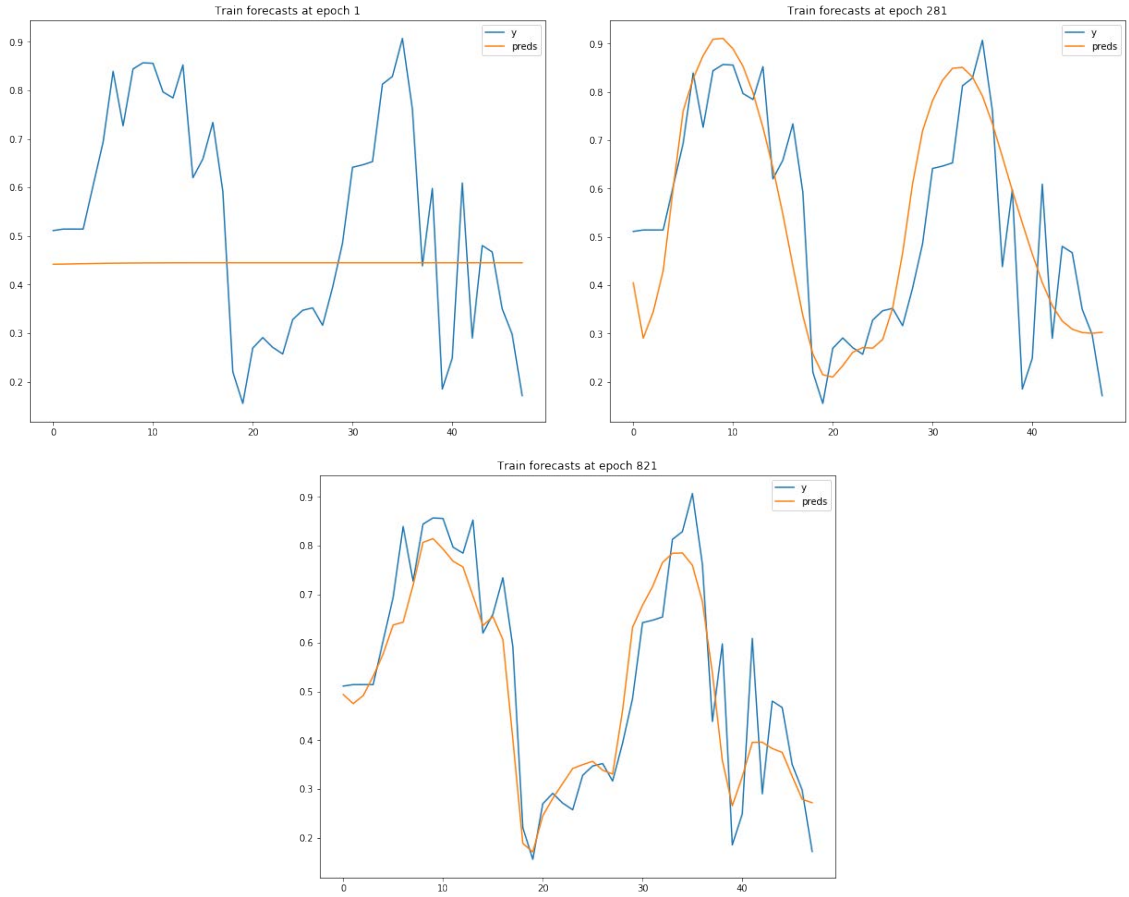


Figure 7.94: Autoencoder training phases with data from the Antwerp sector Pelikaanstraat.

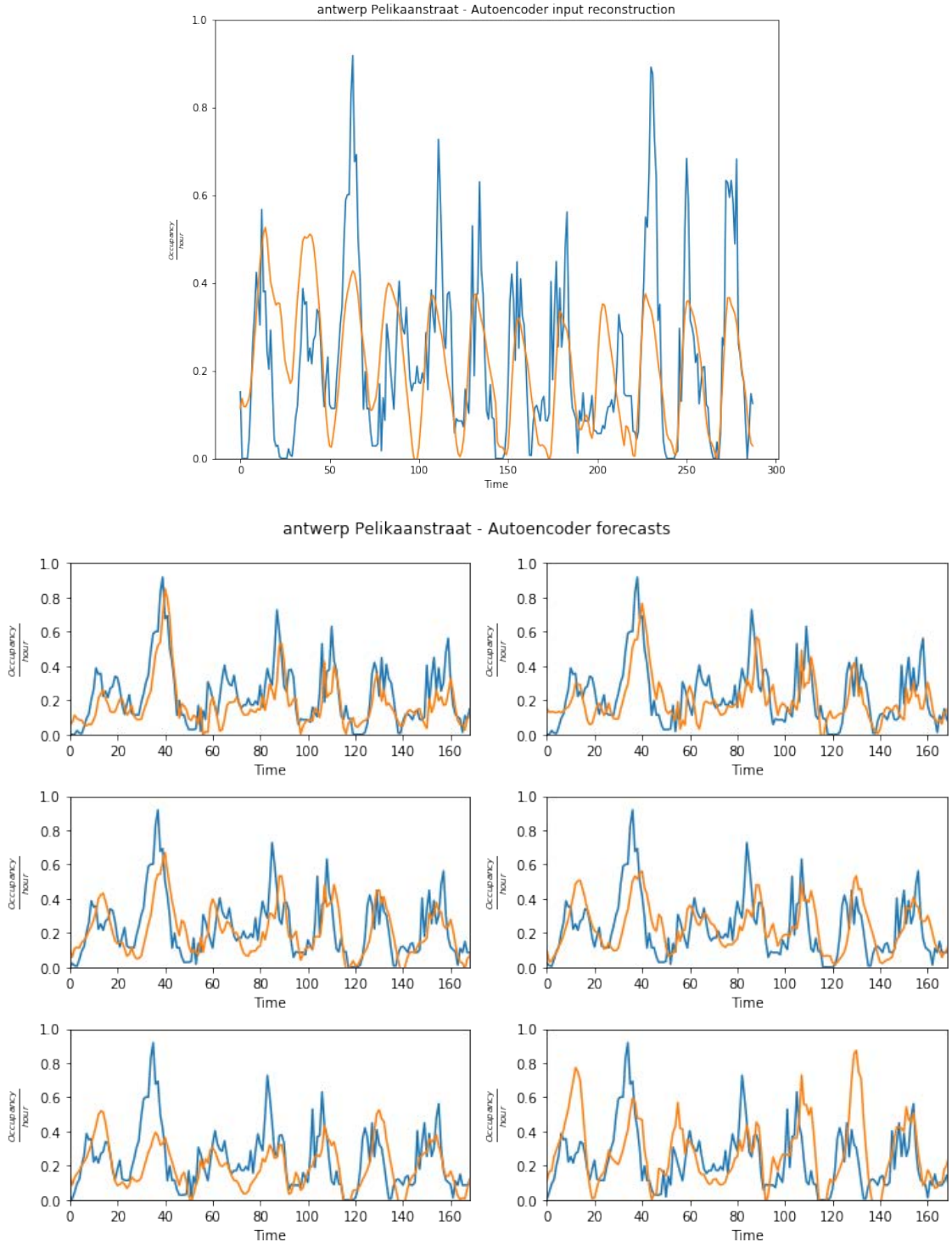


Figure 7.95: Autoencoder outputs: input reconstruction (top) and forecasts (bottom).

8 Neural networks with exogenous information

The previous chapter demonstrated that NN methods achieve a lower RMSE error rate than well-known methods like ARIMA do, although at the expense of model interpretation and statistical properties like confidence intervals. One of the strong points of NN methods is that they can handle data with a high number of dimensions. Focusing on this, this chapter characterizes the different parking sectors that were considered in this work after adding exogenous variables that can have an effect on the systems.

Exogenous variables are those that do not belong to the modeled system but nevertheless have an effect on it. Imposing their values onto the models can improve or worsen the forecasts while providing extra information (interpretability) about the parking systems.

This chapter characterizes a subset of sectors presented in Chapter 3 using Neural Network methods with endogenous and exogenous variables. The goal is to reduce the RMSE and find interesting exogenous variables for adding interpretability to the models. In addition, and similarly to the previous chapter, a study is conducted on the accuracy at different forecasting horizons. Finally, the results are compared to those of the candidate NN models when using only the endogenous information. The objective for this is to evaluate the effects of the different exogenous variables and offer conclusions about their usage.

The subset of sectors used in this chapter are:

- The Pelikaanstraat in Antwerp.
- The Outside sector in Wattens.
- The Standard sector in Los Angeles.
- Sector-10 in Riyadh.

These sectors have been selected because the results achieved by the endogenous models were not satisfactory and they may be able to be improved. Moreover, each of the sectors presents different characteristics and therefore the exogenous variables could have different effects on them.

This chapter is organized as follows. First, the exogenous variables we use are explained, as well as their meaning, the preprocessing conducted on them, and how they are introduced into the models. Next, we study the effects of these exogenous variables and how to interpret them relative to the endogenous variables, then present the proposed

data structure and model architecture used to adjust the models to the new variables. After studying the exogenous variables all together (grouped either categorically or numerically) and individually in order to forecast parking occupancy, we propose the final models based on the lessons learnt and present their RMSE at different time horizons. Finally, this chapter presents comments and conclusions about the obtained results.

8.1 Exogenous variables

One may intuitively think that only a parking system's information is insufficient for building a good model. For example, some days the occupancy is reduced by half of the usual occupancy and after delving into the reasons is it possible to formulate a hypothesis that matches the observations: those days are local holidays.

Taking this as a starting point, we study the effects of two sources of information that intuitively seem to have an effect on parking systems due to their relationships with the human routines. These sources of information are calendar effects and weather conditions.

8.1.1 Description

The exogenous variables we use are the following:

- Calendar effects. Parking behavior can be affected by so-called calendar effects. Occupancy on a weekend is different than on weekdays, but the pattern is maintained over weekends. In order to capture such patterns, this study uses the following calendar information:
  - Day of the Week: Categorical. Integer value between one and seven that identifies the day of the week.
  - Season: Categorical. Integer value between one and four that identifies the season.
  - Month. Categorical. Integer value between one and twelve that identifies the month.
  - Weekday (working day): Categorical. Binary value that indicates if the day is a weekday or on the weekend.
  - Holiday: Categorical. Binary value that indicates if it is a local or global holiday.
- Weather conditions. The objective of using this source of information is to ascertain whether the parking occupancy patterns are modified by weather conditions, as they have an impact on human routines. The information related to weather is:
  - Temperature: Numerical. Self-explanatory.

- Cloud Cover. Numerical. Value in the 0–1 interval indicating the percentage of sky occluded by clouds.
- Dew Point: Numerical. Dew point temperature.
- Humidity: Numerical. Value between zero and one representing the humidity.
- Wind Bearing: Numerical. Wind direction in degrees.
- Wind Speed: Numerical. Wind speed in miles per hour.
- Wind Gust: Numerical. Wind gust speed (miles per hour).
- UV Index: Numerical. Values in the interval bounded at zero and unbounded above.
- Visibility: Numerical. Average visibility as distance (miles).
- Precipitation Intensity: Numerical. Inches of water per hour at a given time.
- Precipitation Accumulated: Numerical. Amount of snowfall accumulated in inches per hour.
- Precipitation Type: Categorical. If precipitation, this describes its type. Possible values are *snow* or *rain*.
- Weather summary for each hour: Text. A readable text that summarizes the weather at a given hour. Possible values are *clean*, *ra*iny, *overcast*, etc.
- Weather summary about the day: Text. A readable text that summarizes the weather on a given day. Possible values are sentences like *Overcast throughout the day* or *Possible light rain in the morning with clear skies at night*.

Through this chapter the value that a categorical variable can take is written in italic letters in order to help the reader to differenciate them from the common usage of the word. For example: foggy days are usually indicated when the Summary categorical variable has the value *foggy*.

8.1.2 Preprocessing

In order to integrate the exogenous variables for modeling, it is necessary to preprocess them to ensure that the models can make use of them. Things to consider are the types of variables and their scale. Exogenous variables can be of two types:

- Categorical: Cannot be handled directly by a model because they are described as text or as an integer value with low cardinality, such as the day of the week that is described as an integer from one to seven in order to designate the seven days of the week. For a human, these integers can mean a given day, but they are just numbers to a computer.
- Numerical: Continuous variables usually in the $\mathbb{R}$ domain. These are the most common types of variables.

To digest these types of variables, NN models must satisfy some constraints:

- In the case of categorical data as text, it must be preprocessed and mapped to the $\mathbb{I}$ domain (like the day of the week). This is known as ordinal representation. It is better to convert the ordinal representation to one-hot encoding. This is a sparse vector of size N that is equal to the number of possible values of the variable. In this vector, all inputs are zero, except for the position that corresponds to the integer value used to codify the value of the variable, which will be one. This approach uses integer values in which the categorical variable values have been mapped as an index whose vector has the value of one. This is a common approach in machine learning and statistical methods, but NNs have problems with such sparse codification. Thus, these representations are transformed to dense valued vectors known as embeddings. Embeddings are part of the NN input layer and are learnt during the training phase of the model.
- For numerical data, all the variables must be in the same scale. Moreover, ML methods like NNs, work better if the data has mean 0 and unit variance. This is essential for ensuring that high-scale variables do not have higher weights and are thus considered more important than others. Furthermore, it avoids vanishing and exploding gradients. Numerical data is usually preprocessed by a normalization method.

Those variables described as type text in the previous section are sequences formed by categorical variables, in which each word (normally called a "token" in the Natural Language Processing literature) has a vocabulary value. If we use one-hot encoding for this type of information, the size of the sparse vectors become very large and computationally inefficient. For example, a text with 200 different words will create one-hot vectors for each word that consists in 199 zeros and 1 one. The embeddings mentioned previously are the key component for handling these types of variables.

What is more, this work uses a windowed approach for creating time series and it is therefore necessary to adapt the exogenous variables to this methodology. Because all the numerical variables are also hourly time series, they have been preprocessed to create 24-element time series in the same way as the occupancy variable. For the categorical variables, a similar approach has been used. This ensures that those categorical/text variables that have changing hourly frequencies can provide useful information at each step, just as the numerical variables do. However, for those exogenous variables that change at higher frequency (likethe variable Month that in average its value changes every 30.42 days) their formed sequences changes very little.

8.1.2.1 Embeddings

Embeddings are formed by using a set of densely connected neurons as part of the input layer of a NN model. This set of neurons uses words codified as integer values (they could also be a one-hot encoded vector), and those integer values are mapped onto a dense vector of values that are learned at the same time that the model is trained, in

other words, the weights of these neurons act as a lookup table that maps the codified words as integers to a vector representation. Once the model is trained, each of the dense vectors offers a unique representation of the token (word or categorical value) with which it is bounded. Moreover, as those vectors have been created during training, they capture the semantic meaning of the token in relation to the problem or task that the model has been trained to model. Finally, it is necessary to consider the size output of an embedding layer, as its input is a vector of integer values (tokens) and it returns a matrix of size $(sequence\,length, number\,dense\,neurons)$. Sequence size is used to determine beforehand a fixed size for the input of the embedding layer. As part of the preprocessing, all sequences must be of the same length and this is achieved by padding the sequences (adding zeros for example). The padded values are masked in order to save computation by using a mask. The number of dense neurons is used in order to determine the length of the embedding vectors (more neurons, larger the embeddings). Taking this into account when training an NN model with N texts representing a data row composed of words (a matrix), the embedding layer output will be a tensor of three dimensions. This transformation into a three dimensional vector makes them perfect candidates for use as input in recurrent layers. Detailed information about embeddings is provided in Mikolov et al. [2013], Guo and Berkhahn [2016].

8.1.2.2 Numerical normalization

There exist different normalization techniques for providing better representations of the numerical variables for NN models. In this work, we use standardization (presented in Equation 8.1) and scaling (see Equation 8.2).

$$z_{scores} = \frac{X - \mu_x}{\sigma_x} \tag{8.1}$$

$$X_{scale} = \frac{x - min(x)}{max(x) - min(x)} \tag{8.2}$$

Standardization is commonly used in statistics to center the variables to zero mean and unit standard deviation, thus making them a normalized standard distribution. On the other hand, scaling changes the range of the data so that it comprises the interval between zero and one (although the formula can be modified in order to have any desired interval). As mentioned previously, ML and statistical methods usually require that the data be normalized, but this can also depend on the application. In order to know which method provides better models (smoother training and lower loss values), 20 models were created for the scenarios, and the losses from each scenario's model were aggregated for comparing which method provides better models. All models have the same structure:

- 2 GRU layers of 128 units each and a dense layer with six neurons as the output (schema of simultaneous multiple predictions).

- Adam as optimizer.
- Batch size of 128.
- Hyperbolic tangent as GRU activation function and linear activation function at the output layer.
- 30 epochs, no early stop.
- MSE as loss function.

Figure 8.1 shows the averaged training and validation loss from 20 models trained with all the exogenous variables together and normalized by means of standardization (blue) and scaling (orange). Each visualization comes from the four scenarios. The important points for extracting useful information from these plots are: the lower point of the validation losses; the gap between training and validation losses at the last epoch (30); and the variability at each epoch (vertical lines). The scaled method is superior in the Antwerp and Los Angeles scenarios, while its scores are equal to the standardized method in Wattens and Riyadh. The gap between training and validation loss is larger for the scaling method, meaning that the model is still far from overfitting the data, while the losses from the standardized method cross each other in Antwerp, Wattens and Riyadh. This indicates that the scaled method allows for longer training, which gives the model room to improve before overfitting. Lastly, the variability at each epoch is similar for both methods.

In light of these results, this work uses scaling to normalize all the exogenous variables. The endogenous variable is not normalized, as it is already in the 0-1 interval.

8.2 Parking characterization

A study has been done to characterize the behaviour of exogenous variables and occupancy. In order to obtain useful information, different models were created using different inputs in order to provide occupancy forecasts for the next six hours. The models are:

- A model using only categorical variables.
- A model using only numerical variables.
- A model for each of the exogenous variables.

These experiments will provide insights into the relationship between parking occupancy and the exogenous information. In the case of the categorical variables, a study of the embeddings can provide useful information about the parking systems.

8.2.1 All the categorical variables

A model has been created using the following variables as input :

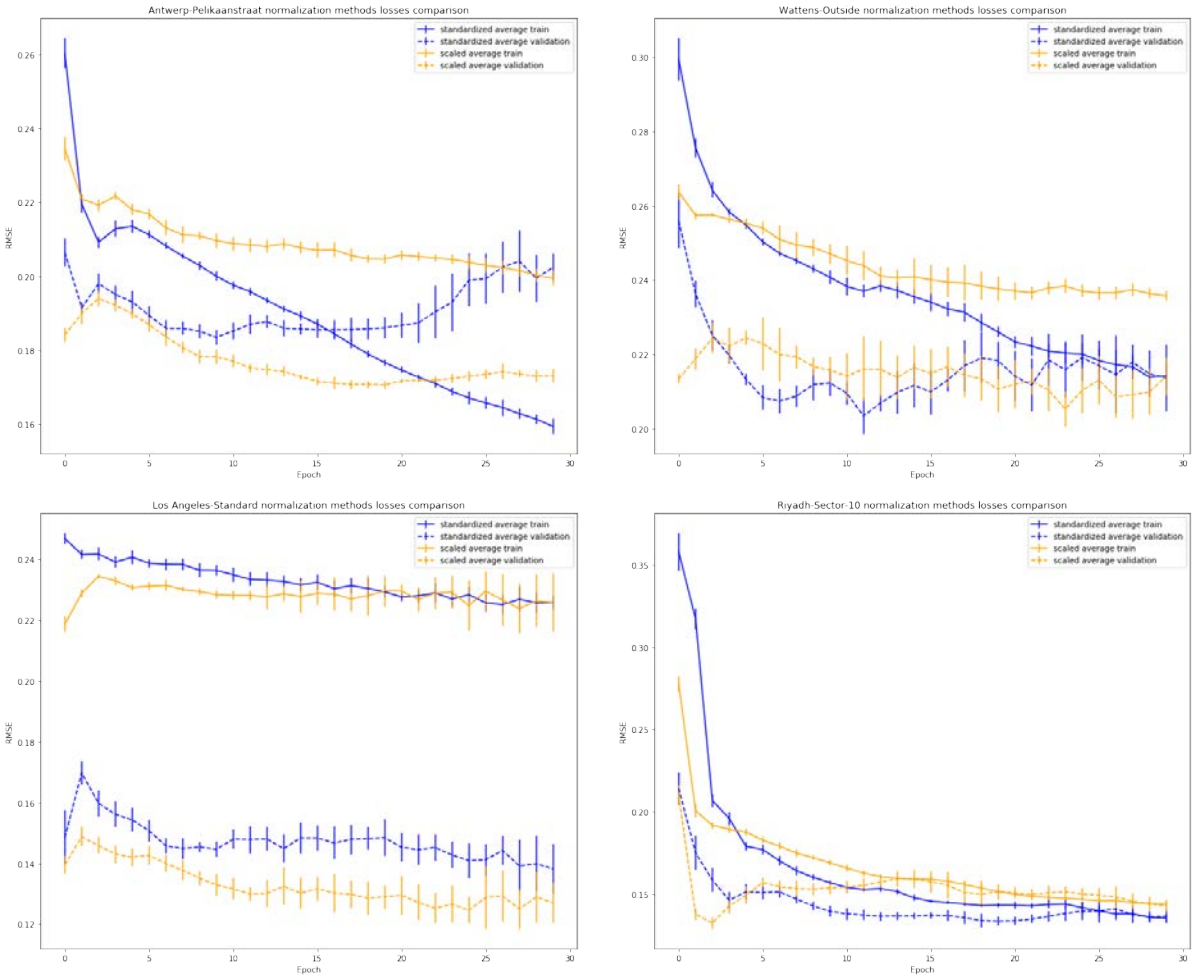


Figure 8.1: Normalization techniques. Losses in 20 models (averaged) for each scenario using standardization and scaling. Vertical lines represent the standard deviation at that epoch. From top left to bottom right: Antwerp Pelikaanstraat, Wattens Outside, Los Angeles Standard and Riyadh Sector-10.

- Day of the Week.
- Season.
- Month.
- Weekday (working day).
- Holiday.
- Weather precipitation type.
- Weather summary for an hour.
- Weather summary about a specific day.

The model consists of:

- An embedding layer with 128 neurons for each variable. Each embedding layer is connected to a GRU layer of 64 units. The output of the recurrent layers are then concatenated through the first dimension (columns), forming a matrix of shape $(batch\,size, 512)$ where the 512 columns come from the eight GRU layers (8*64). This matrix is used as input to the output layer that consists of the well-known six dense neurons that try to predict the next six hours of forecasts.
- Adam optimizer.
- 50 epochs with early stop.
- 128 as batch size.
- RMSE as loss.

The usage of a recurrent layer after embedding is due to the output nature of the embedding layer, as it is a tensor of three dimensions ($batch\,size$, $sequence\,length$, $embedding\,dimension$). These three-dimensional tensors (where the second dimension represents the sequence length, i.e., the time step) are handled better by recurrent layers, although there exist other options like flattening the tensor into a matrix (and losing temporal dimension) or using some kind of operation like global pooling that averages the tensor.

The concatenation of the GRU outputs is interpreted as a matrix where each feature (one of each 512 vectors in the first dimension) represents a summary of the twenty-four previous values of the sequence, as each token has been converted into a dense vector and the sequence of these vectors has been summarised in time by the recurrent layer (one per feature). In this approach, our embeddings are projected into different feature spaces. Other alternatives could be to create unique categorical variables that have the values of all the others and project them all into the same feature space.

A visualization of the previously explained model architecture is presented in Figure 8.2.

Antwerp

Figure 8.3 presents the forecasts at six consecutive hours for the Pelikaanstraat sector in Antwerp. Although the endogenous variable has not been used as input for the model,

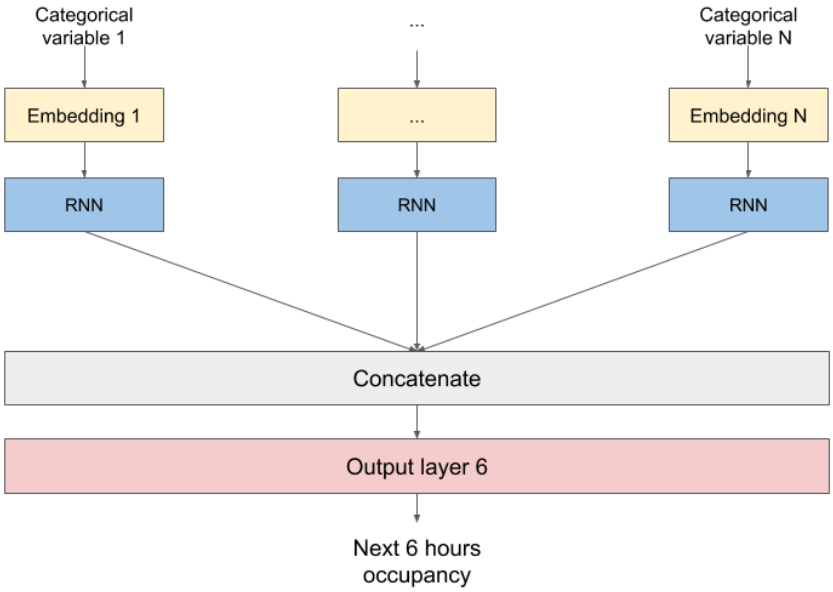


Figure 8.2: Schema of the architecture used to model all categorical variables for forecasting parking occupancy values.

the figure shows that the model has learnt the recurrent patterns of parking occupancy that occur every day. Moreover, occupancy can be differentiated according to the day of the week, as shown by more occupancy being indicated on Sundays and less on Saturdays (April 13 and 14, respectively), something that was observed in Chapter 3.

Wattens

The visualizations in Figure 8.4 show the forecasts at different step horizons for the Wattens Outside sector. The model has been able to learn some of the occupancy patterns during workdays and weekends, like in the Antwerp scenario. Moreover, it is capable of distinguishing that the occupancy pattern is different on Fridays compared to the other working days.

Los Angeles

The plots in Figure 8.5 pertain to the forecasts at six time steps for the Los Angeles Standard sector using only categorical information. Observations are similar to those of the previous scenario models, although this model fails to detect the lowest levels of occupancy.

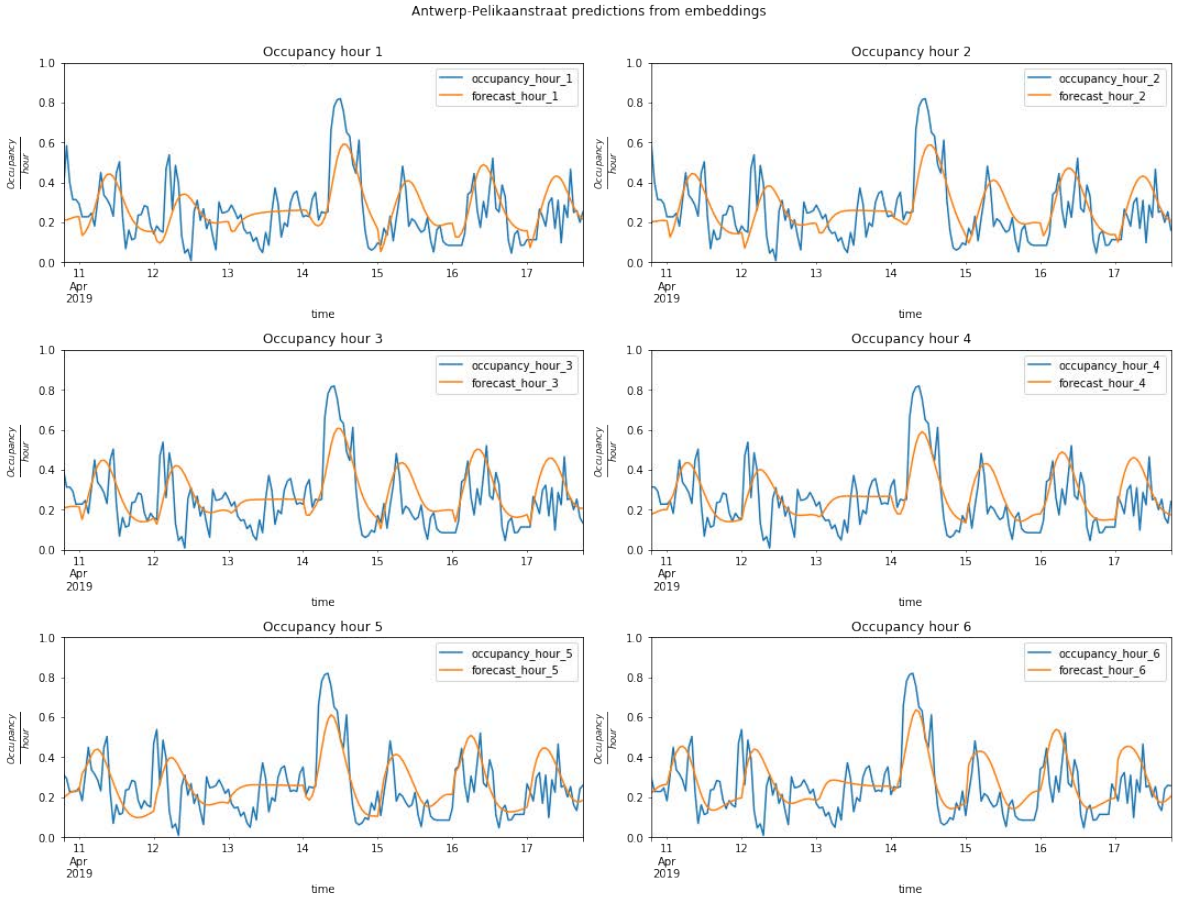


Figure 8.3: Forecasts for the next six hours from a model using only categorical inputs for the Pelikaanstraat sector in the Antwerp scenario.

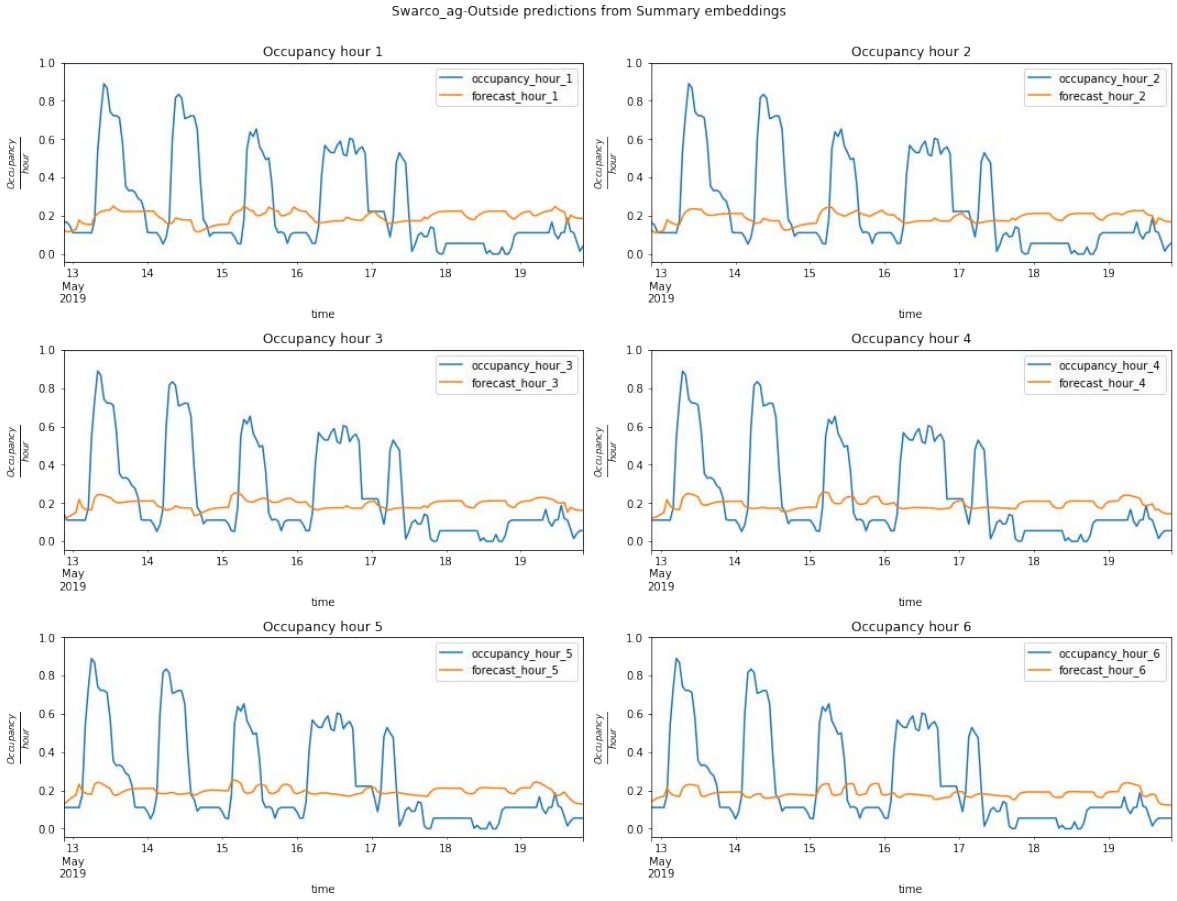


Figure 8.4: Forecasts for the next six hours from a model using only categorical inputs for the Outside sector in the Wattens scenario.

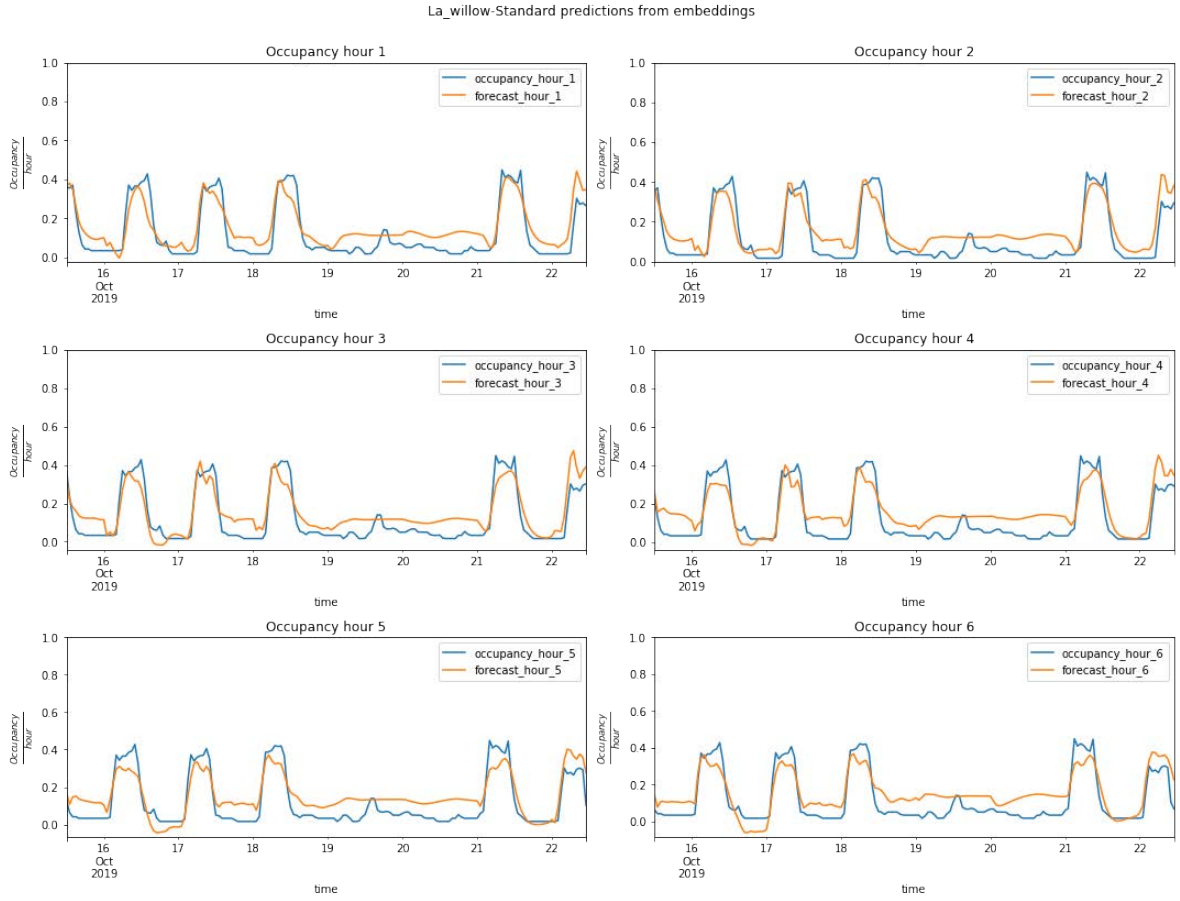


Figure 8.5: Forecasts for the next six hours from a model using only categorical inputs for the Standard sector in the Los Angeles scenario.

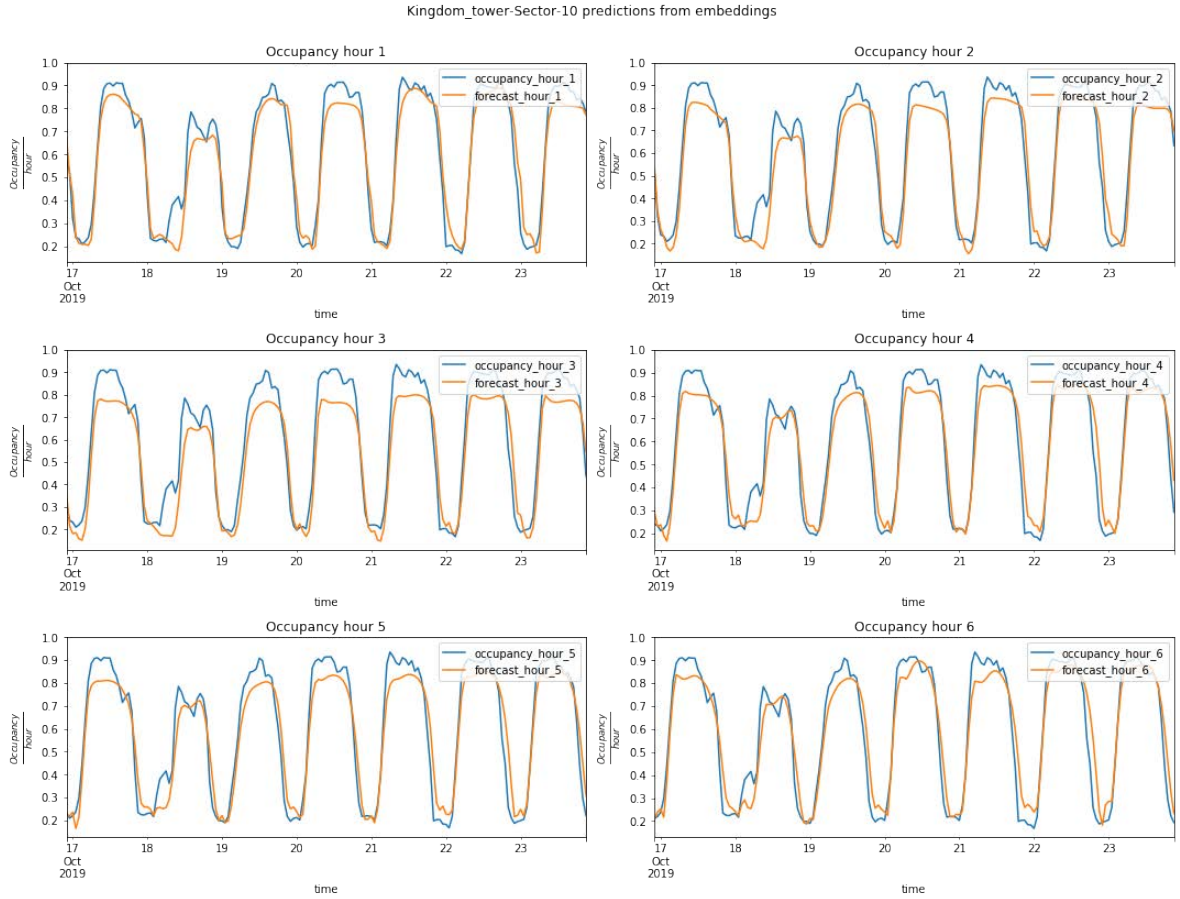


Figure 8.6: Forecasts for the next six hours from a model using only categorical inputs for Sector-10 in the Riyadh scenario.

Riyadh

The last scenario corresponds to Sector-10 in Riyadh. The visualizations of the forecasts at different time horizons presented in Figure 8.6 show that the model has learnt the occupancy patterns of the parking sector throughout the days. The regular behaviours that occur in the Riyadh sectors can easily be modeled with calendar and weather information.

Those four models have shown that it is possible to model parking occupancy patterns using exogenous variables, but the level of error is conditioned by the regularity of the patterns in the modeled sectors. Moreover, the exogenous variables that have been used correspond to categorical information of different types, like ordinals, categories and texts. This shows that the proposed architecture based purely on the NN methodology is a good candidate for modeling such types of problems.

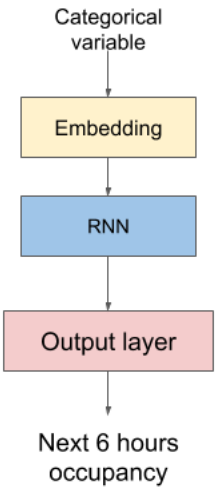


Figure 8.7: Architecture used for the models with only one categorical variable.

8.2.2 Categorical variables, one by one

A model has been created for each of the categorical variables with the goal of forecasting the parking occupancy for the next six hours. This approach will provide insights into which categorical variables contribute the most to the forecasts. In addition, the embeddings from each input are inspected to find clues about the relationship between the categorical variables and the parking occupancy.

The models consist of:

- An embedding layer with 128 neurons connected to a GRU layer of 64 units. The output is used as input for the output layer, which has six neurons.
- Adam optimizer.
- 50 epochs with early stop.
- 128 as batch size.
- RMSE as loss function.

The architecture of the models with one categorical variable is presented in Figure 8.7.

Antwerp

Figure 8.8 presents the occupancy forecasts for the next six hours from the model created with the categorical variable Summary. The model forecasts the mean of the series when

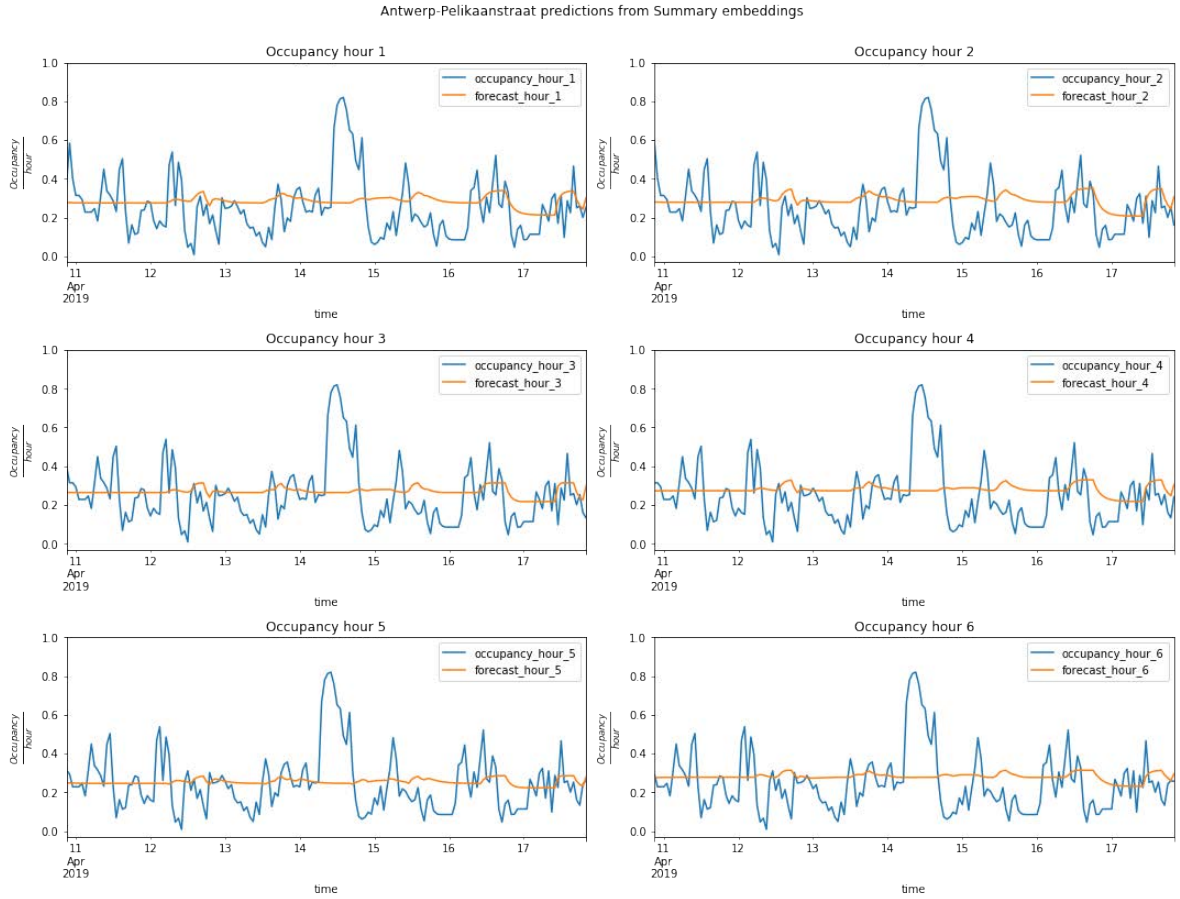


Figure 8.8: Antwerp Summary variable forecasts.

the weather condition is the same during the day (11 April was clear at all hours), but changes related to different weather conditions can be noticed in the forecasts, as days 16 and 17 became overcast, foggy and rainy in the afternoons and evenings (shown in Table 8.1), which the model used to forecast lower values of parking occupancy.

It is possible to extract useful information from the embedding models, as they capture the semantic relationships between the inputs and the outputs. The steps to do that are:

1. Extract the weights of the embedding layer.
2. Perform dimensionality reduction using Principal Component Analysis (linear method) and/or t-Distributed Stochastic Neighbor Embedding (tSNE) (non-linear method) on the embedding weights in order to project the data into a lower dimension and obtain clusters of similar tokens.
3. Visualize the data.

Some notes about tSNE: although tSNE can be used directly, it is recommended to first use PCA to reduce computational loading by filtering redundant dimensions. This is

time	precipType	summary	summary2
2019-04-17 00:00:00	rain	Overcast	Mostly cloudy throughout the day.
2019-04-17 01:00:00	rain	Overcast	Mostly cloudy throughout the day.
2019-04-17 02:00:00	rain	Overcast	Mostly cloudy throughout the day.
2019-04-17 03:00:00	none	Overcast	Mostly cloudy throughout the day.
2019-04-17 04:00:00	none	Overcast	Mostly cloudy throughout the day.
2019-04-17 05:00:00	none	Overcast	Mostly cloudy throughout the day.
2019-04-17 06:00:00	none	Overcast	Mostly cloudy throughout the day.
2019-04-17 07:00:00	none	Overcast	Mostly cloudy throughout the day.
2019-04-17 08:00:00	none	Overcast	Mostly cloudy throughout the day.
2019-04-17 09:00:00	none	Overcast	Mostly cloudy throughout the day.
2019-04-17 10:00:00	none	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 11:00:00	none	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 12:00:00	none	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 13:00:00	rain	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 14:00:00	rain	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 15:00:00	rain	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 16:00:00	rain	Overcast	Mostly cloudy throughout the day.
2019-04-17 17:00:00	rain	Overcast	Mostly cloudy throughout the day.
2019-04-17 18:00:00	rain	Overcast	Mostly cloudy throughout the day.
2019-04-17 19:00:00	rain	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 20:00:00	rain	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 21:00:00	rain	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 22:00:00	none	Mostly Cloudy	Mostly cloudy throughout the day.
2019-04-17 23:00:00	none	Clear	Mostly cloudy throughout the day.

Table 8.1: Antwerp Pelikaanstraat weather 17 April 2019.

because tSNE is an algorithm that requires a lot of computational and memory resources, due to it having an $O(N^2)$ complexity (where N is the number of data points). Then, tSNE is used to reduce the dimensionality to two or three dimensions by mapping the high dimensional data neighbourhoods in a lower dimension.

The tSNE method has a hyperparameter called perplexity, which is related to the bandwidth of the distributions in the high dimensional data (as it uses a radial basis function), meaning that higher values of this hyperparameter permits to have larger neighbourhoods. In order to obtain interpretable visualizations, it is necessary to try different values for the perplexity hyperparameter. What is more, one could tune the learning rate of the optimizer as the algorithm makes use of SGD. Although tSNE preserves local structure, we must be cautelous with interpretations when using this method, specifically because there are no guarantees about the preserving of the global structure of the high dimensionality space when it is mapped to a lower one. Thus, interpreting cluster gaps must be done cautiously (in other words, the distances between clusters do not mean anything). The work by van der Maaten and Hinton [2008] provides details about using this methodology to visualize this type of high dimensional data.

Using the cosine distance (shown in Equation 8.3), the distance of the dense vectors can be measured from the trained embeddings to obtain semantic relationships between the words represented by the dense vectors represent in the current task. The cosine distance output is in the interval between 0 and 2, where 0 means equal vectors, 1 is an orthogonal vector, and 2 represents vectors with opposite directions.

$$d_{cosine} = 1 - \frac{a \cdot b}{\|a\| * \|b\|} \tag{8.3}$$

Figure 8.9 shows the distance matrix obtained when the cosine distance is applied to the embeddings of the Summary variable as a heatmap. As the embeddings are learned by modeling the input data to forecast parking occupancy, their distances are conditional on the parking occupancy.

Figures 8.10 and 8.11 present the visualizations of the embeddings from the variable Summary after applying PCA and tSNE. Our interpretation is that similar visualizations mean that the information lost by the subset of principal components selected in the PCA (i.e., the total variance ratio explained by the sum of the subset of principal components) is low and, furthermore, that the relationships are linear. Moreover, this allows using PCA interpretability methods that give meaning to the visualization, as the meaning of the principal components is based on the variance of the features.

By observing the heatmap of distances, the PCA, and the tSNE visualizations for the Summary variable, it is possible to interpret that the parking occupancy is conditioned by different weather conditions. PCA shows that *flurries*, *heavy snow*, *wind* and *fog* are different from the rest of the Summary values, while tSNE shows the following neighbourhoods:

- *Mostly heavy rain and fog.* Weather conditions related to rainy days.

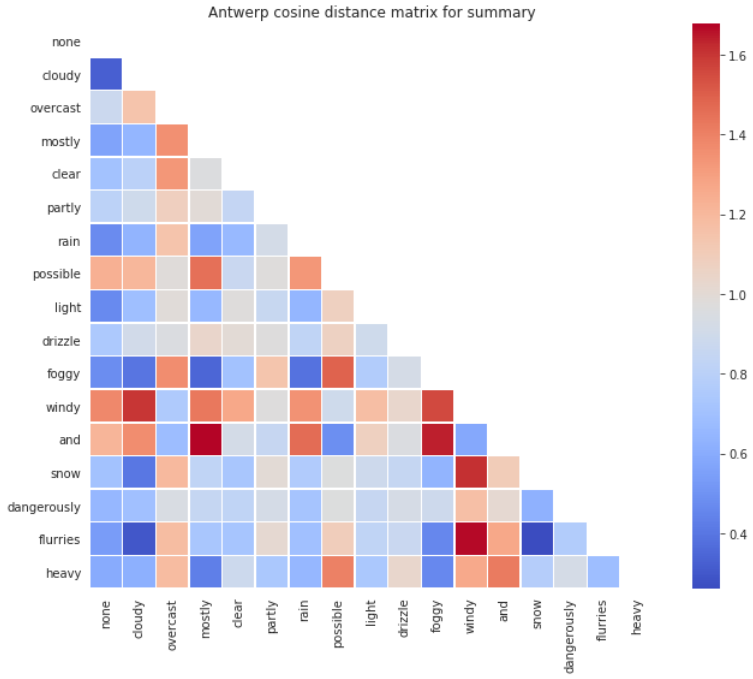


Figure 8.9: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the values of the variable Summary.

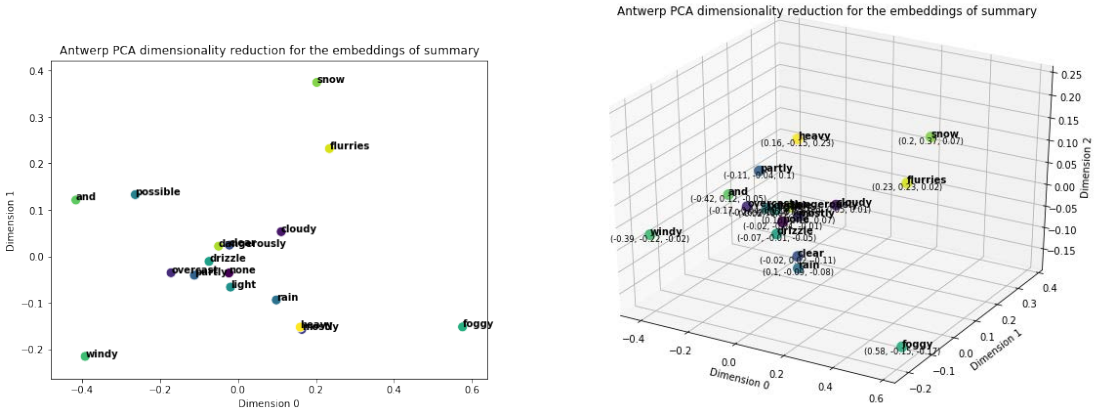


Figure 8.10: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Summary after PCA. .

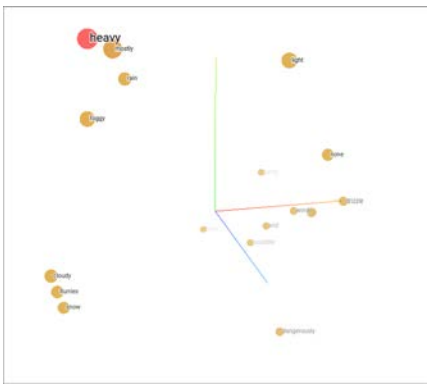


Figure 8.11: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Summary after tSNE.

- *Cloudy, flurries and snow.* Weather conditions related to bad weather.
- *Dangerous.*
- The remaining values as like *windy*, *drizzling*, *overcast*, and *clear*, among others, can be interpreted as mild weather conditions. Being the most usual, they do not much affect parking occupancy.

Thus, it seems that weather conditions have an effect on parking occupancy, especially unusual conditions that can be considered as bad weather conditions.

Another variable related to the weather is Summary2, which gives a daily and detailed description of the weather throughout the day. Because it is more detailed, it uses more tokens (words) to describe the weather. However, because it is modeled in a windowed approach, the model receives the same input for the twenty-four hours of the day, so the information is sparse and the model is not expected to obtain useful information for hourly predictions.

Figure 8.12 shows the occupancy forecasts when using only the Summary2 variable. As expected, the levels of occupancy change at the same frequency as the variable does, that is, when a new day starts. Nevertheless, because the occupancy levels change according to this information, so the Summary2 variable provides some useful information for the model.

The heatmap of the cosine distance presented in Figure 8.13 shows results similar to those from the Summary variable, as tokens for bad weather conditions are closer to each other (*flurries*, *cloudy*, *foggy* and *snow*, for example).

The visualizations from the dimensionality reduction methods presented in Figures 8.14 and 8.15 allow for the interpretation that there are clearly two different weather conditions, but they differ between PCA and tSNE. The tSNE seems to be more interpretable, as one of the groups is related to calm weather conditions while the other (more heterogeneous) has the tokens *foggy*, *snowy*, *rainy* and *windy* together with words like *evening*,

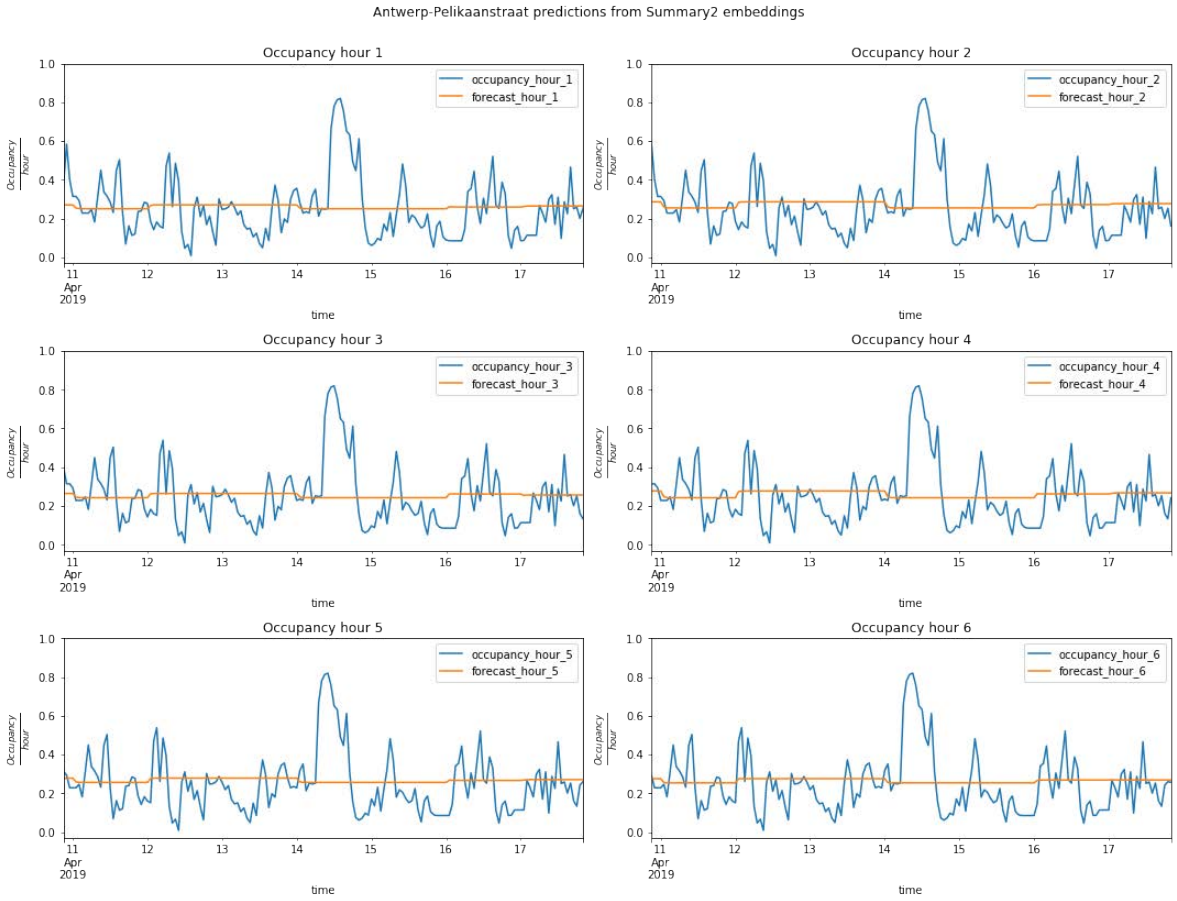


Figure 8.12: Pelikaanstraat forecasts using the Summary2 exogenous variable.

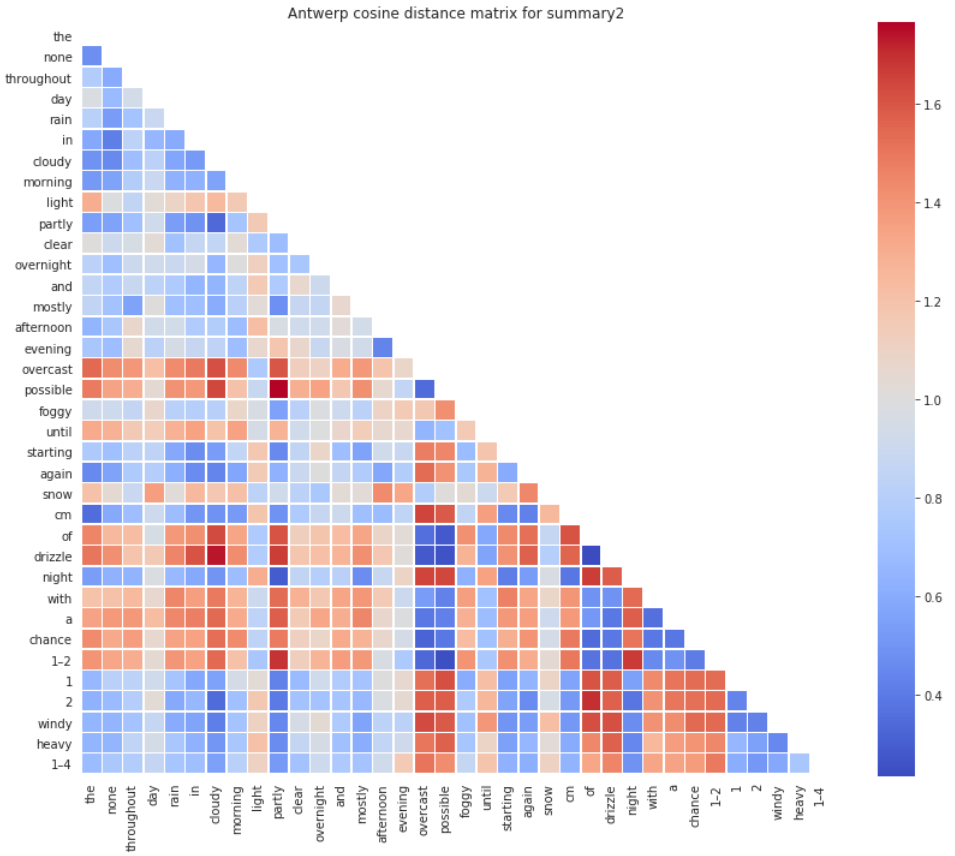


Figure 8.13: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the values of the variable Summary2.

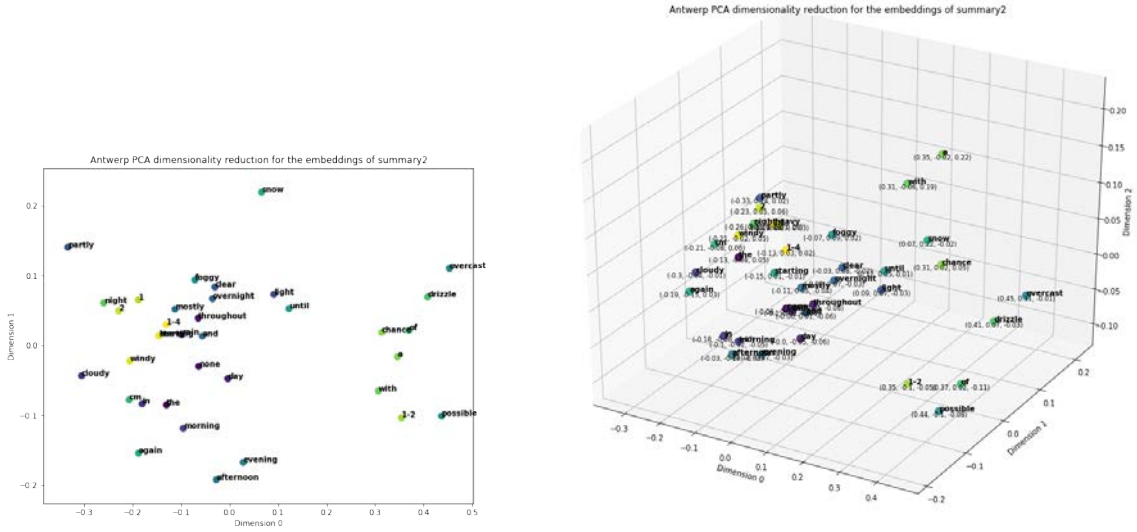


Figure 8.14: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Summary2 after PCA.

morning and *throughout*, which could indicate when the weather events will happen. Therefore, these tokens usually appear together.

The last categorical variable related to weather is Precipitation Type, and it has three possible values for each hour of the day: *snow*, *rain* or *none*. Its forecasts, presented in Figure 8.16, show that the model predicts lower parking occupancy with the *rain* value (days 13 and 17 of April in the afternoon/evening (shown in Table 8.1)), thus indicating that parking occupancy is affected by those weather conditions.

The cosine distances in Figure 8.17 show that occupancy is totally different on rainy and snowy days, while no precipitation has an effect that is similar to both values.

Although the 3D scatterplot of the PCA in Figure 8.18 seems to indicate that the values

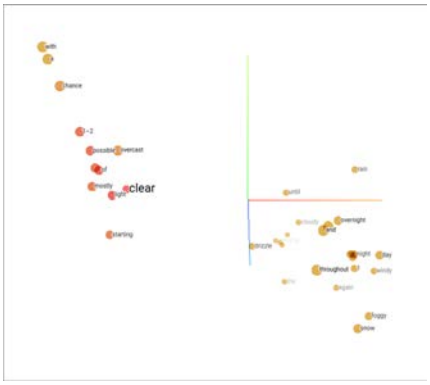


Figure 8.15: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Summary2 after tSNE.

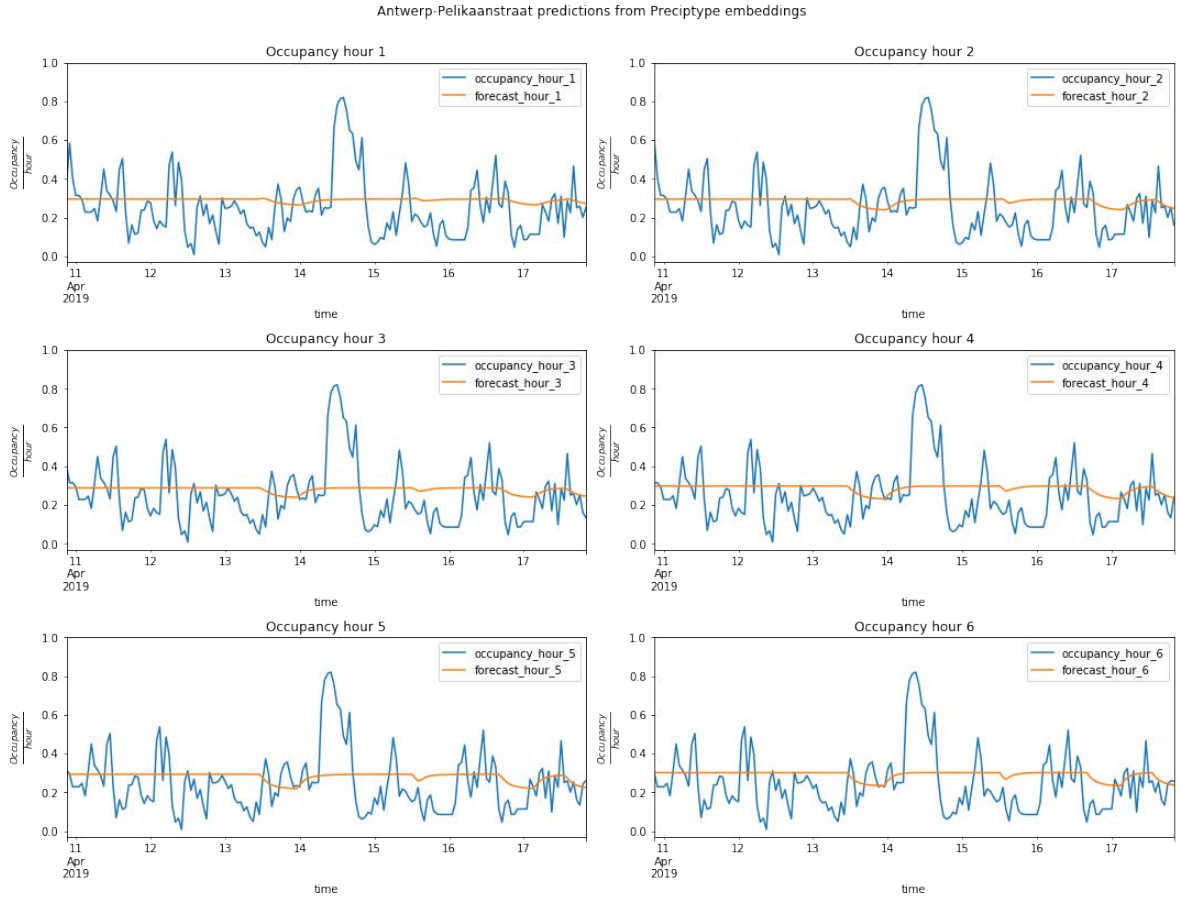


Figure 8.16: Pelikaanstraat forecasts using the Precipitation Type exogenous variable.

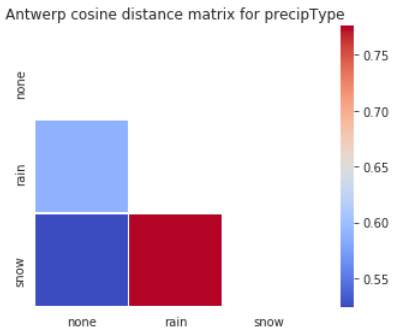


Figure 8.17: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the values of the variable Precipitation Type.

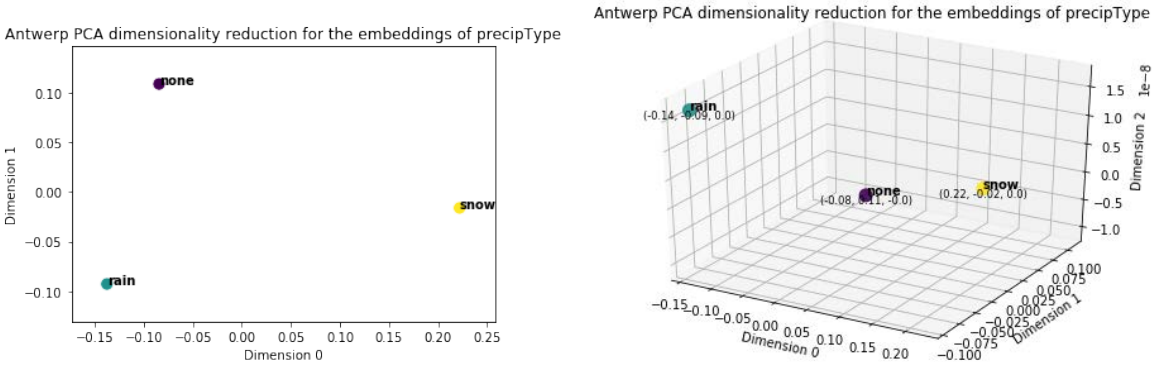


Figure 8.18: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Precipitation after PCA.

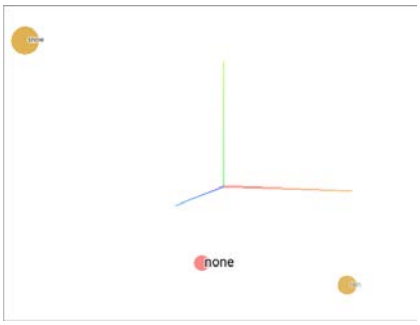


Figure 8.19: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Precipitation after tSNE.

none and *snow* are more similar to each other, careful observation of their values in each dimension reveals that the value *none* is closer to *rain* than to *snow*. This can be seen more easily in the tSNE visualization (Figure 8.19), as it shows that the Precipitation Type values *none* and *rain* are closer to the parking occupancy values than *snow* is, which can be seen in the 2D PCA visualization.

The models created using each of three categorical variables related to weather present overlapping forecasts, as the information is redundant. Of the three, the most promising is the variable Summary, as it has hourly frequency and includes information from Summary2 and Precipitation Type.

The next categorical variable is Season, whose values are *Winter*, *Spring*, *Summer* and *Autumn*. Although its frequency change is very small when compared to the scale of the problem, it is of interest to inspect how parking occupancies change throughout the year.

In order to visualize the forecasts, it is necessary to use the whole interval of data, which is approximately one year for the Pelikaanstraat sector. Figure 8.20 shows that the model change the values of its forecasts based on the season, although the levels of occupancy change from one step to the other. The model identifies that Spring is the

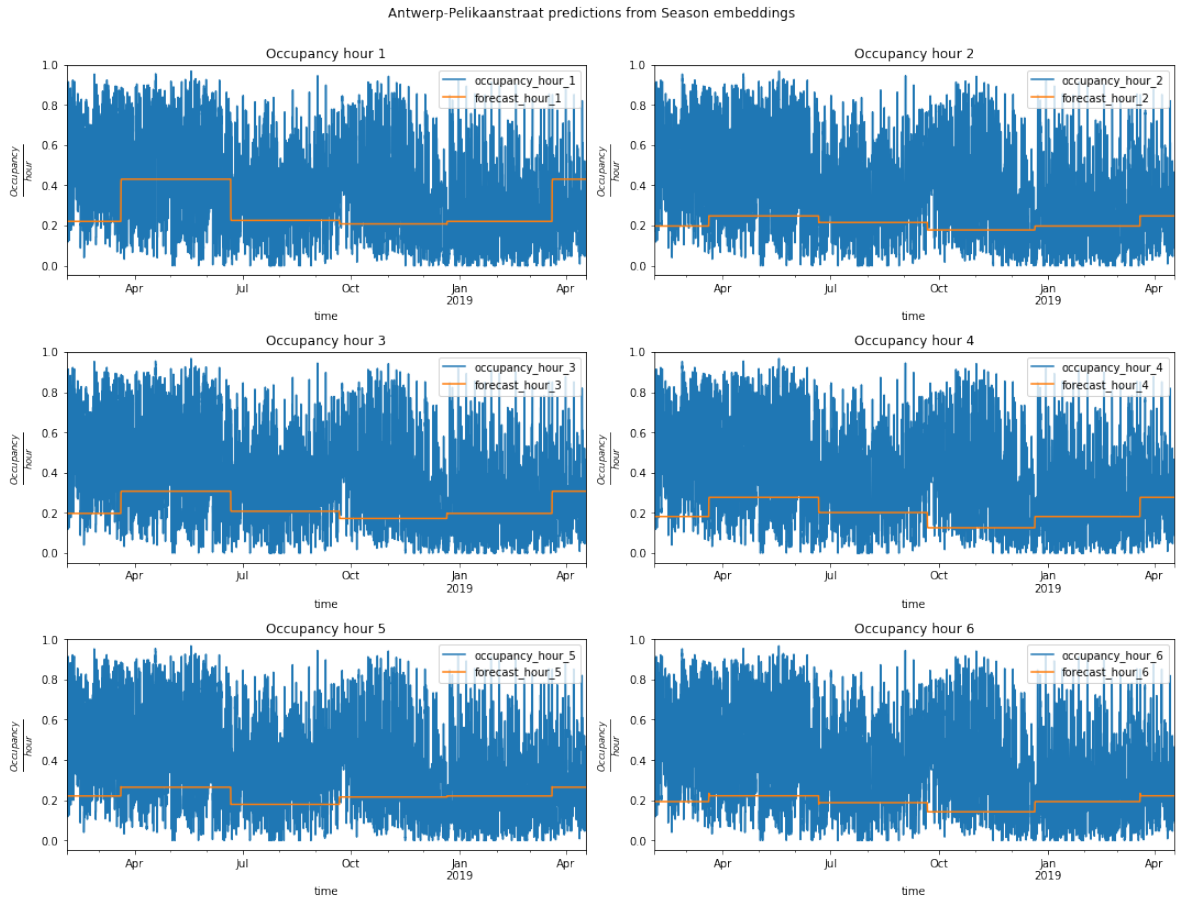


Figure 8.20: Pelikaanstraat forecasts using the exogenous variable Season.

season with the highest parking occupancy and Winter the one with the lowest values.

The heatmap of the cosine distances is shown in Figure 8.21. It can be observed that the occupancy patterns are more similar throughout consecutive seasons.

Figure 8.22 presents the scatterplots after reducing the dimensionality of the embeddings for the categorical variable Season. As observed in the heatmap, consecutive seasons are closer to each other, especially Autumn and Winter. Another fact that can be observed is that Spring and Autumn are dissimilar. The tSNE visualization (Figure 8.23) shows that each season is different from the others. It must notably be taken into account that occupancy patterns can change over time, which could be the reason why Autumn–Spring and Winter–Summer show orthogonal positions.

Another categorical variable with a small frequency change rate is Month. However, like Season, it could offer interesting insights into parking occupancy patterns. Figure 8.24 shows the forecasts of the model trained only with the variable Month in order to forecast the parking occupancy. What the model has learnt seems to be the same as it did with the variable Season – but with finer granularity – as the forecasts for Month are similar to those for Season in that they show more parking occupancy levels in April

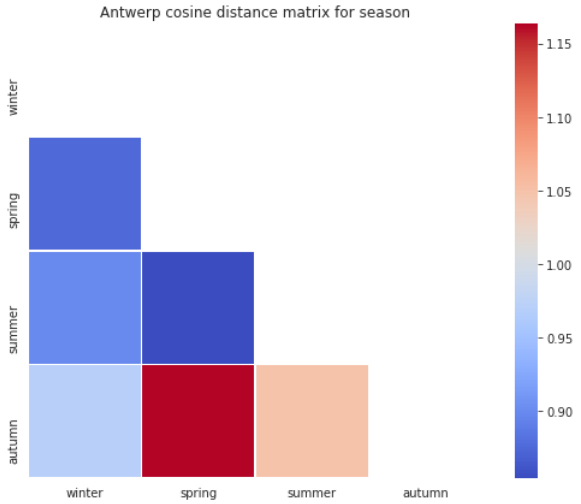


Figure 8.21: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the values of the variable Season.

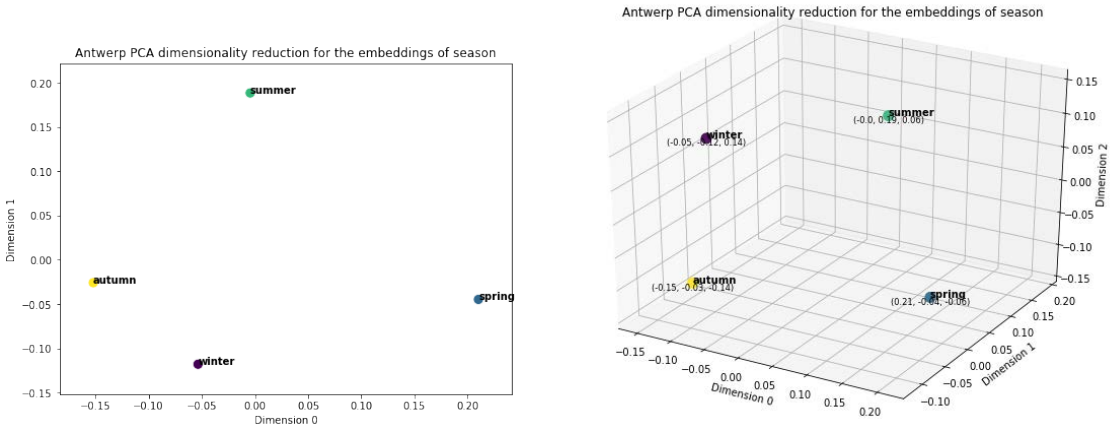


Figure 8.22: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Season after PCA.

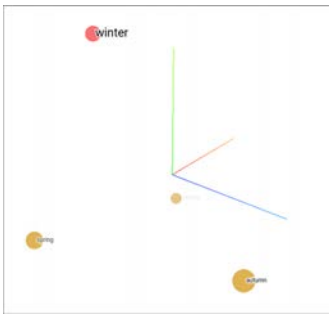


Figure 8.23: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Season after tSNE.

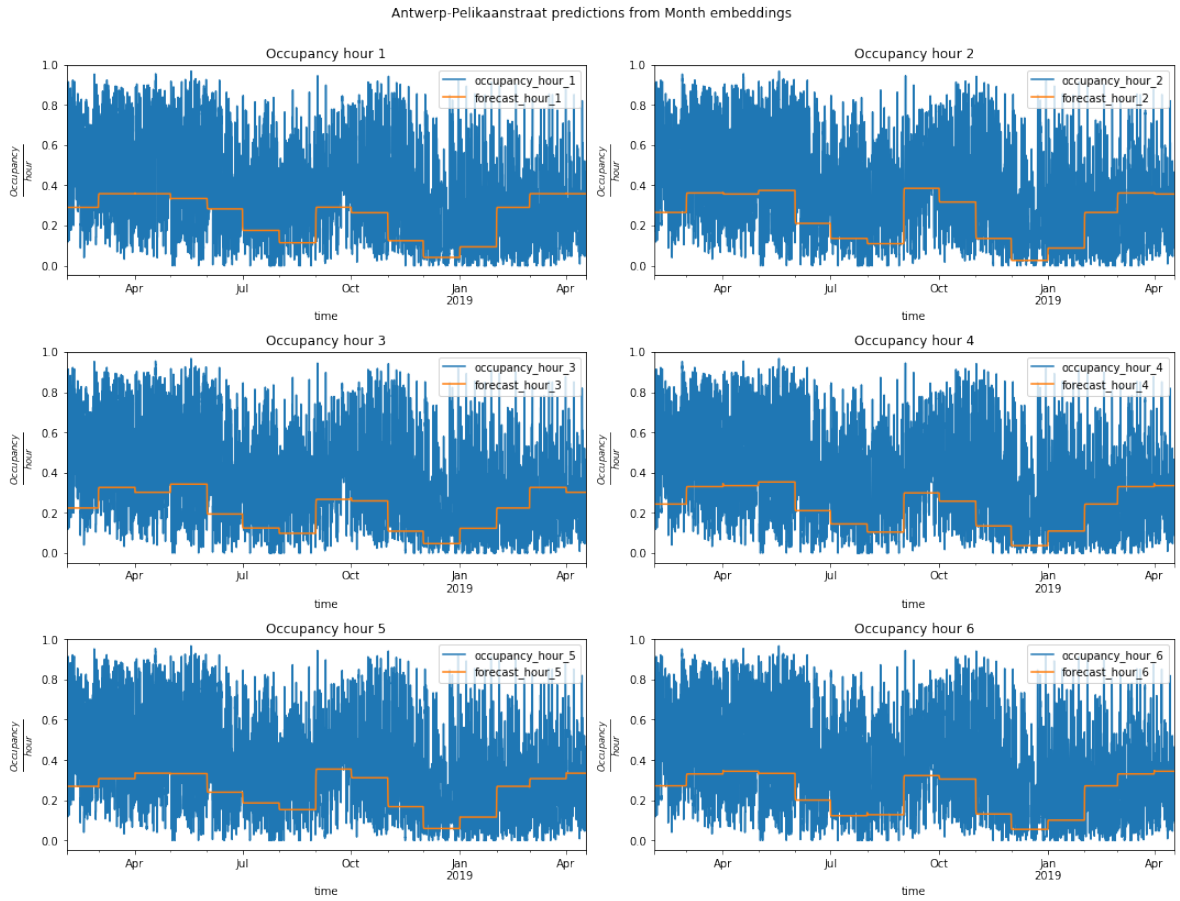


Figure 8.24: Pelikaanstraat forecasts using the Month exogenous variable.

(Spring) and fewer in December (Winter). Therefore, it is possible to just use Month instead of both variables.

The heatmap of the cosine distances for the embeddings of the Month variable is shown in Figure 8.25. It can be observed that the occupancy patterns are more similar throughout consecutive months, something that was already observed with the variable Season. Moreover, the influence of vacations can be observed, as months like July and December show a small distance between each other due to having fewer parking occupancies.

Figures 8.26 and 8.27 present the scatterplots from the dimension reduction algorithms. All three visualizations show clearly that two different clusters correspond to Summer/Winter months versus Spring/Autumn. This could be interpreted as the influence of holidays and vacations on the parking occupancies.

Daily information can also be of interest, and it is represented with the categorical variables Working Day, Holiday and Day of the Week.

Forecasts for the model trained with the categorical variable Working Day are presented in Figure 8.28. The model differentiates the occupancy between April 13 to 15 (weekend), and forecasts the same level of occupancy throughout the weekdays.

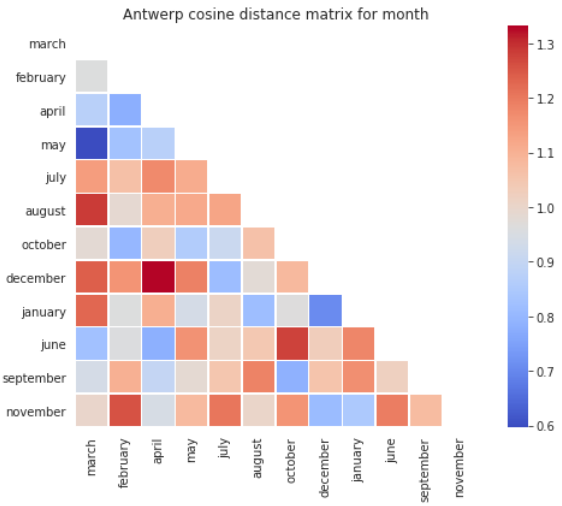


Figure 8.25: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the values of the variable Month.

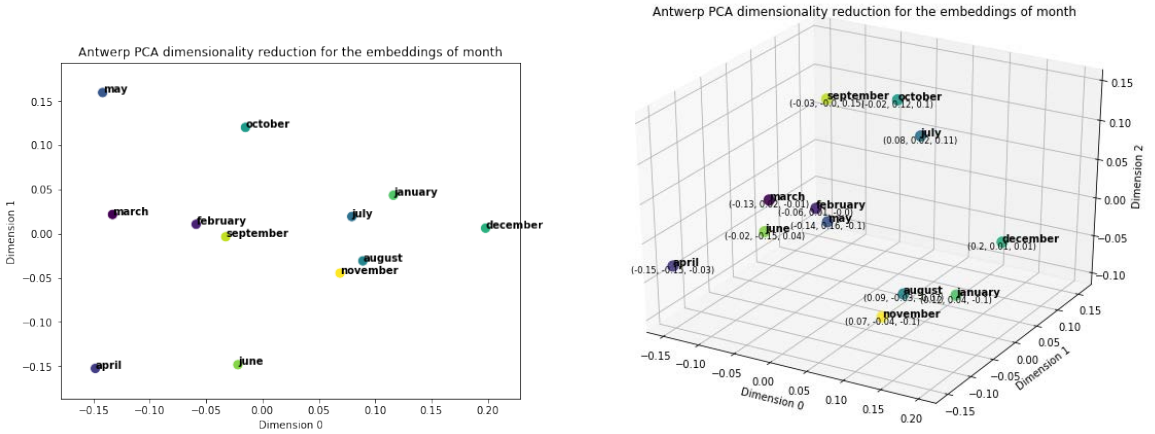


Figure 8.26: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Month after PCA.

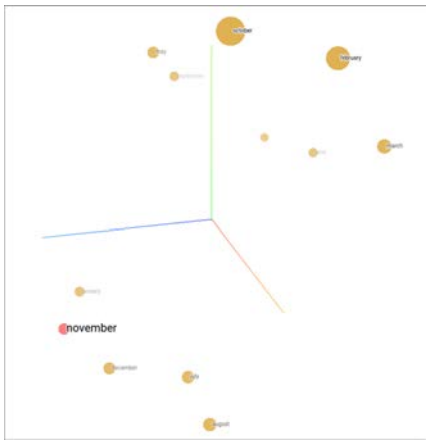


Figure 8.27: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Month after tSNE.

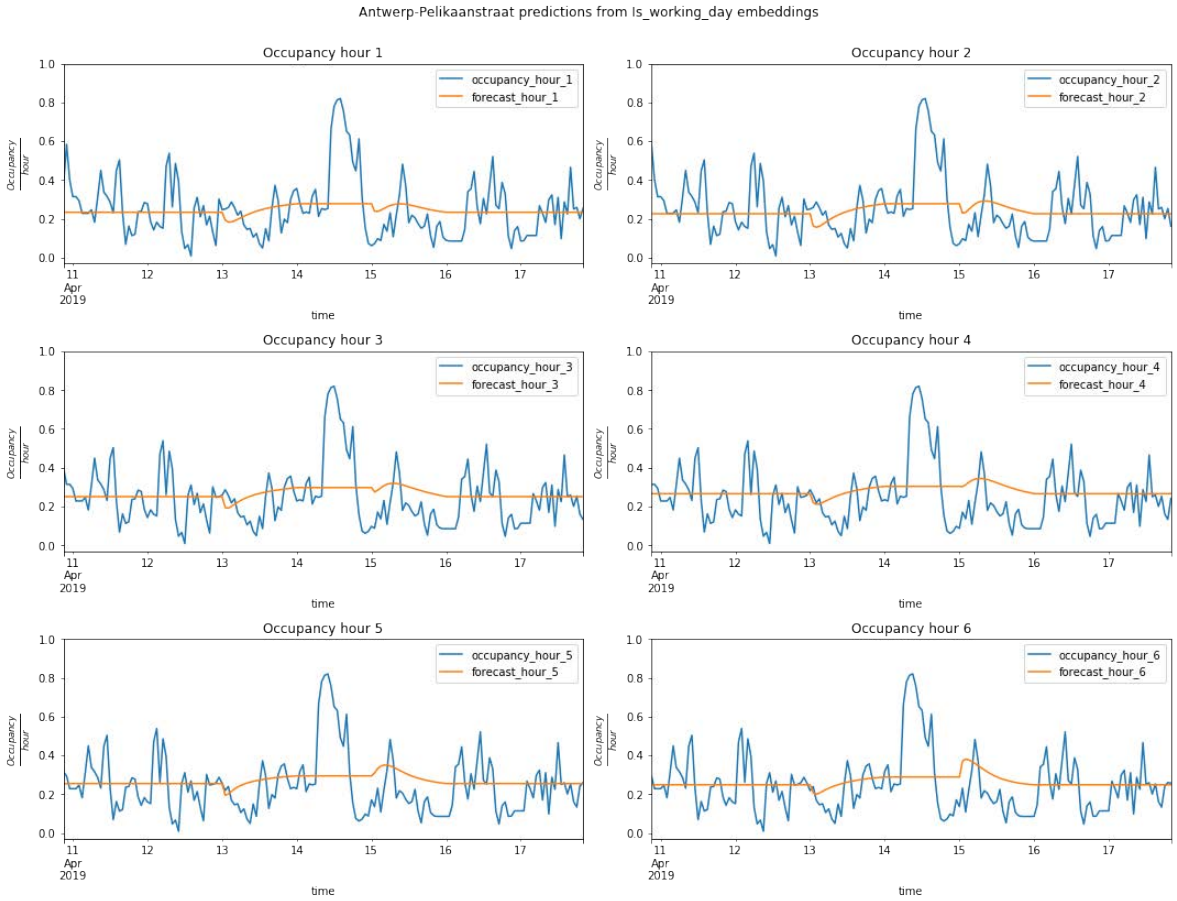


Figure 8.28: Pelikaanstraat forecasts using the Working Day information.

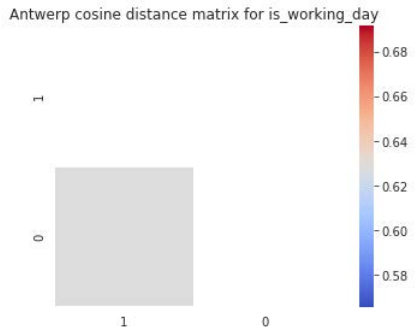


Figure 8.29: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the values of the variable Working Day.

Figure 8.29 shows the heatmap of the cosine distance between the two possible values of the variable Working Day. It shows clear orthogonality.

Dimensional reduction methods cannot offer relevant information, as the categorical variable's number of values is just two, and the heatmap already shows that the parking occupancies are different between weekdays and weekends.

The categorical variable Holiday is of interest, as it could provide the model useful calendar information if it detects that holidays behave differently than workdays. Figure 8.30 shows how the model predicts higher occupancy values for the days of the Easter Holiday, which can be counterintuitive because holidays are usually associated with low parking occupancies. However, this depends on the users of the parking sector. For the non-holiday days, the forecasts are always the mean of the series.

Figure 8.31 shows the heatmap of the cosine distance between the two possible values of the variable Holiday, with clear neutrality (distance one) being indicated between the values.

Because of the low number of values, the dimension reduction methods do not offer any useful information.

The last categorical variable is Day of the Week. It is expected that the model learns more specific information than the variable Working Day, as Day of the Week can also provide this information.

Figure 8.32 shows the forecasts when using only the categorical variable Day of the Week. Surprisingly, of all the categorical variables, this is the one that provides forecasts with the lowest error rate while making clear distinctions between the patterns for each day of the week, although the variable has a daily frequency change. The monthly visualization offered in Figure 8.33 clearly shows that the model differentiates the occupancy according to the days of the week. This implies that parking occupancy is heavily influenced by the day of the week. Furthermore, this could provide information that is not available when using only the endogenous variable in sequences of twenty-four past values.

The heatmap from the cosine distance matrix is shown in Figure 8.34. Observe that

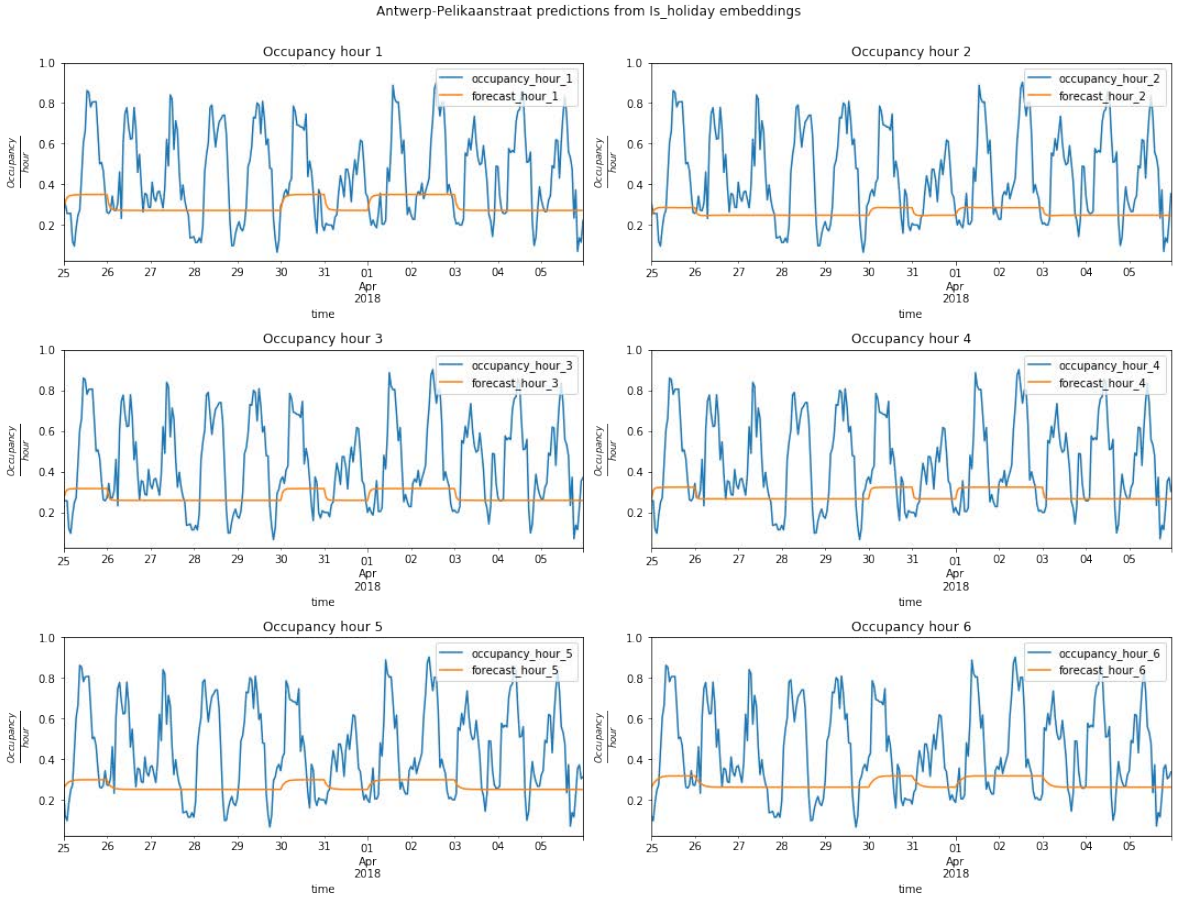


Figure 8.30: Pelikaanstraat forecasts using the categorical variable Holiday.

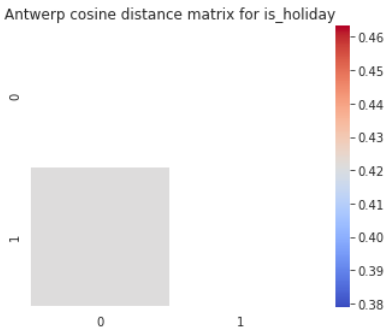


Figure 8.31: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the values of the variable Holiday.

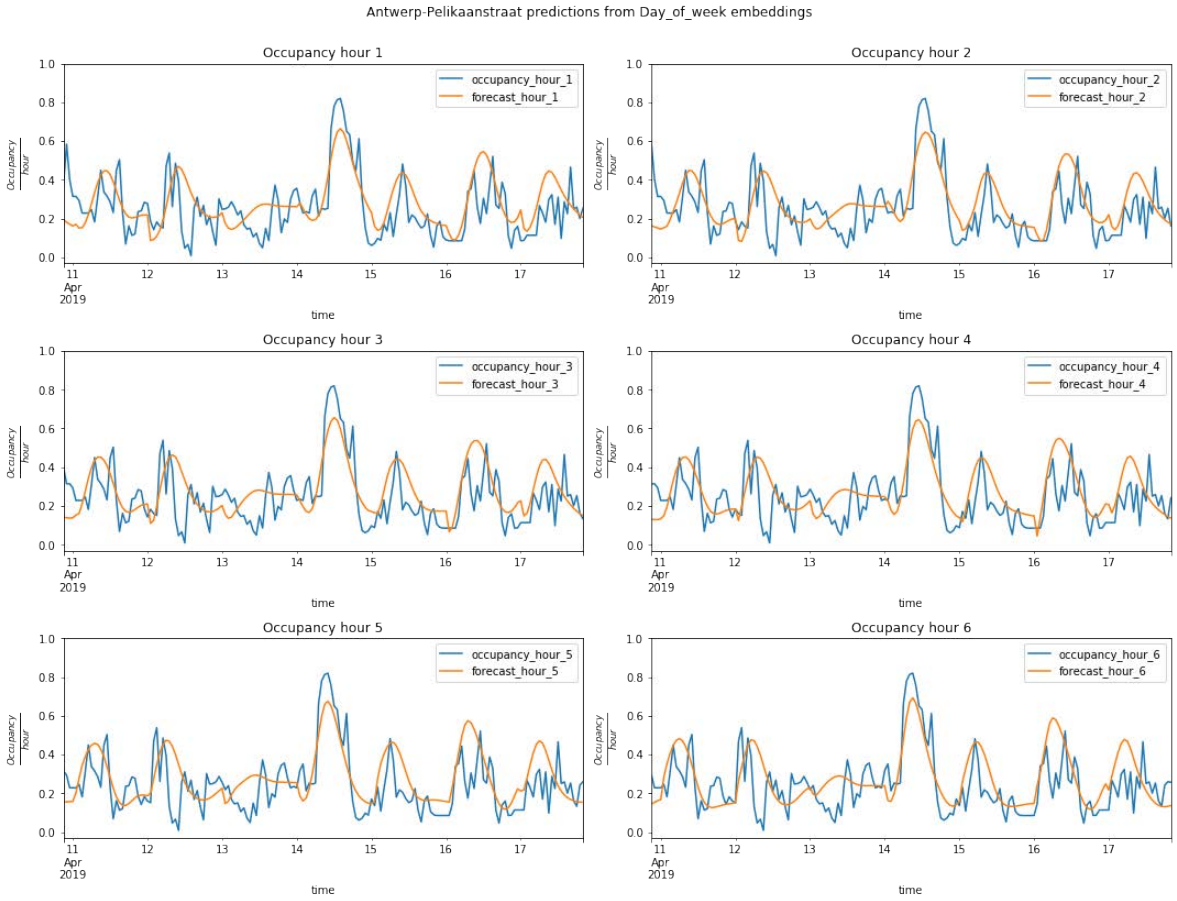


Figure 8.32: Pelikaanstraat forecasts using the categorical variable Day of the Week.

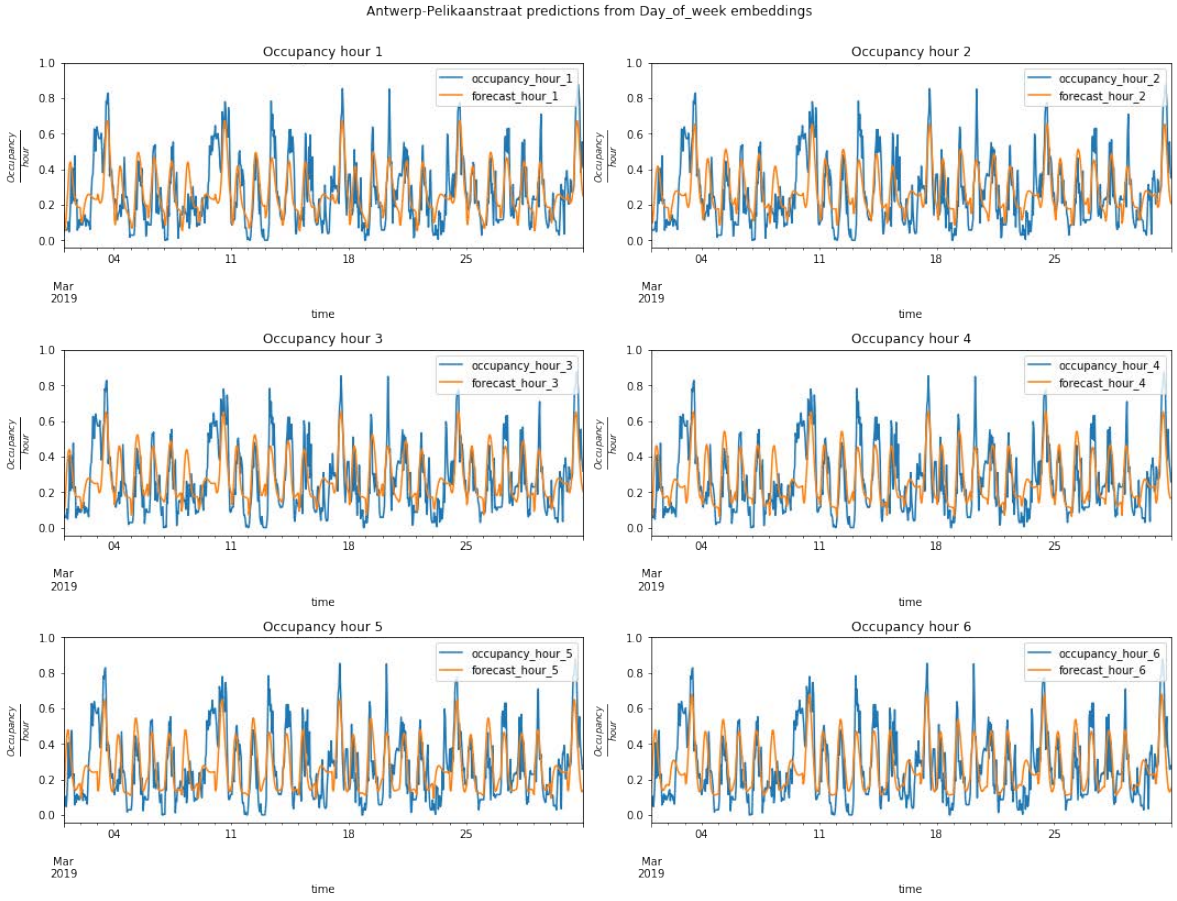


Figure 8.33: Pelikaanstraat forecasts through March 2019 using the categorical variable Day of the Week.

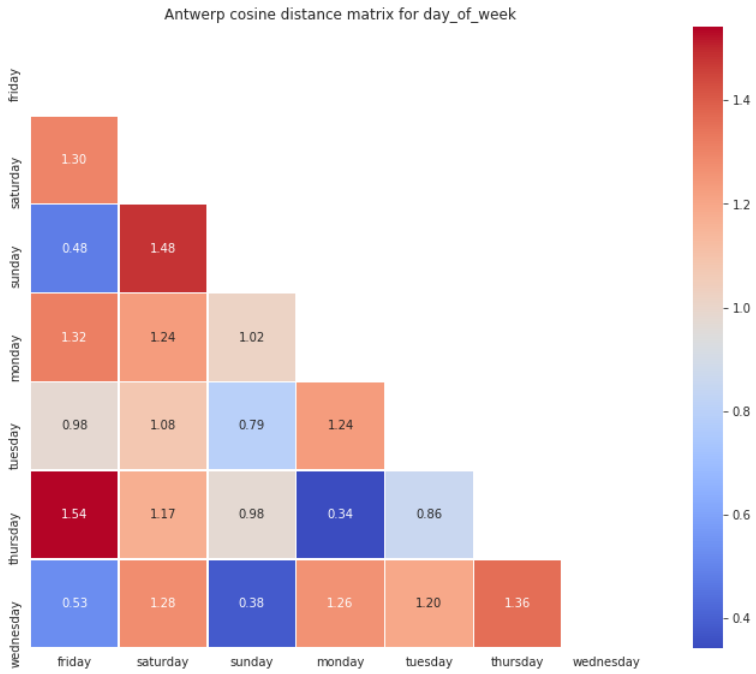


Figure 8.34: Antwerp Pelikaanstraat sector heatmap of the cosine distance between the days of the week.

some days of the week are very similar, namely: Wednesday, Friday and Sunday; and Monday and Thursday.

The scatterplots in Figures 8.35 and 8.36 provide a better visualization of the different parking behaviours in the Pelikaanstraat sector, based on the values of the variable Day of the Week. PCA shows that the days are organized into four groups:

1. Thursday and Monday
2. Friday, Wednesday and Sunday.
3. Tuesday.
4. Saturday

While tSNE presents four groups that differ a little from the PCA visualization:

1. Sunday and Wednesday.
2. Tuesday and Friday.
3. Thursday and Monday.
4. Saturday.

Day of the Week clearly offers information that is the same as or better than that of Working Day. Therefore, the latter can be replaced by the former.

From the exogenous categorical variables in Antwerp, the following information has been obtained:

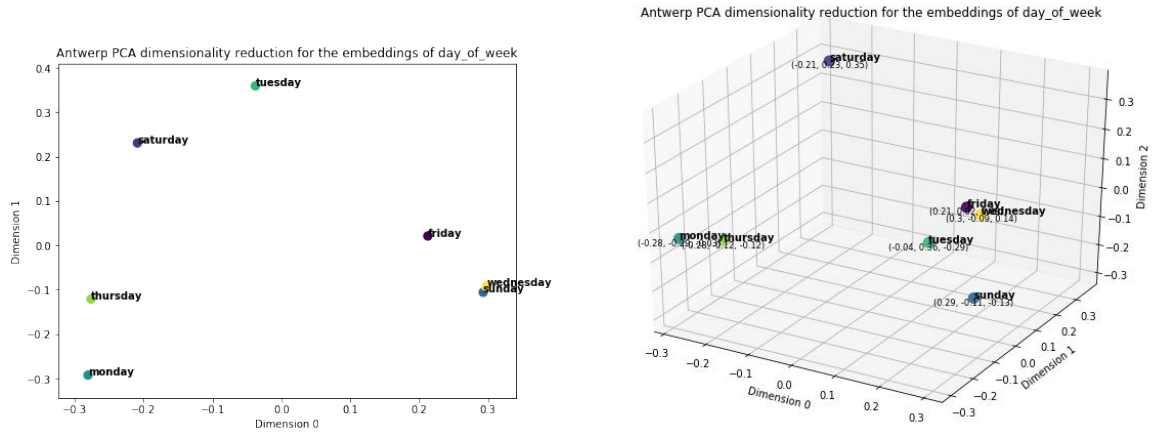


Figure 8.35: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Day of the Week after PCA.

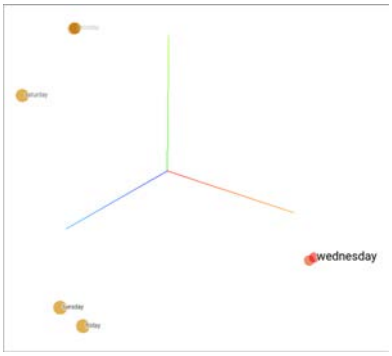


Figure 8.36: Antwerp Pelikaanstraat scatterplots for the embeddings of the variable Day of the Week after tSNE.

- Day of the Week is the one that contributes the most to model parking occupancy.
- There are redundant variables:
  - Working Day can be replaced by Day of the Week.
  - Season can be replaced by Month.
  - Summary2 can be replaced by Summary, as it offers hourly descriptions of the weather conditions, which is in contrast to the daily descriptions of Summary2.
- The Holidays variable seems to have an effect on the behaviour of the parking occupancies.

For the next scenarios, the categorical variables for the analysis are:

- Summary.
- PrecipType
- Day of the Week.
- Month.
- Holiday.

Wattens

Figure 8.37 presents the occupancy forecasts for the next six hours from the model created with the categorical variable Summary for the Wattens Outside sector. The visualizations show that the forecasted levels of occupancy are influenced by the weather information, but it is hard to quantify the importance of the effect. Table 8.2 shows the information used as input for the day 15 May 2019.

Figure 8.38 shows the distance matrix obtained when the cosine distance is applied to the embeddings of the Summary variable as a heatmap for the Wattens Outside sector. It gives the impression that the occupancy is conditioned by the weather, as bad weather conditions have small distances.

Figures 8.39 and 8.40 show the scatterplots after applying the dimensionality reduction methods PCA and tSNE. It seems that PCA dimensions can be related to weather conditions because *windy*, *snow*, *heavy*, *flurries* and *drizzle* are positioned at similar levels in the third axis (PCA with three dimensions). tSNE provides a cluster for *cloudy*, *rain*, *partly* and *heavy* (all weather conditions related to rainy days) and another for *windy*, *snow*, *flurries* and *drizzle* (weather conditions related to snowy days) and the other values are in between, forming little clusters of pair elements like *none* and *clear* (neutral weather conditions in relation to occupancy).

The other categorical weather variable is Precipitation Type, whose forecasts are presented in Figure 8.41. Similarly to what was observed in Antwerp, the model tries to

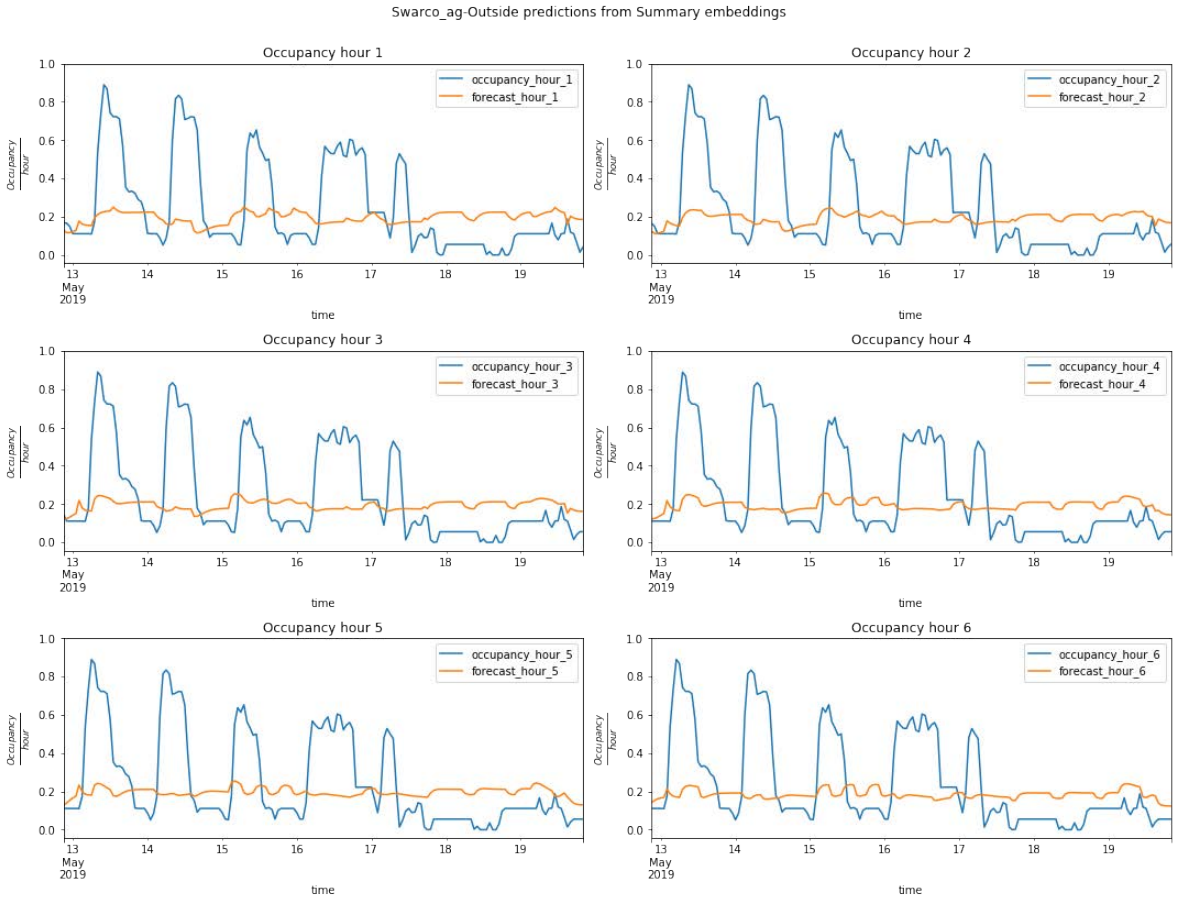


Figure 8.37: Wattens forecasts with the categorical variable Summary.

time	precipType	summary
2019-05-15 00:00:00	rain	Foggy
2019-05-15 01:00:00	rain	Foggy
2019-05-15 02:00:00	rain	Overcast
2019-05-15 03:00:00	rain	Overcast
2019-05-15 04:00:00	rain	Overcast
2019-05-15 05:00:00	rain	Overcast
2019-05-15 06:00:00	rain	Mostly Cloudy
2019-05-15 07:00:00	rain	Mostly Cloudy
2019-05-15 08:00:00	rain	Mostly Cloudy
2019-05-15 09:00:00	rain	Mostly Cloudy
2019-05-15 10:00:00	rain	Overcast
2019-05-15 11:00:00	rain	Overcast
2019-05-15 12:00:00	rain	Overcast
2019-05-15 13:00:00	rain	Overcast
2019-05-15 14:00:00	rain	Mostly Cloudy
2019-05-15 15:00:00	rain	Mostly Cloudy
2019-05-15 16:00:00	rain	Mostly Cloudy
2019-05-15 17:00:00	rain	Mostly Cloudy
2019-05-15 18:00:00	rain	Overcast
2019-05-15 19:00:00	rain	Overcast
2019-05-15 20:00:00	rain	Overcast
2019-05-15 21:00:00	rain	Overcast
2019-05-15 22:00:00	none	Mostly Cloudy
2019-05-15 23:00:00	none	Mostly Cloudy

Table 8.2: Wattens Outside sector weather information on 15 May 2019.

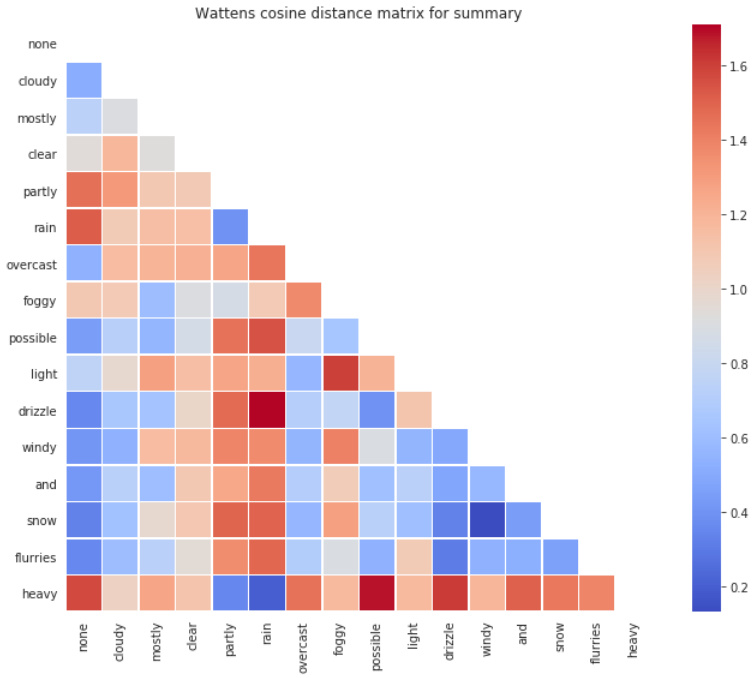


Figure 8.38: Wattens Outside sector heatmap of the cosine distance between the values of the variable Summary.

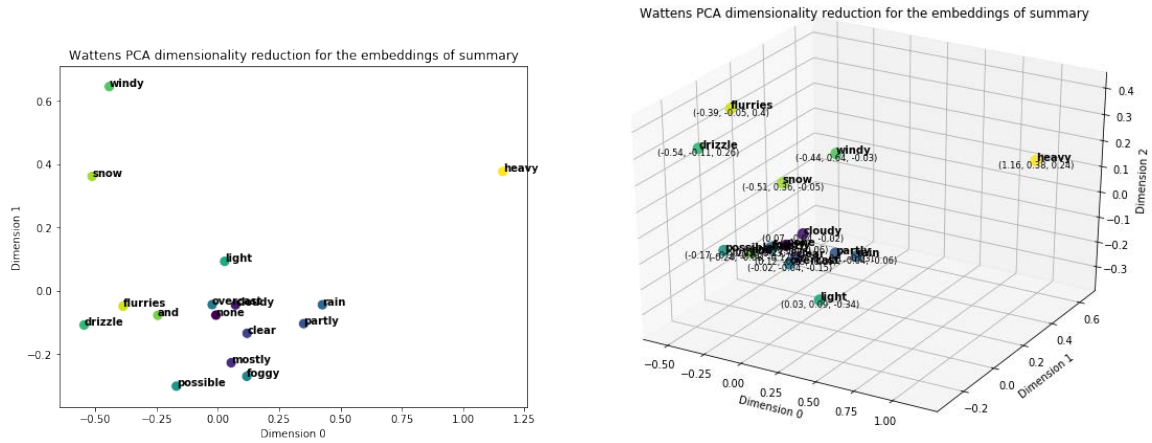


Figure 8.39: Wattens Outside scatterplots for the embeddings of the variable Summary after PCA.

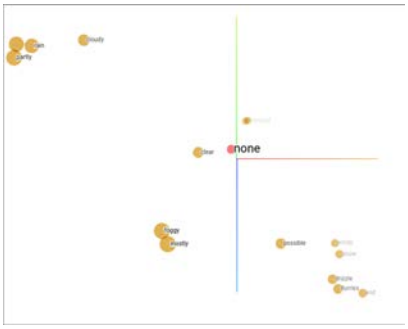


Figure 8.40: Wattens Outside scatterplots for the embeddings of the variable Day of the Week after tSNE.

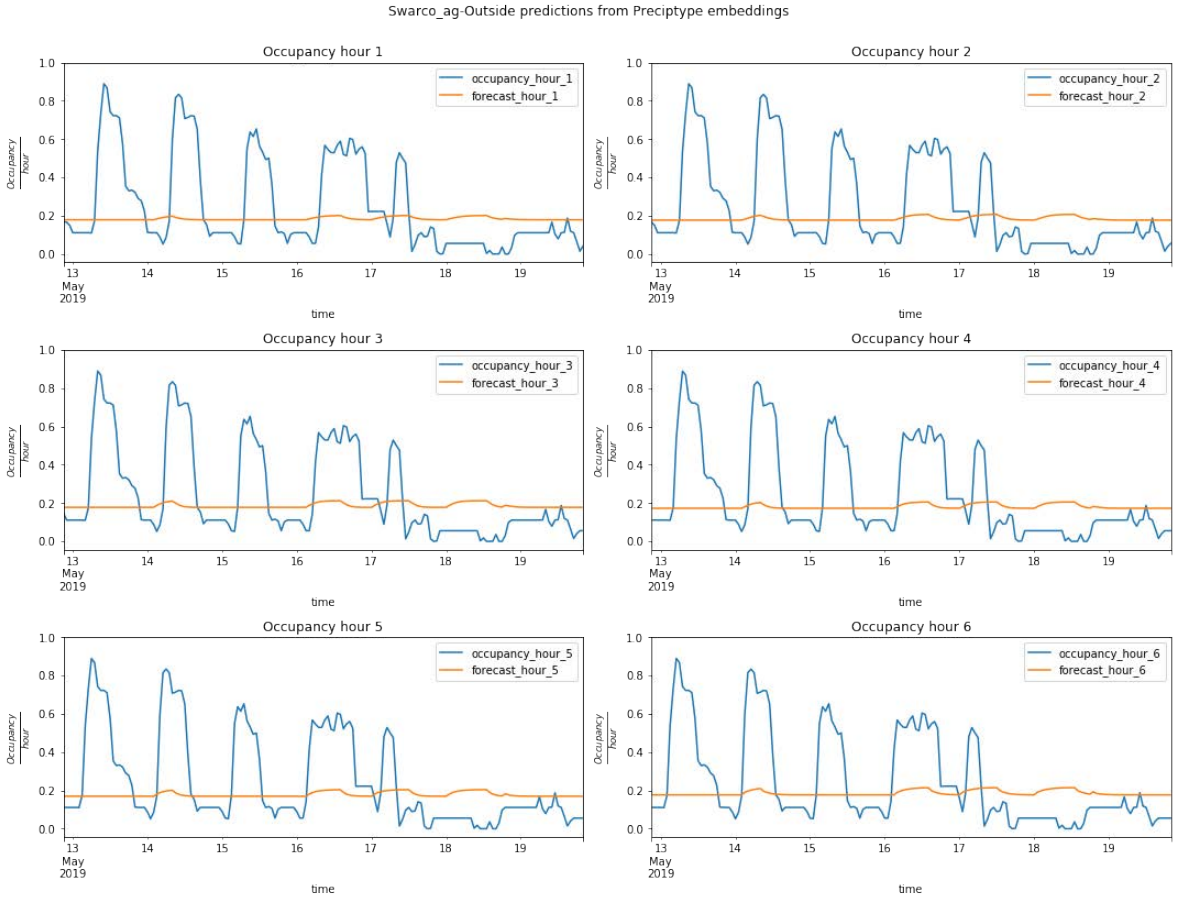


Figure 8.41: Wattens Outside sector forecasts with the categorical variable Precipitation Type.

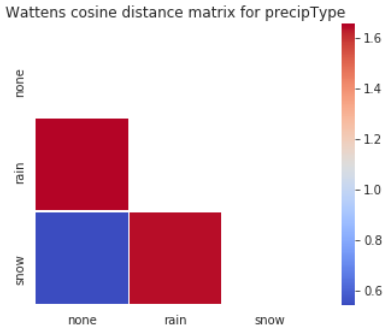


Figure 8.42: Wattens Outside sector heatmap of the cosine distance between the values of the variable Precipitation Type.

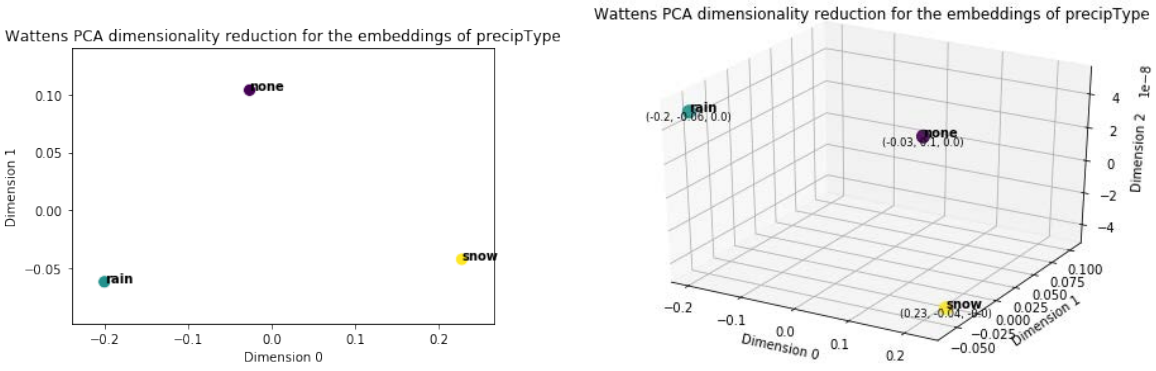


Figure 8.43: Wattens Outside scatterplots for the embeddings of the variable Precipitation Type after PCA.

forecast the parking occupancy based on the values of Precipitation, which are *none*, *rain* and *snow*. However, it lacks expressiveness power to produce good forecasts.

The cosine distance matrix (Figure 8.42) clearly shows that occupancy behaves differently on rainy days.

The scatterplots in Figure 8.43 show the distribution of the embedding values when a dimensionality reduction method is applied. PCA shows that the three values have different relationships with the parking occupancy. No visualization is offered for the tSNE method because it is not able to converge, maybe because of the low number of values.

Forecast visualizations for the different time steps of the model trained with the exogenous variable Month are provided in Figure 8.44. It shows that the season with the highest levels of occupancy is Autumn, as it coincides with the months between September and November. December shows the lowest occupancy levels, perhaps due to the holidays. Remember that the Wattens sectors are private.

The heatmap of the cosine distance matrix is shown in Figure 8.45. It shows that February behaves differently than the rest of the months and that the December occupancy

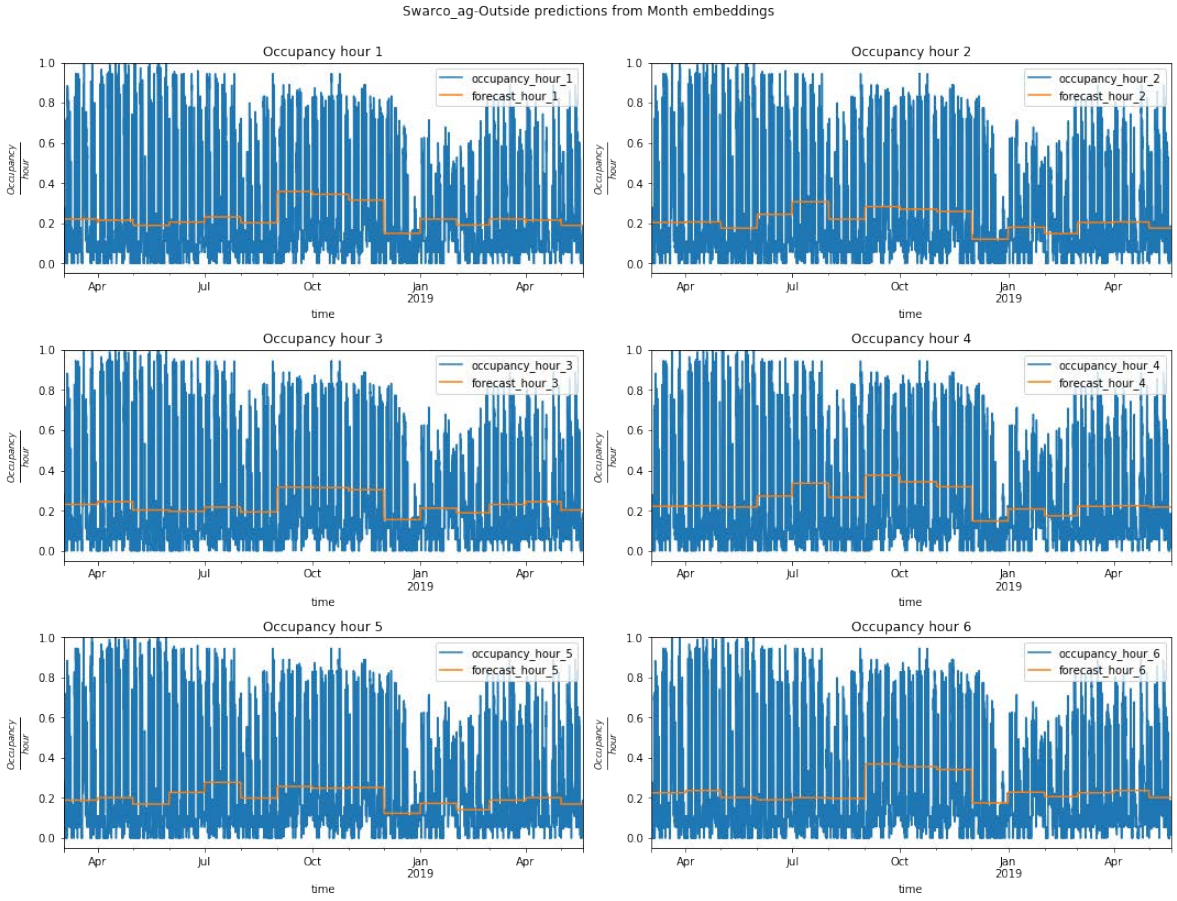


Figure 8.44: Wattens Outside sector forecasts from a model trained with the categorical variable Month.

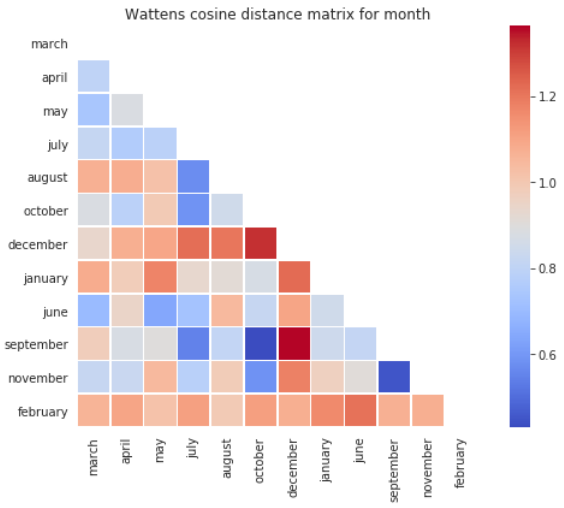


Figure 8.45: Wattens Outside sector heatmap of the cosine distance between the values of the variable Month.

levels are clearly different than those of September and October, which have the highest occupancy levels.

The results from applying PCA and tSNE to the embeddings of Month are shown in Figures 8.46 and 8.47. PCA shows that there is a clear linear pattern that distinguishes November, October and September from the rest of the months, especially from December through the first axis. tSNE brings together October and September; February and December; and January and March.

The forecasts for the model trained with the categorical variable Holiday are presented in Figure 8.48. The visualizations show that the model predicts a mean value for non-

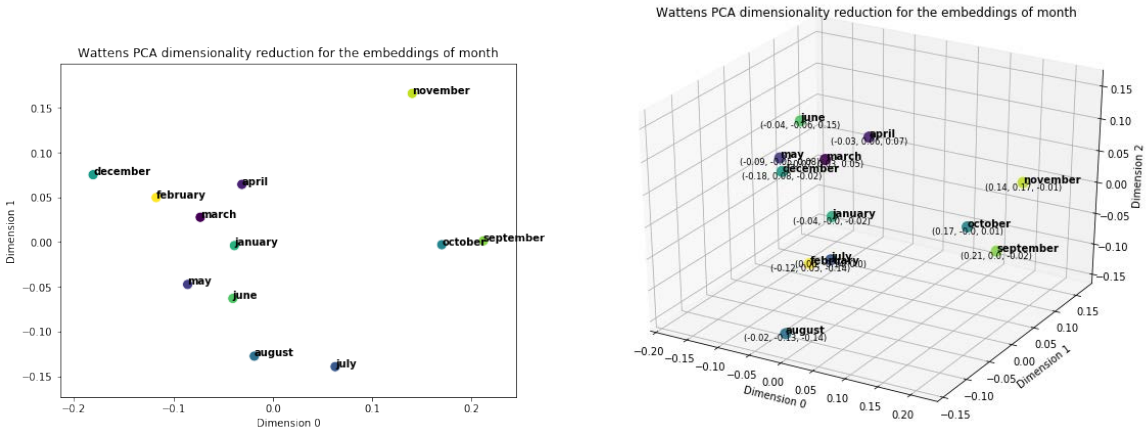


Figure 8.46: Wattens Outside scatterplots for the embeddings of the variable Month after PCA.

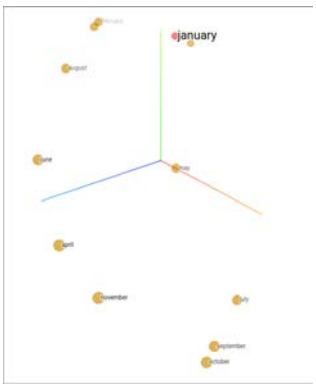


Figure 8.47: Wattens Outside scatterplots for the embeddings of the variable Month after tSNE.

holidays (21, 22 and 23 of December) and a different occupancy value for holidays (24, 25 and 26 of December). Interestingly, the model provides more detailed forecasts for days that transition between holiday and non-holiday. The reason is the sparsity of the variable, as it usually has the same value (non-holiday). However, the visualizations suggest that holidays in the Wattens sectors have an effect on the parking occupancies.

Because of the lack of values in the variable (which is binary), the heatmap and the dimensionality reduction plots do not offer any information.

Lastly, the visualizations of the forecasts for the categorical variable Day of the Week are presented in Figure 8.49. Similarly to the Antwerp scenario, this is the categorical variable that offers better forecasts, as the model has learnt the different patterns of occupancy during the week. What is more, it is impressive how it is able to learn that the occupancy on Fridays is different from the other days due to the company in that parking sector closing earlier that day.

The heatmap of the cosine distance matrix from the Day of the Week is shown in Figure 8.50. Monday and Wednesday are similar to each other in terms of parking occupancy, as well as Tuesday and Thursday.

The visualizations in Figures 8.51 and 8.52 show how the values of the embeddings are positioned after applying PCA and tSNE. PCA visualizations show that Tuesday and Thursday are similar while Monday and Wednesday (and maybe Sunday) are also similar to each other, as well as Friday and Saturday. tSNE offers a slightly different version in which the groups are Friday, Saturday and Sunday (weekends), Monday and Wednesday and then, Tuesday and Thursday. Visually, it is clear that weekends are different from weekdays, but it is not evident why Monday and Wednesday / Tuesday and Thursday are so similar to each other and different from the rest, as those days have the same office hours.

The categorical exogenous variables for the Wattens Outside sector prove to be useful for correcting the forecast of parking occupancy on holidays and for taking into account

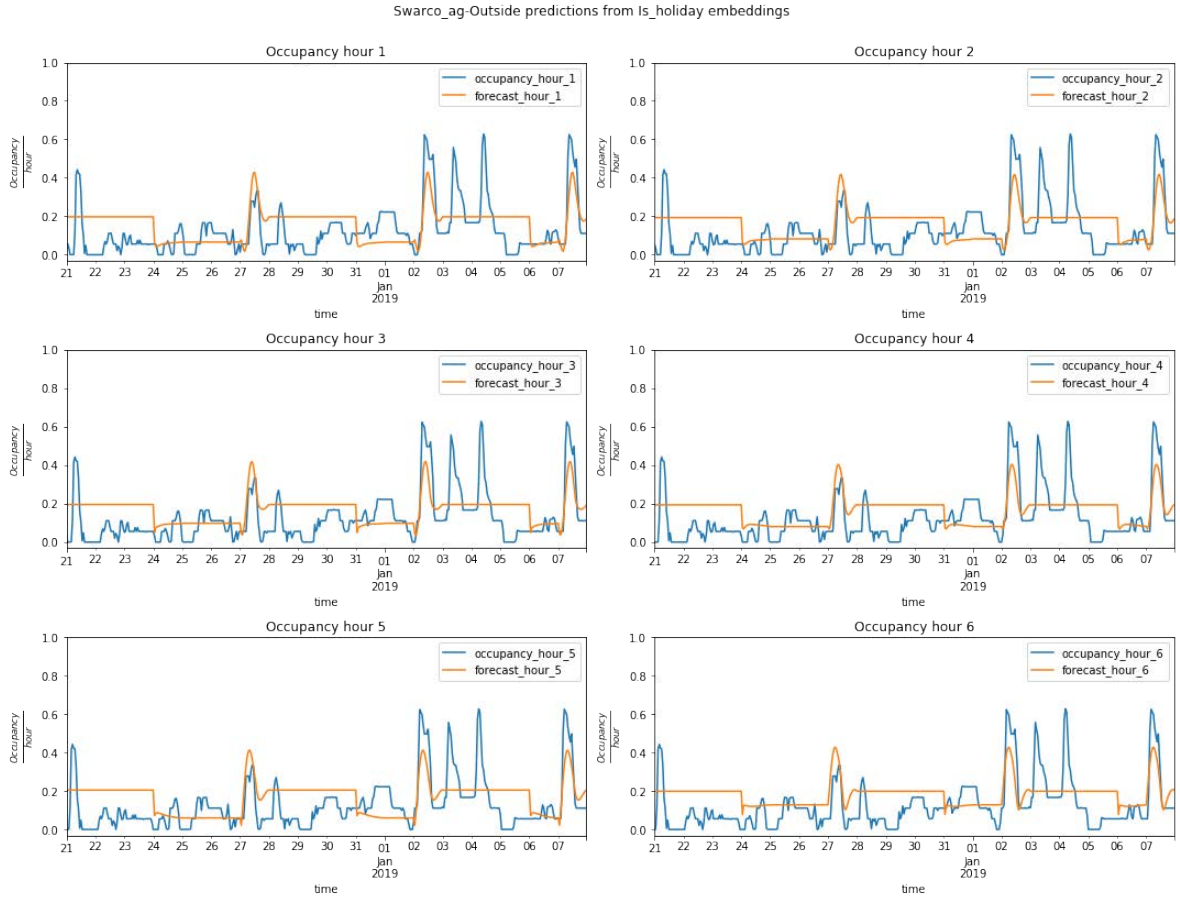


Figure 8.48: Wattens Outside sector forecasts from the model trained with the categorical variable Holiday.

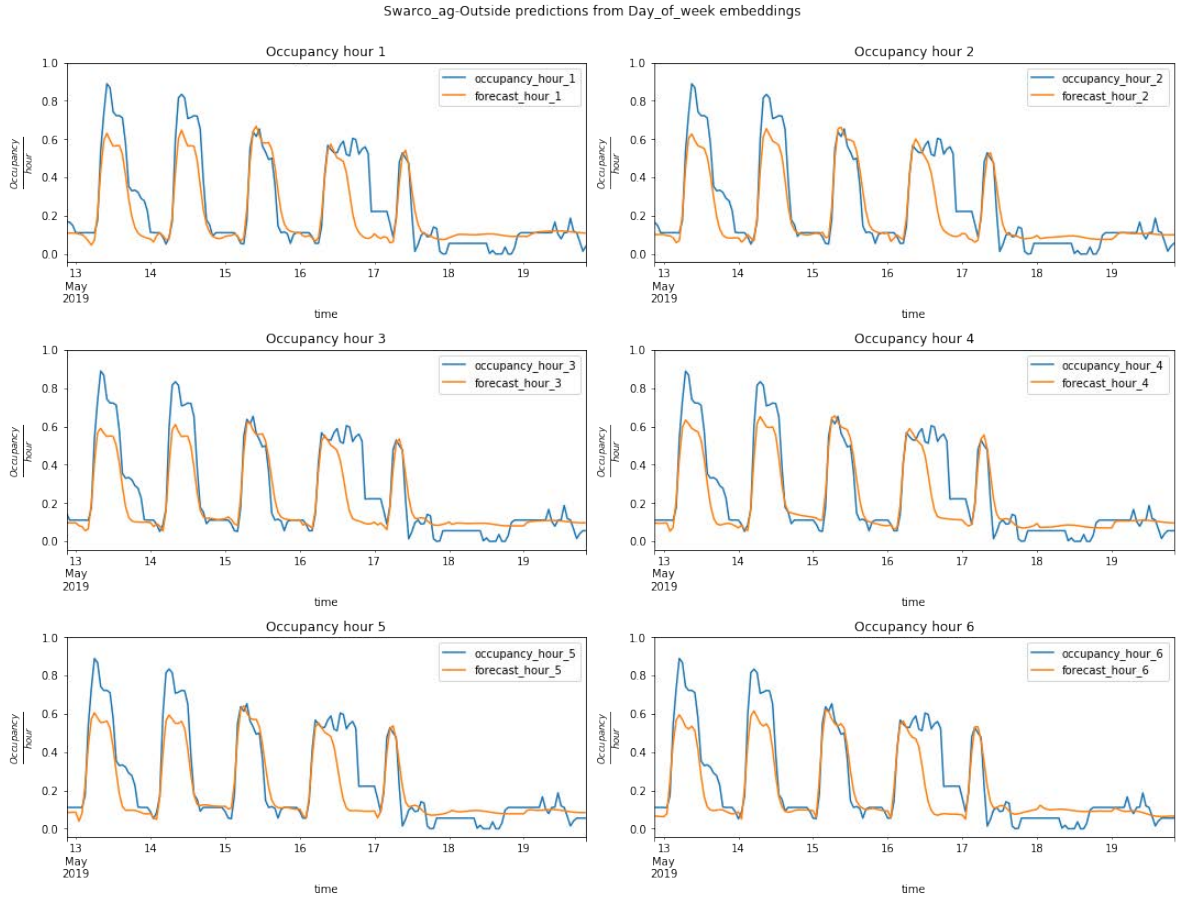


Figure 8.49: Wattens Outside sector forecasts from a model trained with the categorical variable Day of Week.

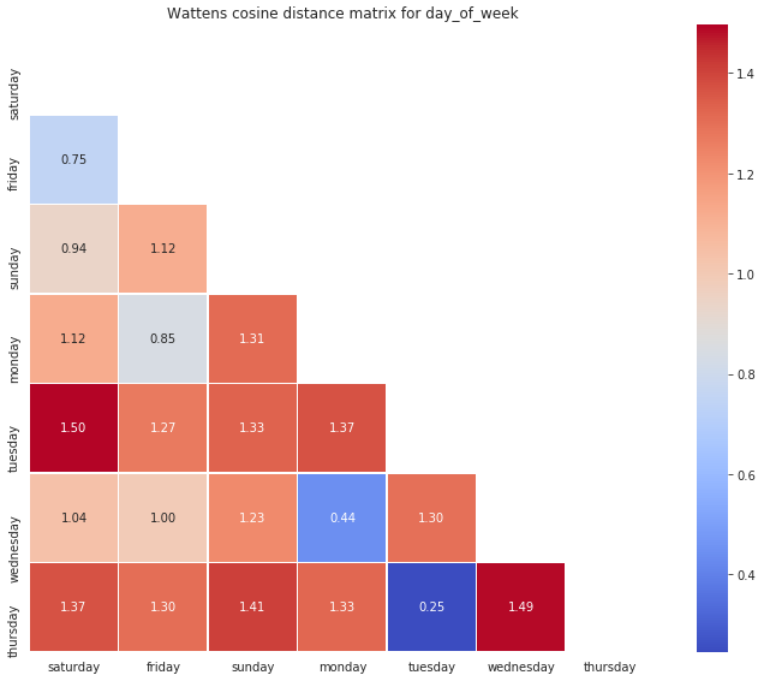


Figure 8.50: Wattens Outside sector heatmap of the cosine distance between the values of the variable Day of the Week.

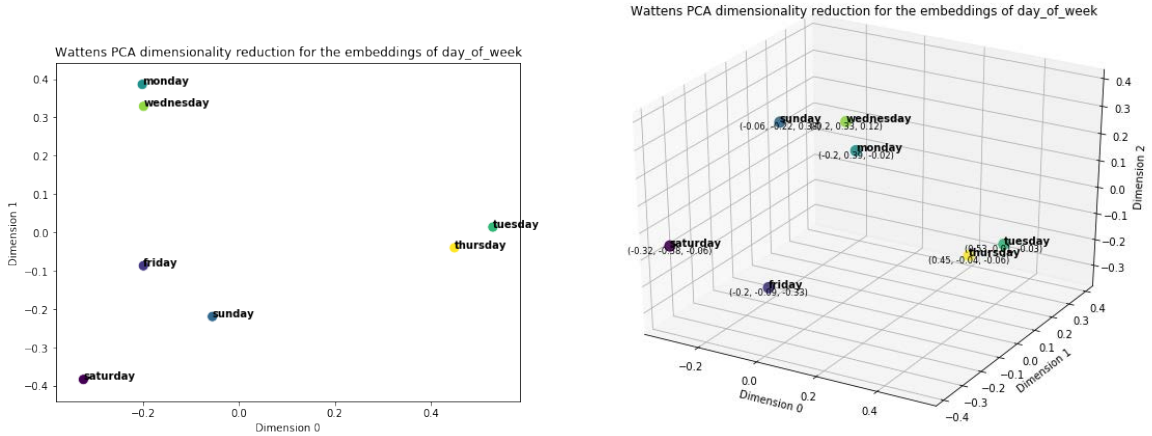


Figure 8.51: Wattens Outside scatterplots for the embeddings of the variable Day of the Week after PCA.

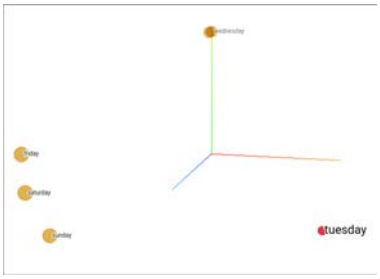


Figure 8.52: Wattens Outside scatterplots for the embeddings of the variable Day of the Week after tSNE.

patterns based on the day of the week, as the Wattens sectors have different behaviours according to those characteristics.

Los Angeles

Figure 8.53 presents the occupancy forecasts for the next six hours from the model created with the categorical variable Summary for the Los Angeles scenario (Standard sector). At first sight, it would seem that there is some effect on the parking occupancy, but not enough for forecasting. A weather table for Los Angeles on 17 October is shown in Table 8.3, which supports the explanations presented in this section.

Figure 8.54 shows the distance matrix, which is obtained after applying the cosine distance to the embeddings matrix of the Summary variable, thus serving as a heatmap for the Los Angeles Standard sector. Some of the vectors that form the embeddings seem to be closer in the embedding space, for example, *heavy* and *overcast* or *none* and *overcast*. Alternatively, other vectors like *clear* and *overcast* have the opposite meaning in relation to parking occupancy.

Figures 8.55 and 8.56 show the visualizations of the embedding values projected into lower dimensional space by PCA and tSNE. The PCA variance ratio explained by two and three principal components is 80% and 87%, meaning that there is no huge loss in information from using PCA. The principal components (axis in scatterplot) can be interpreted as dimension 1 corresponding to weather conditions related to *rainy*, higher values coincide with words like *rain*, *foggy* or *heavy* (this last always appears as *heavy rain*), while lower values have a more neutral meaning (*clear*, *none*, *partly*, or *possible*). For dimension 0, no intuitive explanation can be given. The visualization from tSNE shows three clusters that consist of *drizzle*, *heavy* and *overcast* (rain adjectives); *cloudy*, *foggy* and *rain* (weather conditions); and, lastly, *clear*, *partly*, *light*, *mostly* and *possible*, which could be related to tokens that indicate a more neutral relationship between weather and parking occupancy.

The Summary forecasts of parking occupancy can be similarly interpreted in the model trained with the categorical variable Precipitation Type, which is shown in Figure 8.57. Although the model offers different parking occupancy values based on the Precipitation

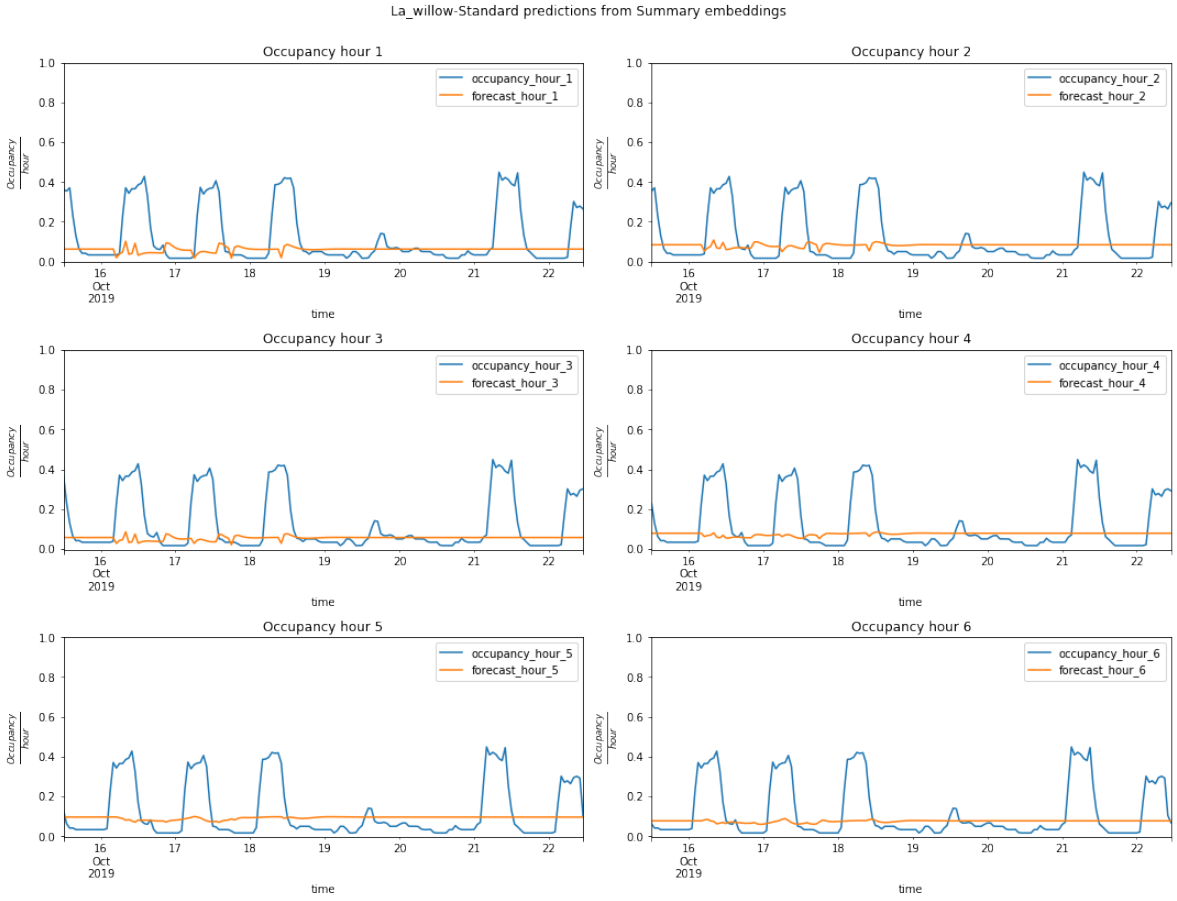


Figure 8.53: Los Angeles Standard sector forecasts using the categorical variable Summary.

time	precipType	summary
2019-05-17 00:00:00	none	Clear
2019-10-17 01:00:00	none	Clear
2019-10-17 02:00:00	none	Clear
2019-10-17 03:00:00	none	Clear
2019-10-17 04:00:00	none	Clear
2019-10-17 05:00:00	none	Partly Cloudy
2019-10-17 06:00:00	none	Partly Cloudy
2019-10-17 07:00:00	none	Partly Cloudy
2019-10-17 08:00:00	none	Partly Cloudy
2019-10-17 09:00:00	none	Partly Cloudy
2019-10-17 10:00:00	none	Partly Cloudy
2019-10-17 11:00:00	none	Partly Cloudy
2019-10-17 12:00:00	none	Partly Cloudy
2019-10-17 13:00:00	rain	Clear
2019-10-17 14:00:00	rain	Clear
2019-10-17 15:00:00	rain	Clear
2019-10-17 16:00:00	none	Clear
2019-10-17 17:00:00	none	Partly Cloudy
2019-10-17 18:00:00	rain	Clear
2019-10-17 19:00:00	none	Clear
2019-10-17 20:00:00	none	Clear
2019-10-17 21:00:00	rain	Clear
2019-10-17 22:00:00	rain	Clear
2019-10-17 23:00:00	rain	Clear

Table 8.3: Los Angeles weather information for 17 October 2019.

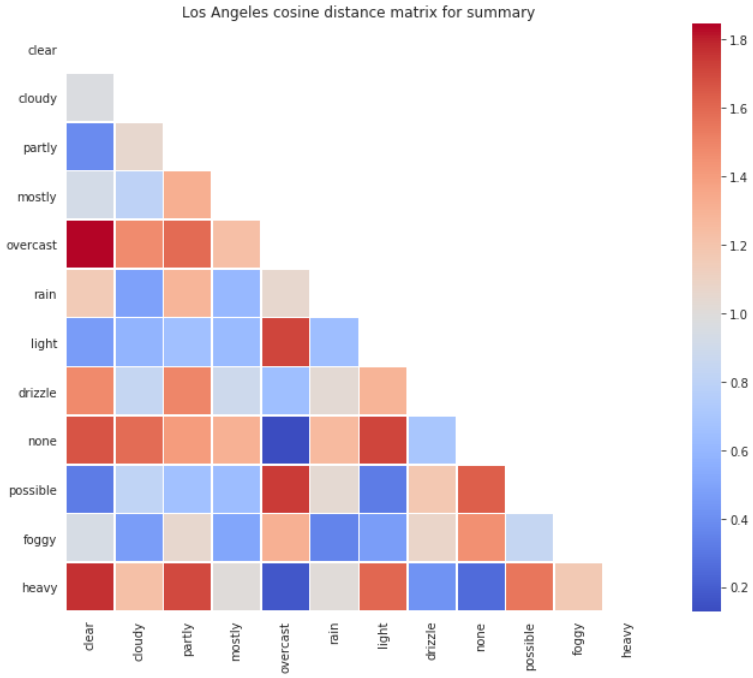


Figure 8.54: Los Angeles Standard sector heatmap of the cosine distance between the values of the variable Summary.

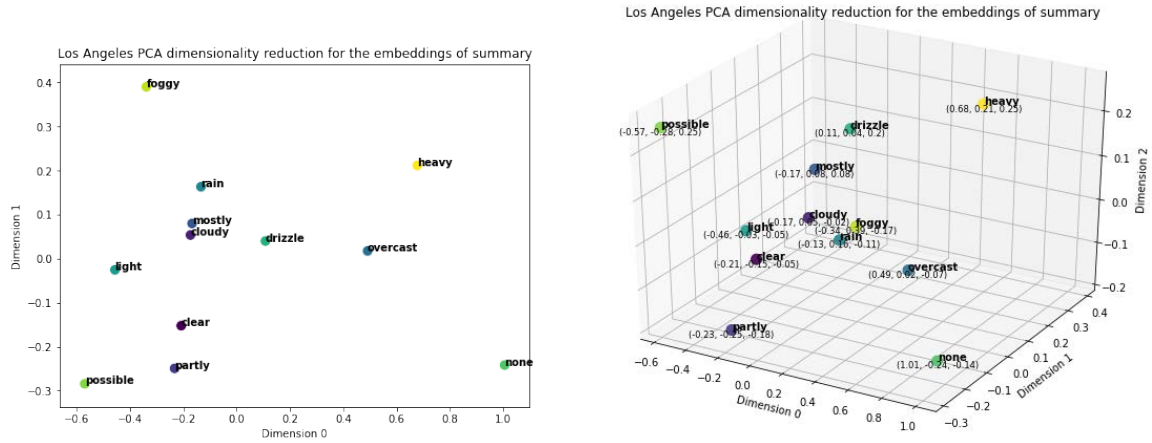


Figure 8.55: Los Angeles Standard sector scatterplots for the embeddings of the variable Summary after PCA.

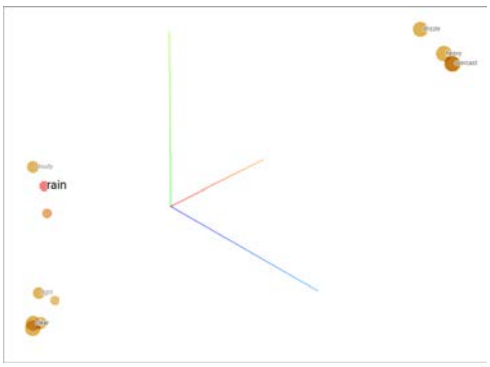


Figure 8.56: Los Angeles Standard sector scatterplots for the embeddings of the variable Summary after tSNE.

Type value, it is not enough to provide good forecasts. Therefore, it could be used as support information. One reason for the the categorical variable Precipitation Type having low expressivity for Los Angeles is due to it having only two possible values: none and rain.

Due to the lack of values in the Precipitation Type variable, cosine distance and dimension reduction methods provide no useful information.

Figure 8.58 provides the visualizations for the Los Angeles Standard sector forecasts at different time steps from the model trained with the exogenous variable Month. They show that patterns exist between the current month and the occupancy levels in the sector, and these can be useful for the modeling (at least for the forecasts at the first time steps). Clearly, the occupancy is lower in the Summer months and higher in the Autumn months, although the the occupancy pattern changes due to street works (2019) could have a biased effect on the model's learning, as this could cause forecasts below zero at higher time steps and low parking occupancy values for the Spring months.

The heatmap of the cosine distance matrix is shown in Figure 8.59. Although there are a lot of months with neutral distance (value of 1), it can be clearly seen that the Spring and Autumn months have a greater distance between them. This could be an effect of the street works on the sector during the Spring and Summer of 2019.

The results from applying PCA and tSNE to the embeddings of Month are shown in Figures 8.60 and 8.61. PCA brings together those months that are less affected by the street works (January, December, November, September, October and August) and those that are more affected by them (April, May, June, July), placing each group to one side along dimension 0 (first principal component). The visualization from tSNE is similar to the one obtained by PCA, as it creates one group for the months of April, March and February, and in another for November, October and August. The rest of the months are scattered throughout the visualization.

The forecasts for the model trained with the categorical variable Holiday are presented in Figure 8.62. The visualizations show that the model predicts a mean value for non-holidays (21, 22 and 23 of December) and a different occupancy value for holidays (24, 25

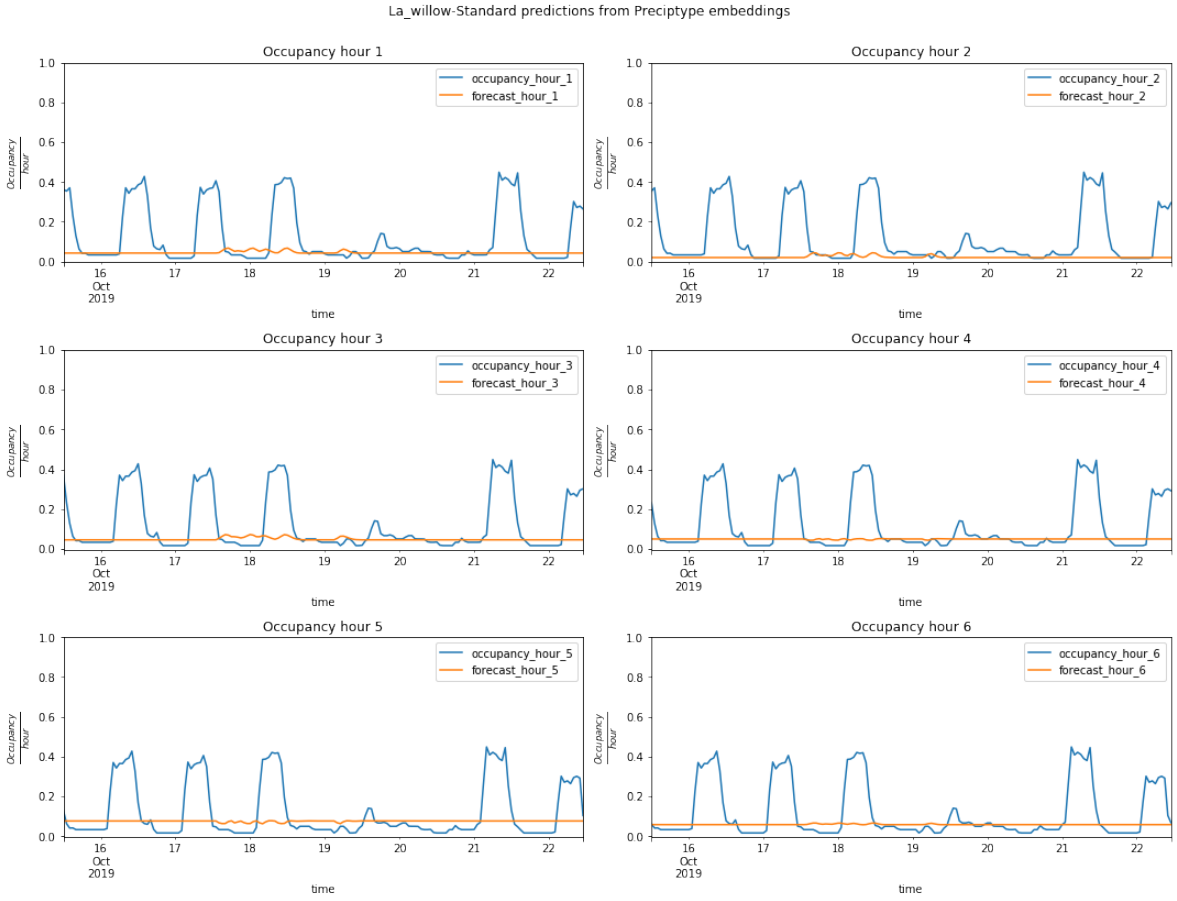


Figure 8.57: Standard sector forecasts using the categorical variable Precipitation Type.

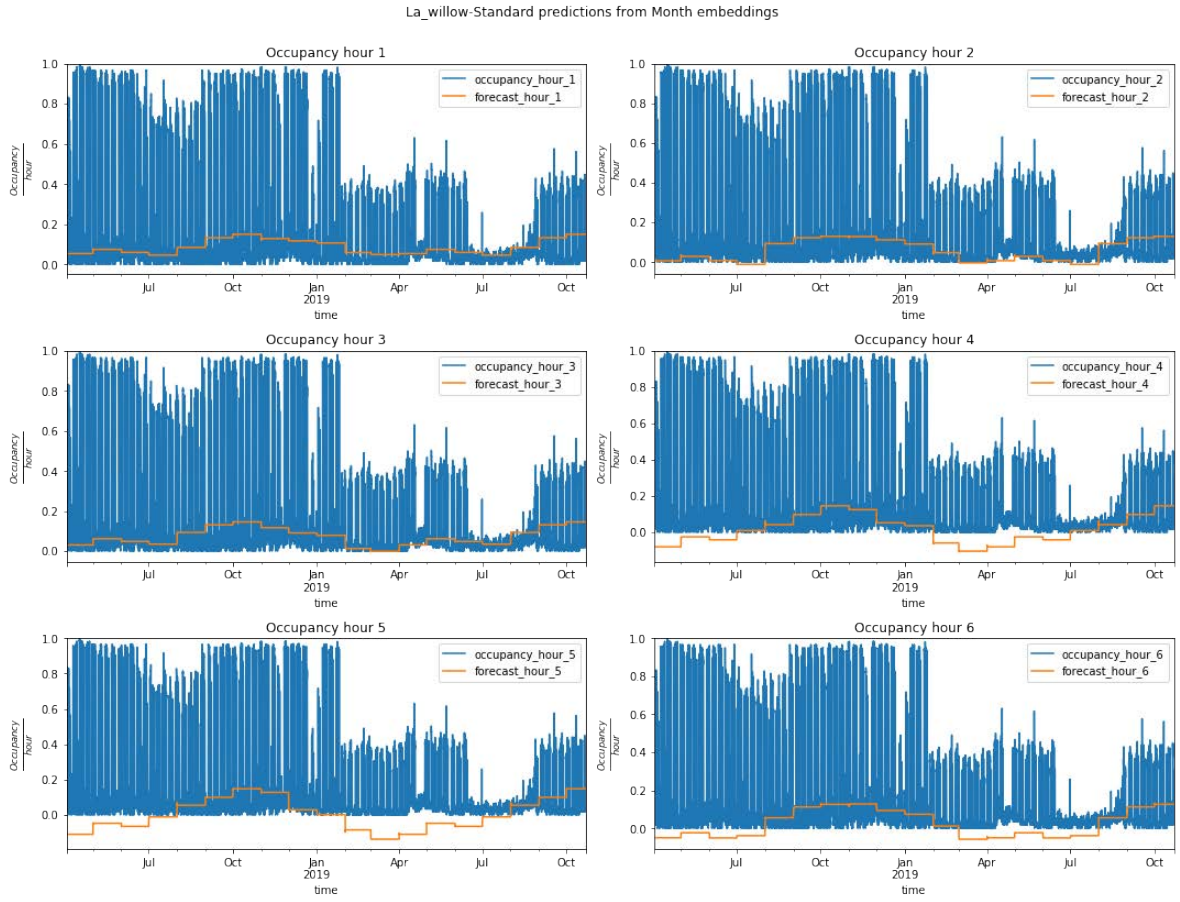


Figure 8.58: Los Angeles Standard sector forecasts from a model trained with the categorical variable Month.

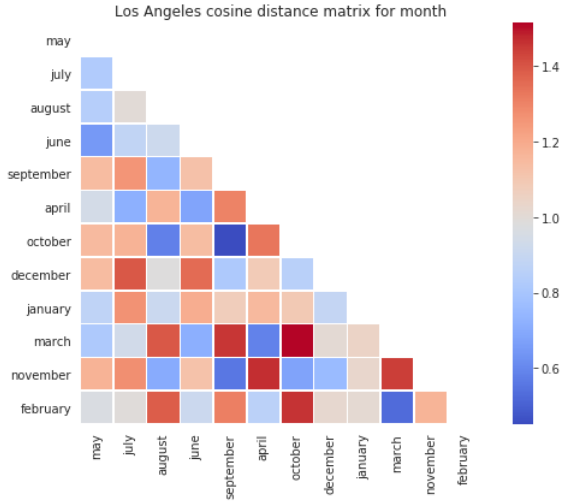


Figure 8.59: Los Angeles Standard sector heatmap of the cosine distance between the values of the variable Month.

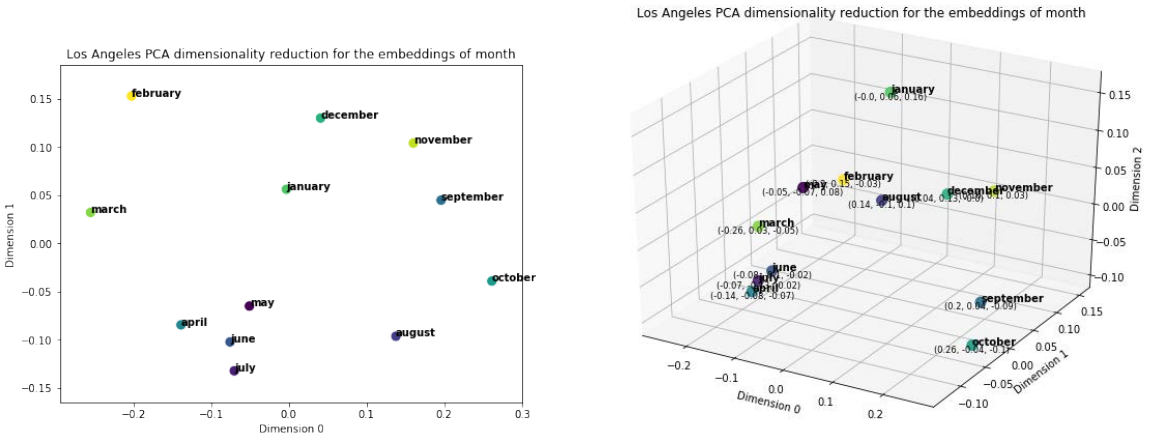


Figure 8.60: Los Angeles Outside scatterplots for the embeddings of the variable Month after PCA.

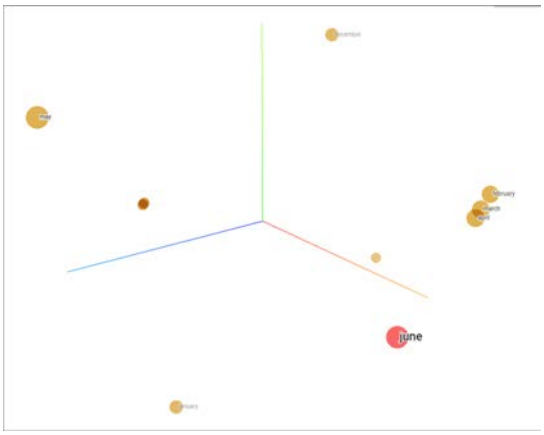


Figure 8.61: Los Angeles Standard scatterplots for the embeddings of the variable Month after tSNE.

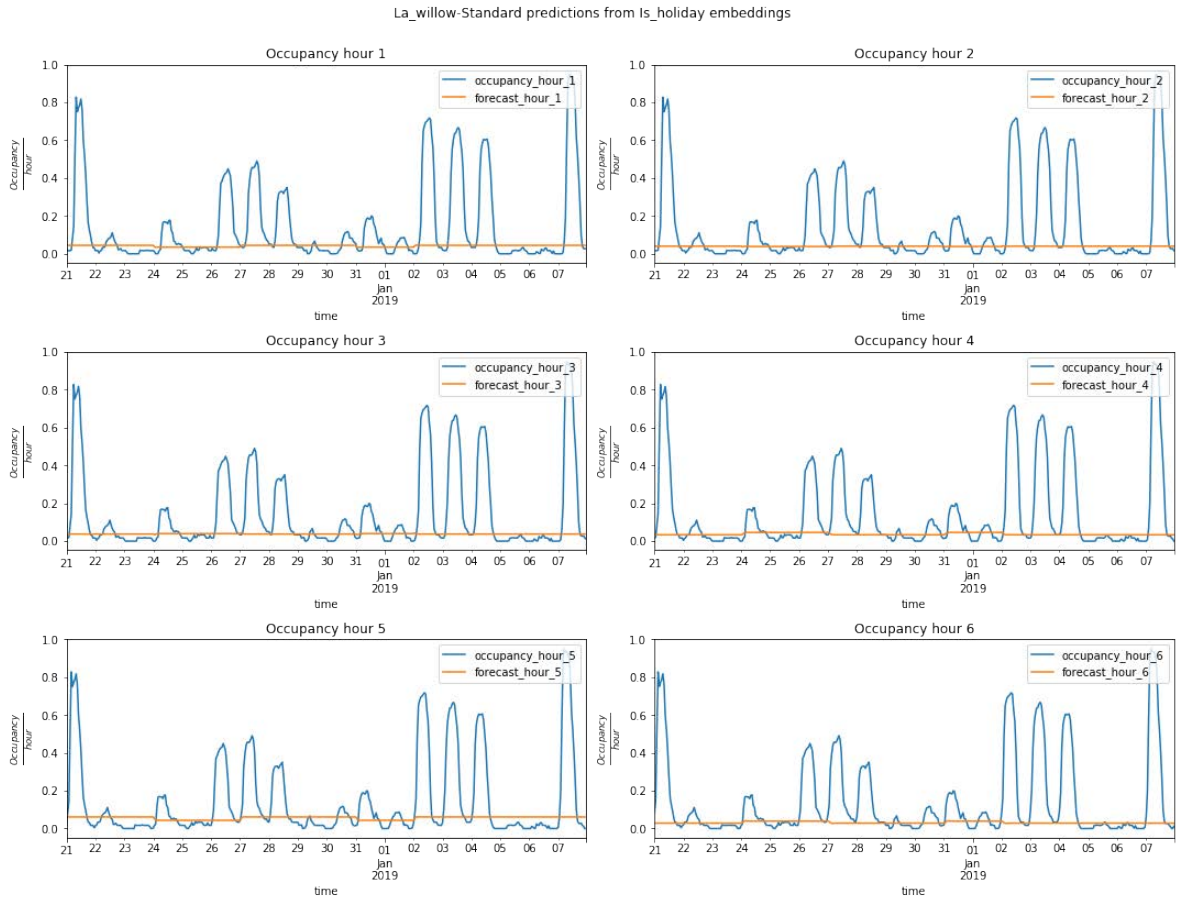


Figure 8.62: Los Angeles Standard sector forecasts from the model trained with the categorical variable Holiday.

and 26 of December). Although the model forecasts less occupancy levels for holidays, it does not show any behaviour like that of the Wattens Outside sector, in which the value of the variable Holiday could provide the model enough information to give good forecasts for some days (specially when changing from holiday to non-holiday).

Due to the lack of values in the variable (which is binary), the heatmap and the dimensionality reduction plots do not offer any information.

The last model for Los Angeles using only categorical variables corresponds to the one trained with the Day of the Week information. Figure 8.63 shows the forecasts from the model when trying to predict the next six hours of parking occupancy. Similarly to the previous scenario, the Day of the Week data provides the model with enough information to forecast the recurrent patterns of the parking occupancy while discriminating weekdays from weekends.

The heatmap of the cosine distance matrix from Day of the Week is shown in Figure 8.64. Vectors representing weekend days are closer to each other than the rest of the days, and some of the weekdays present greater distances (Monday and Tuesday, for

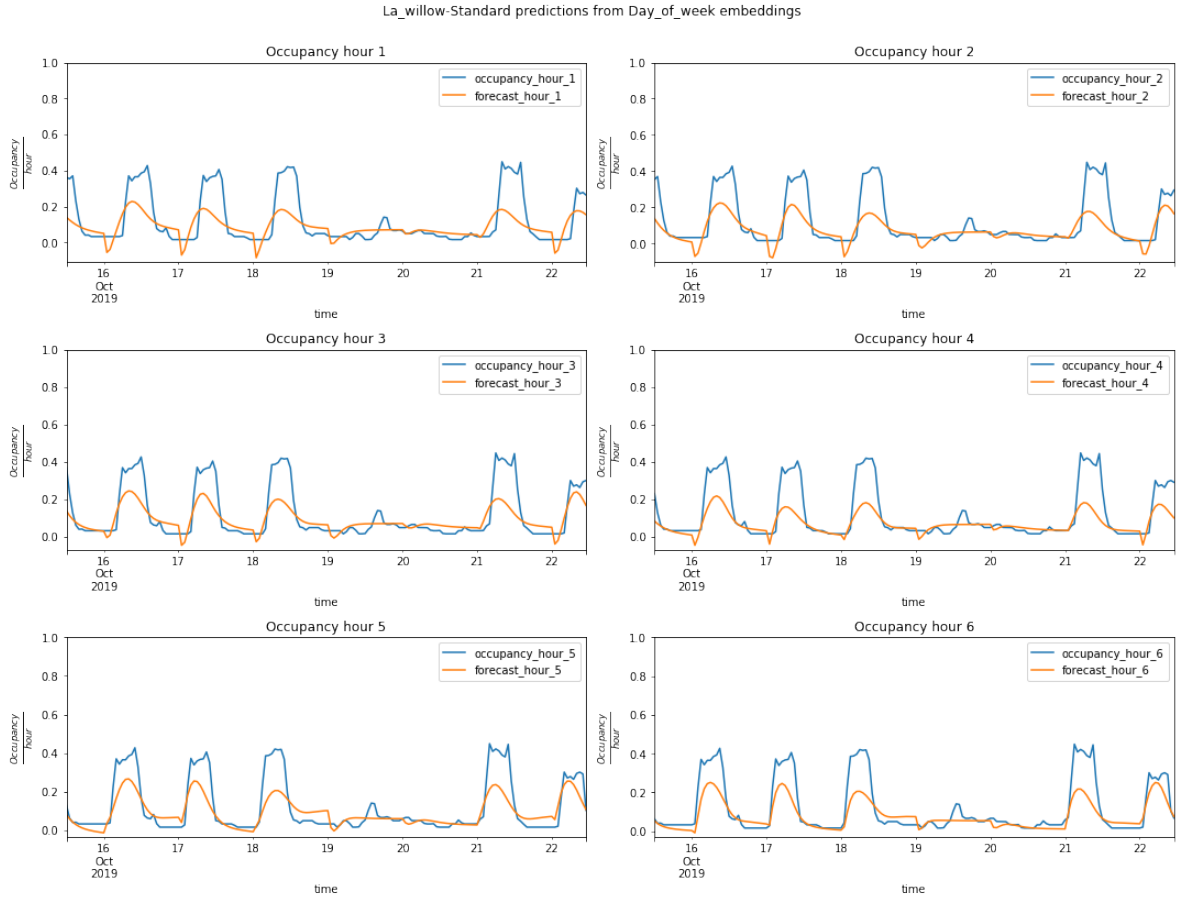


Figure 8.63: Forecasts from a model trained with the categorical variable Day of Week for the Los Angeles Standard sector.

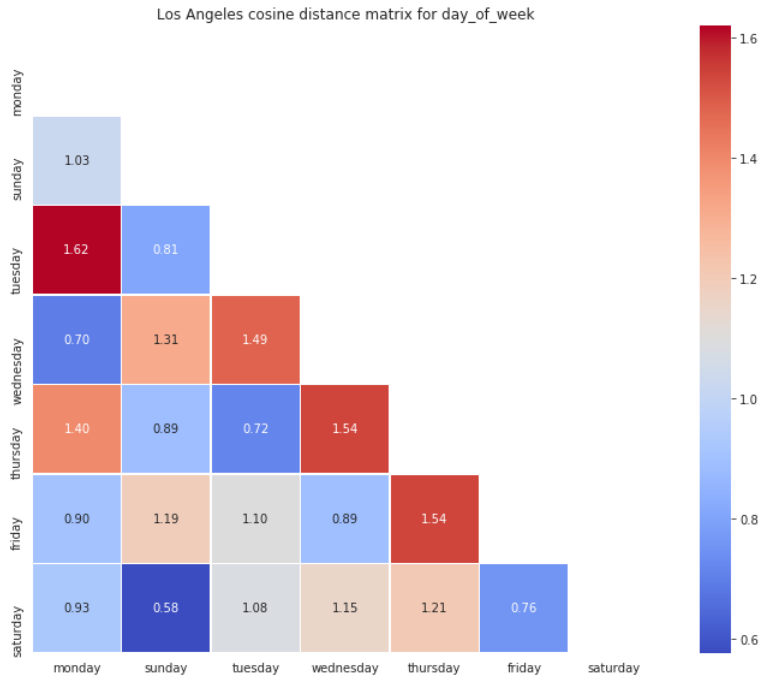


Figure 8.64: Los Angeles Standard sector heatmap of the cosine distance between the values of the variable Day of the Week.

example).

The visualizations in Figures 8.65 and 8.66 show how the values of the embeddings are positioned after applying PCA and tSNE. Three principal components explain up to 77% of the variance of the data (60% with two), so the projection of the data provided by PCA can provide strong enough intuitions. PCA shows that Friday and Saturday are very similar, and the same happens with Wednesday and Monday. The tSNE visualization presents three distinct groups of relationships between day of the week and parking occupancy:

1. Tuesday and Thursday.
2. Saturdays and Sunday.
3. Wednesday, Monday and Friday.

The categorical exogenous variables for the Los Angeles Standard sector show that they can provide some useful information, especially day of the week, as it is able to differentiate weekdays and weekend patterns. The visualizations show that street works in 2019 affected how exogenous information changed its relationship with the parking occupancies.

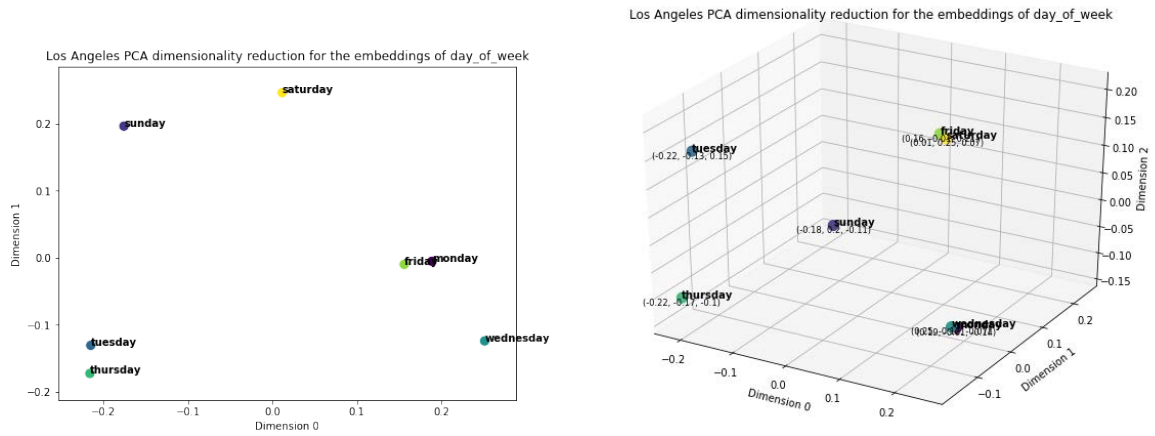


Figure 8.65: Los Angeles Standard scatterplots for the embeddings of the variable Day of the Week after PCA.

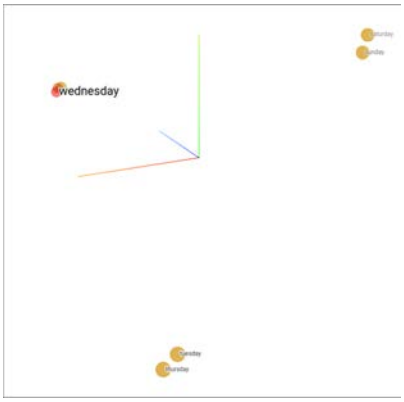


Figure 8.66: Los Angeles Standard scatterplots for the embeddings of the variable Day of the Week after tSNE.

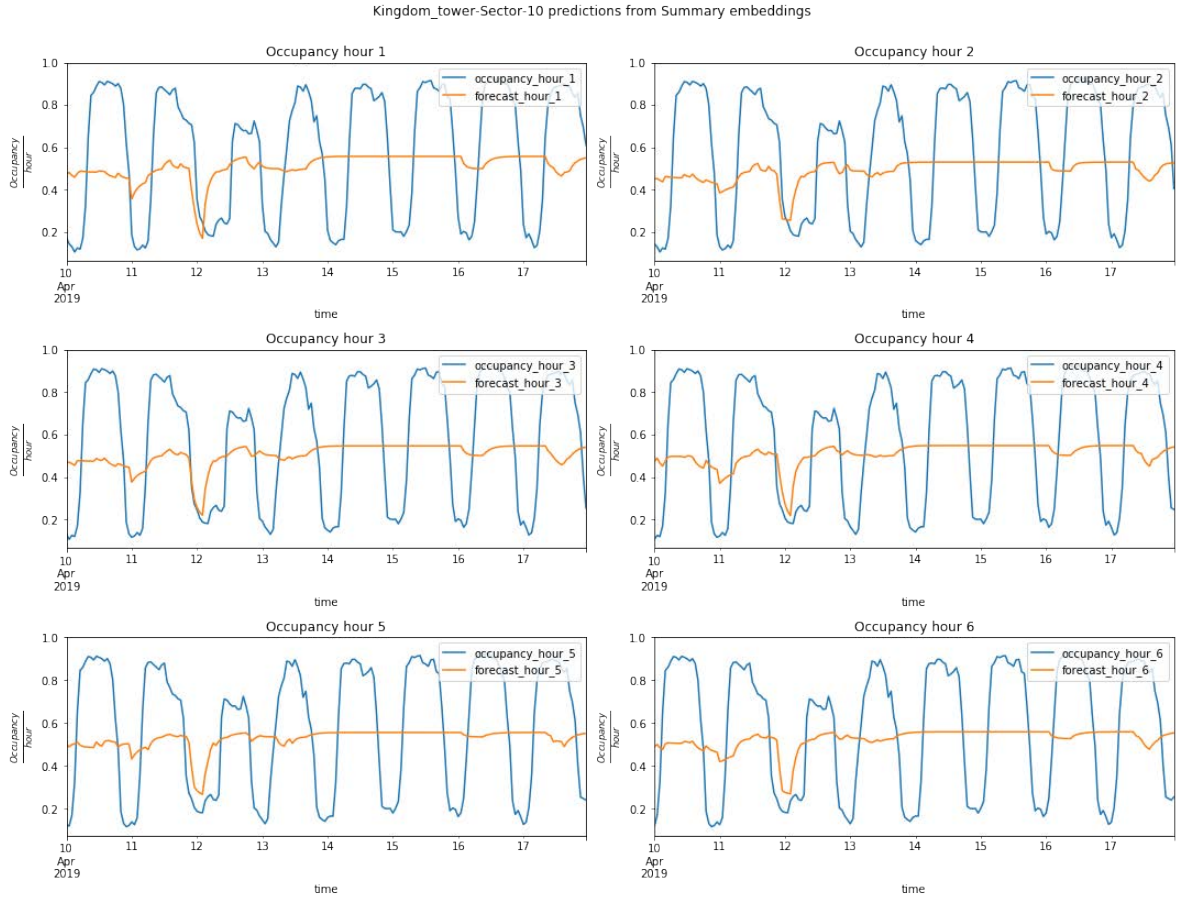


Figure 8.67: Sector-10 from Riyadh forecasts using the categorical variable Summary.

Riyadh

The regular patterns of special weather conditions, cultural behaviours and parking occupancy indicated by the high number of sensors show that exogenous variables can be very useful for improving the parking occupancy forecasts for the Riyadh scenario.

Figure 8.67 presents the occupancy forecasts for the next six hours from the model created with the categorical variable Summary for Riyadh Sector-10. The visualizations show that weather information has an effect on the parking occupancy, which is notable on rainy days (12 April). The problem is that the lack of variability in Riyadh limits the usefulness of the weather information.

Table 8.4 shows the weather information for 12 April 2019 to support the text.

Figure 8.68 shows the distance matrix obtained when the cosine distance is applied to the embeddings of the Summary variable as a heatmap for Sector-10 in Riyadh. The lack of variability in the weather conditions is reflected in the cosine distance matrix, as the Summary information presents lower possible values when compared to previous scenarios. The most noticeable fact is that *foggy* and *rain* (and to a lesser extent, *windy*),

time	precipType	summary
2019-04-12 00:00:00	rain	Rain
2019-04-12 01:00:00	rain	Rainr
2019-04-12 02:00:00	rain	Mostly Cloudy
2019-04-12 03:00:00	rain	Mostly Cloudy
2019-04-12 04:00:00	rain	Partly Cloudy
2019-04-12 05:00:00	rain	Partly Cloudy
2019-04-12 06:00:00	none	Mostly Cloudy
2019-04-12 07:00:00	none	Partly Cloudy
2019-04-12 08:00:00	none	Partly Cloudy
2019-04-12 09:00:00	none	Partly Cloudy
2019-04-12 10:00:00	rain	Partly Cloudy
2019-04-12 11:00:00	rain	Clear
2019-04-12 12:00:00	rain	Clear
2019-04-12 13:00:00	rain	Clear
2019-04-12 14:00:00	rain	Clear
2019-04-12 15:00:00	none	Clear
2019-04-12 16:00:00	none	Clear
2019-04-12 17:00:00	none	Clear
2019-04-12 18:00:00	rain	Partly Cloudy
2019-04-12 19:00:00	rain	Mostly Cloudy
2019-04-12 20:00:00	rain	Partly Cloudy
2019-04-12 21:00:00	rain	Clear
2019-04-12 22:00:00	rain	Clear
2019-04-12 23:00:00	rain	Partly Cloudy

Table 8.4: Riyadh Sector-10 weather information for 12 April 2019.

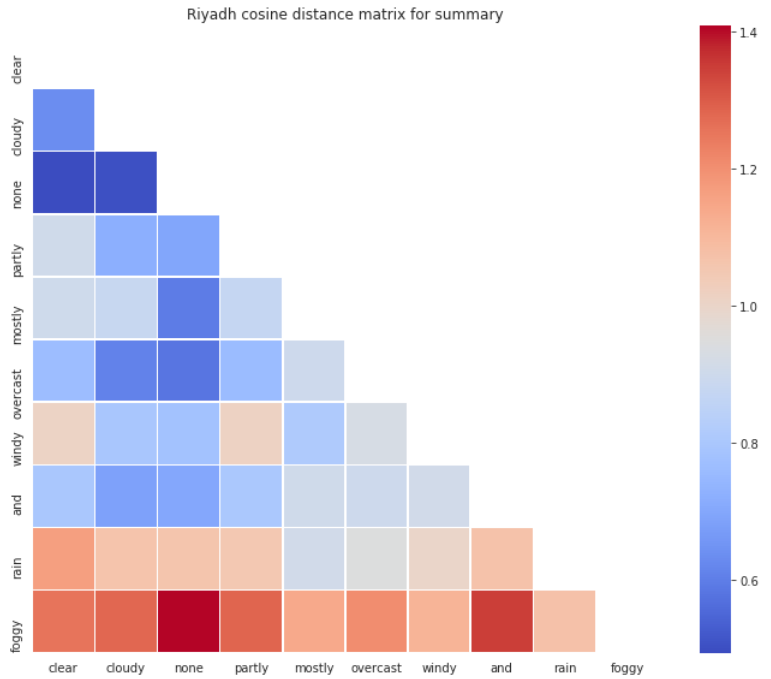


Figure 8.68: Heatmap of the Riyadh Sector-10 cosine distance between the values of the variable Summary.

have an opposite effect on parking occupancy when compared to the values *clear*, *cloudy* or *none*.

After applying the dimensionality reduction methods PCA and tSNE, the scatterplots are shown in Figures 8.69 and 8.70 . When using two principal components, PCA is able to explain up to 40% of the variance in the data. With three of them, it explains up to 51%. Therefore, tSNE will provide better explanations, as the relationships could be non-linear. The dimension 0 in the PCA with 3 principal components has positive values for *foggy*, *windy* and *rain*, while the rest are negative, indicating a relationship between parking occupancy and the variable Summary. After trying different values for the perplexity parameter, the visualization of the tSNE projections shows a small neighbourhood formed by *windy*, *rain* and *foggy*; and another formed by the remaining words with tokens like *mostly*, thus forming its own cluster apart from the rest. It could be interpreted that Sector-10 in Riyadh has different occupancy behaviours due to weather conditions.

The other weather categorical variable is Precipitation Type, whose forecasts are presented in Figure 8.71. It could be expected to behave similarly to the Summary variable on rainy days, but the forecasts show that the information offered by the variable is not powerful enough to forecast parking occupancies, although it seems that it could be used to provide auxiliary information that can help the model in its forecasts.

Due to the lack of values in the variable Precipitation Type, the heatmap and scatterplots from dimension reduction algorithms cannot offer useful information.

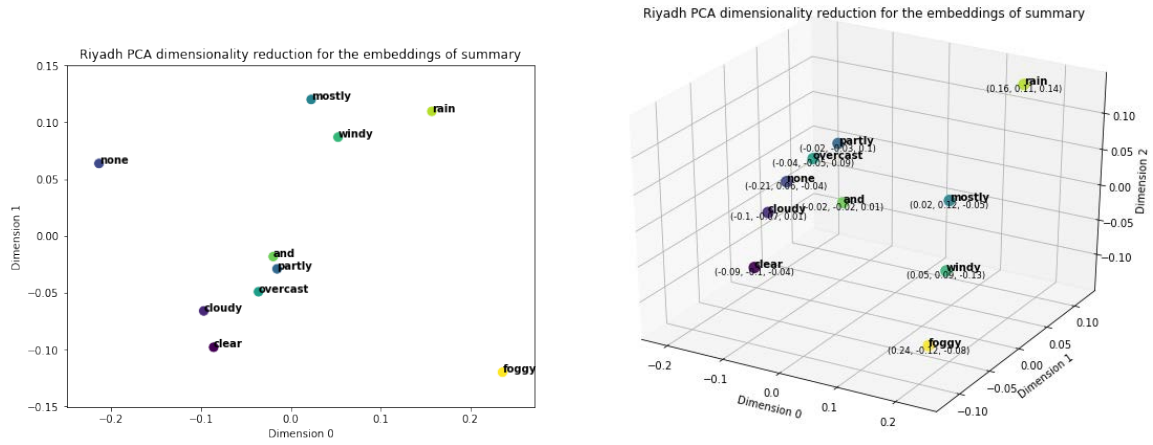


Figure 8.69: Riyadh Sector-10 scatterplots for the embeddings of the variable Summary after PCA.

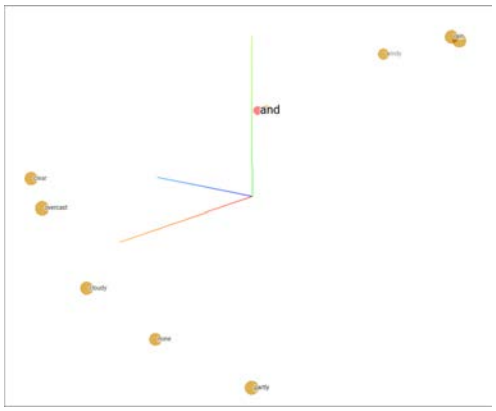


Figure 8.70: Riyadh Sector-10 scatterplots for the embeddings of the variable Summary after tSNE.

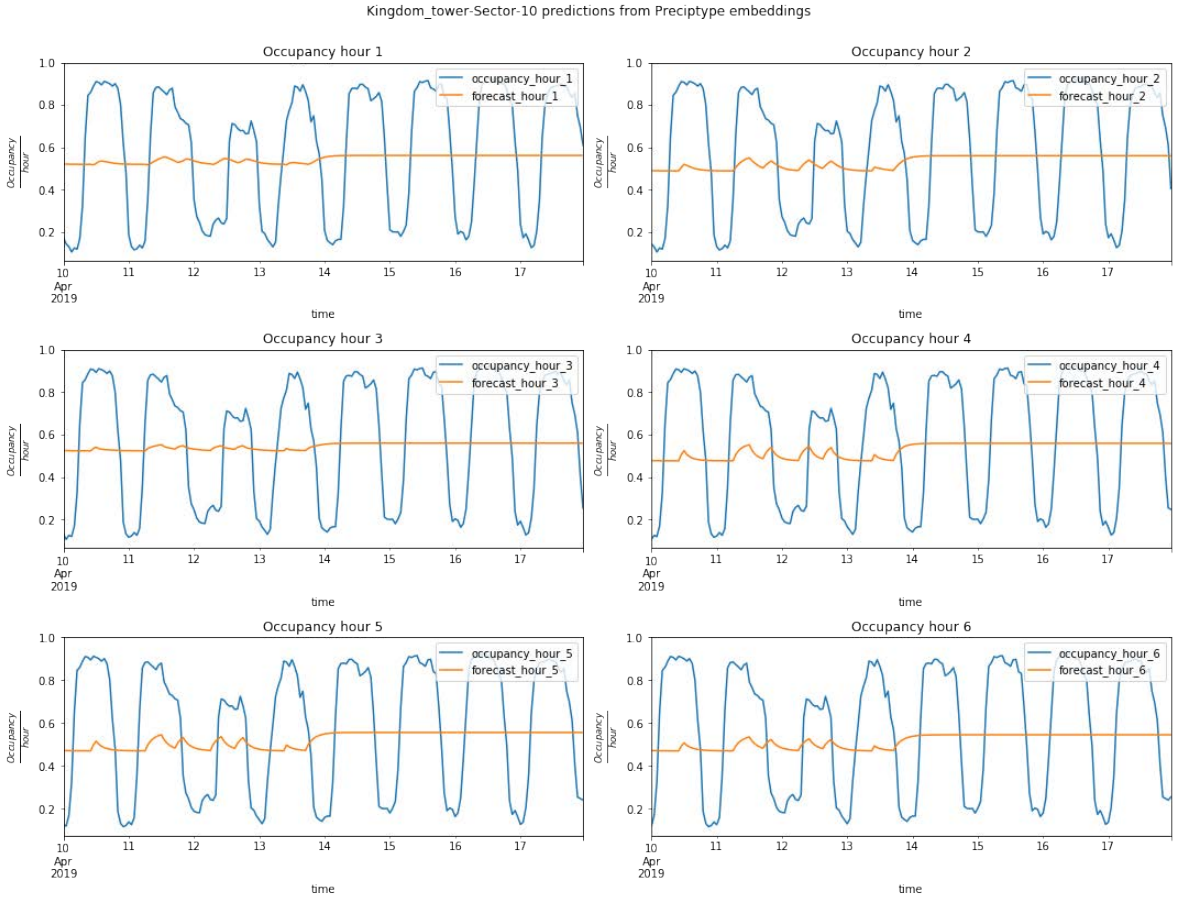


Figure 8.71: Forecasts for Riyadh Sector-10 with the categorical variable Precipitation Type.

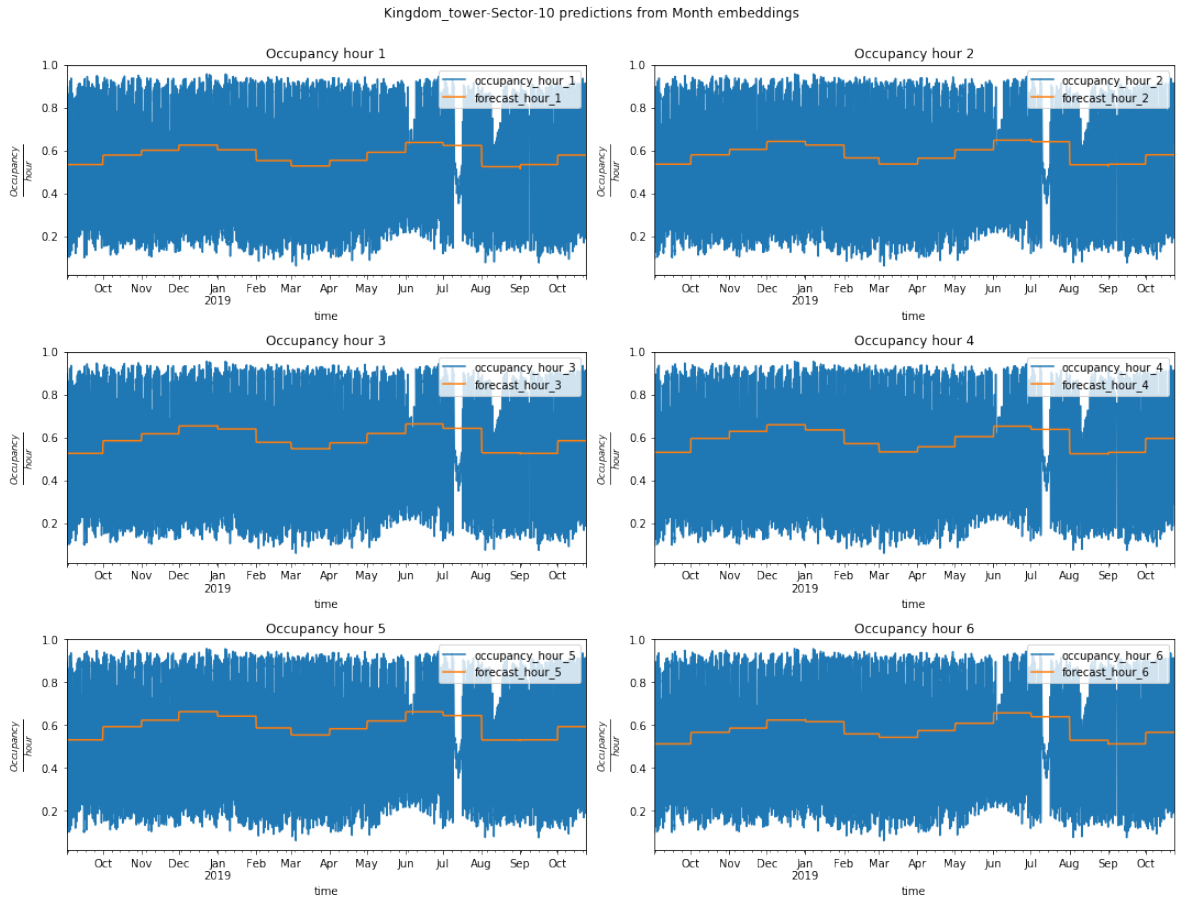


Figure 8.72: Riyadh Sector-10 forecasts from a model trained with the categorical variable Month.

Forecast visualizations for different time steps for the model trained with the exogenous variable Month are provided in Figure 8.72. The cultural aspects related to the holidays can be useful for interpreting the forecasts of the model. The lower levels of occupancy happen in August and September, as these months have a lot of national holidays, specifically the Feast of Sacrifice and the celebration of the Islamic new year. On the other hand, and contrary to Western scenarios, the month with the most occupancy forecasted by the model is December, as no holidays take place in that month.

The heatmap of the cosine distance matrix is shown in Figure 8.73. While some months are very close to actual parking occupancy patterns (January/February or January/July), there is only one pair with opposite meanings, namely May/August.

The results from applying PCA and tSNE to the embeddings of Month are shown in Figures 8.74 and 8.75. PCA with three components explains up to 53% of the variability of the embeddings layer. Therefore, it is possible that tSNE provides better visualization. The tSNE visualization shows the following groups:

- June, July, August, September and October.

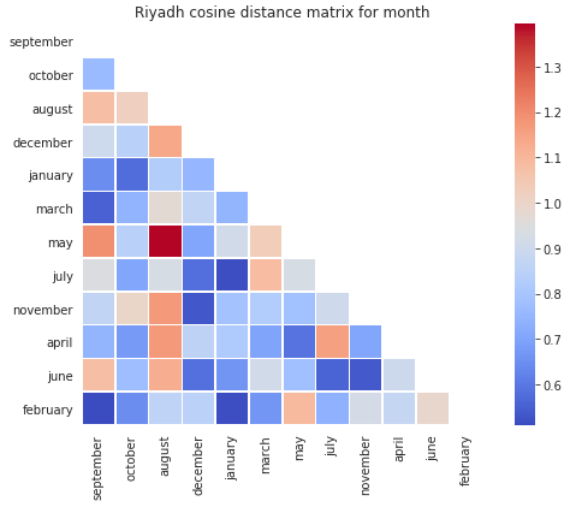


Figure 8.73: Sector-10 in Riyadh heatmap of the cosine distance between the values of the variable Month.

- January, February, March and April.
- November and December.
- May.

It seems that the first months of the year have parking occupancy patterns that are different from those between the middle and the end of the year. As already explained in Chapter 3, May coincides with Ramadan, for which changes in the parking occupancy patterns were detected.

The forecasts for the model trained with the categorical variable Holiday are presented in Figure 8.76. The visualizations show that the model predicts a constant value for

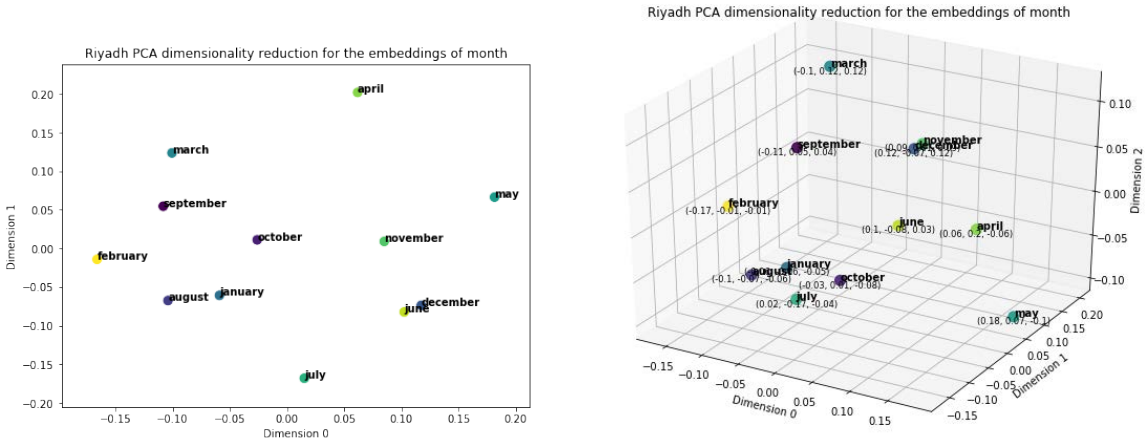


Figure 8.74: Riyadh Sector-10 scatterplots for the embeddings of the variable Month after PCA.

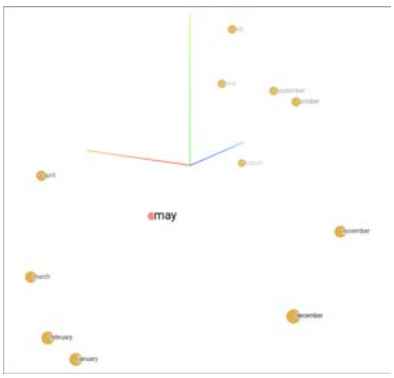


Figure 8.75: Riyadh Sector-10 scatterplots for the embeddings of the variable Month after tSNE.

non-holidays (1, 2 and 3 of June) and a different occupancy value (lower) for holidays (5, 6 and 7 of December). Similarly to what was observed in the Wattens scenario, the model provides more accurate forecasts when there is a change from holiday to non-holiday. The forecasts show that the model learns that the occupancy levels are lower on holidays than on the other days, making this useful information to incorporate into the final model.

Due to the lack of values in the variable (which is binary), the heatmap and the dimensionality reduction plots do not offer any information.

Lastly, the visualizations of the forecasts for the categorical variable Day of the Week are presented in Figure 8.77. Similarly to the Antwerp and Wattens scenarios, this is the categorical variable that offers better forecasts, as the model has learnt the different patterns of occupancy during the week.

The heatmap of the cosine distance matrix from the Day of the Week is shown in Figure 8.78. The most similar days in terms of parking occupancy are Tuesday/Thursday, Sunday/Wednesday and Monday/Saturday. The most dissimilar pair is Monday/Wednesday.

The visualizations in Figure 8.79 and 8.80 show how the values of the embeddings are positioned after applying PCA and tSNE. Using two principal axes, PCA explains up to 80% of the variance of the Day of the Week embedding, and adding a third principal component increases the explained variance up to nearly 90%. The two dimensional visualization shows that Tuesday and Thursday occupancy values are very similar, and the same happens with Wednesday and Sunday. The tSNE visualization shows three clusters:

- Wednesday, Friday and Sunday.
- Tuesday and Thursday.
- Monday and Saturday.

For the Riyadh scenario, the categorical exogenous variable proves to offer useful information that can help improve parking occupancy predictions. Of all the variables

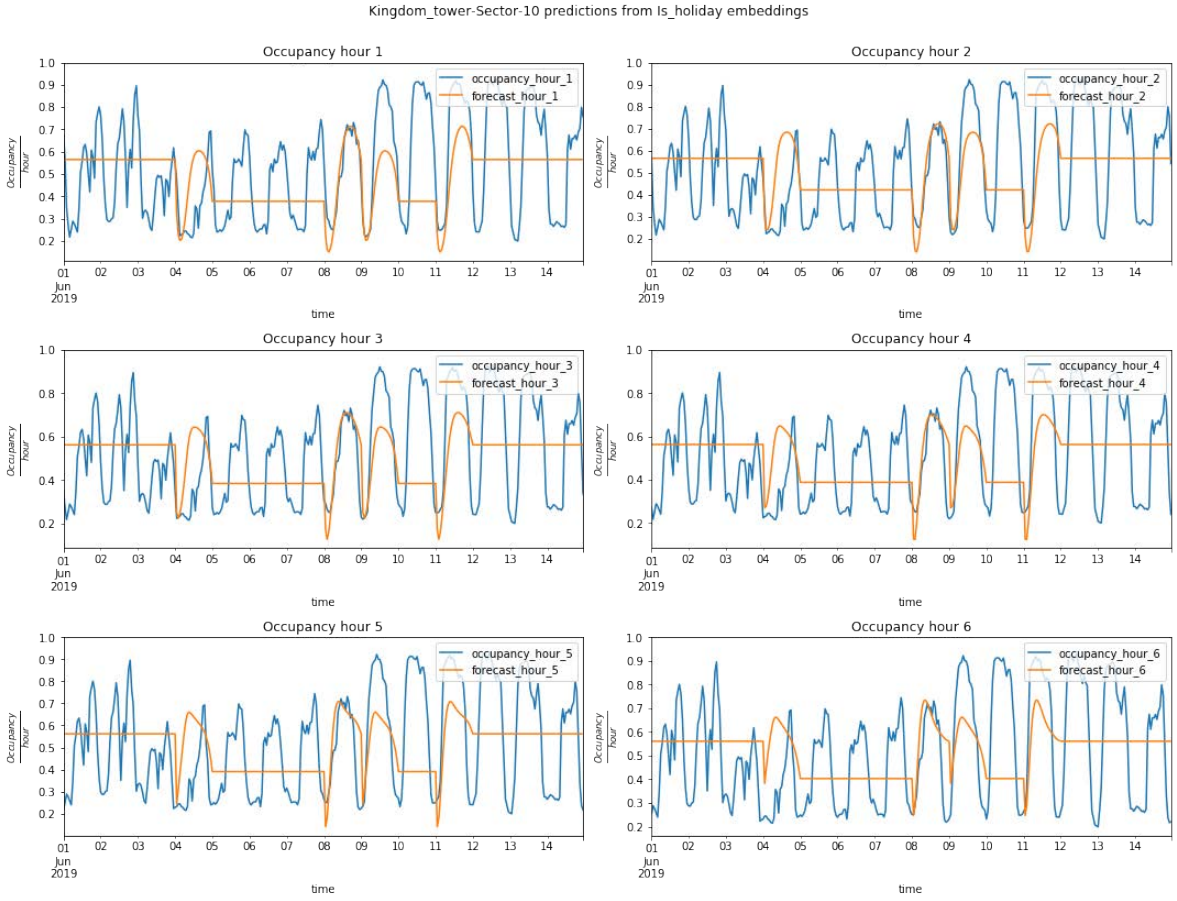


Figure 8.76: Riyadh sector forecasts from the model trained with the categorical variable Holiday.

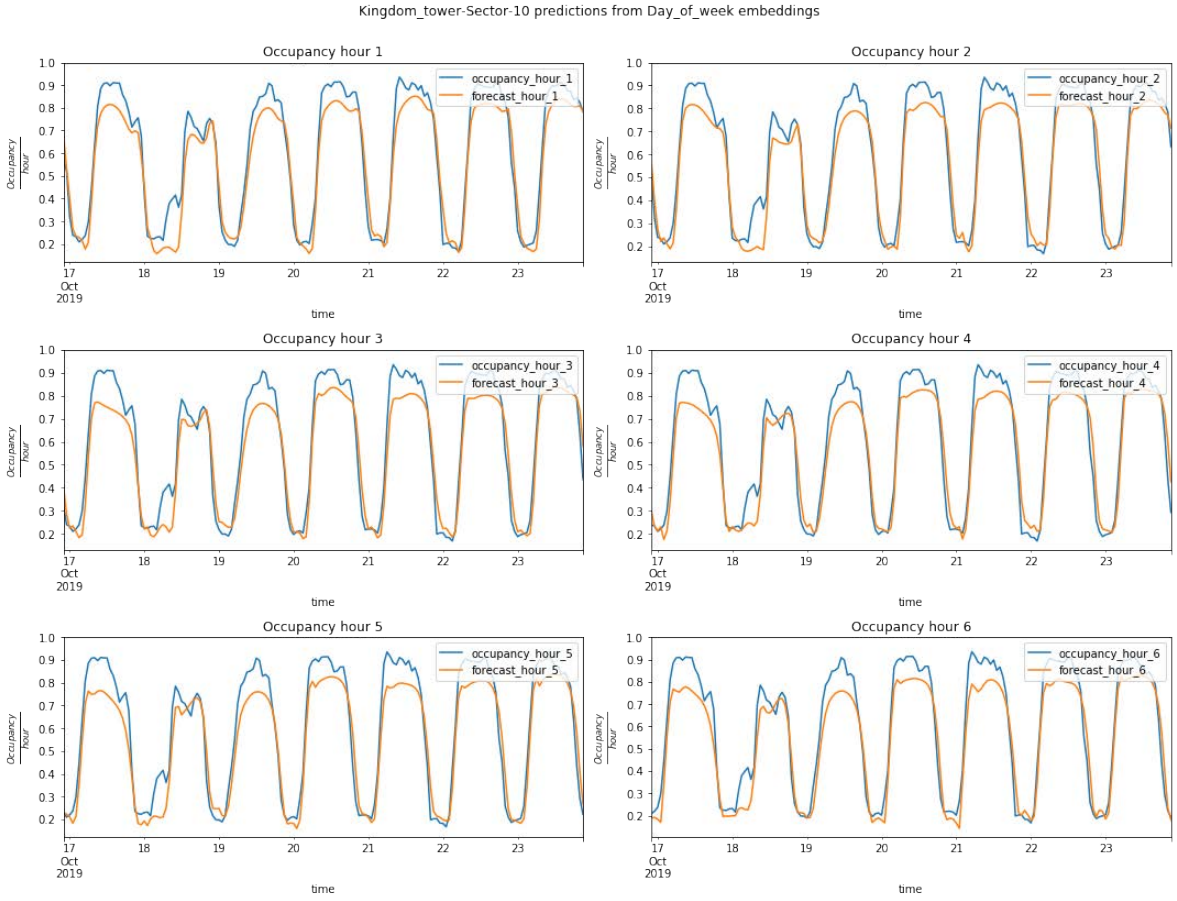


Figure 8.77: Forecasts of parking occupancy for Riyadh Sector-10 from a model trained with the categorical variable Day of Week.

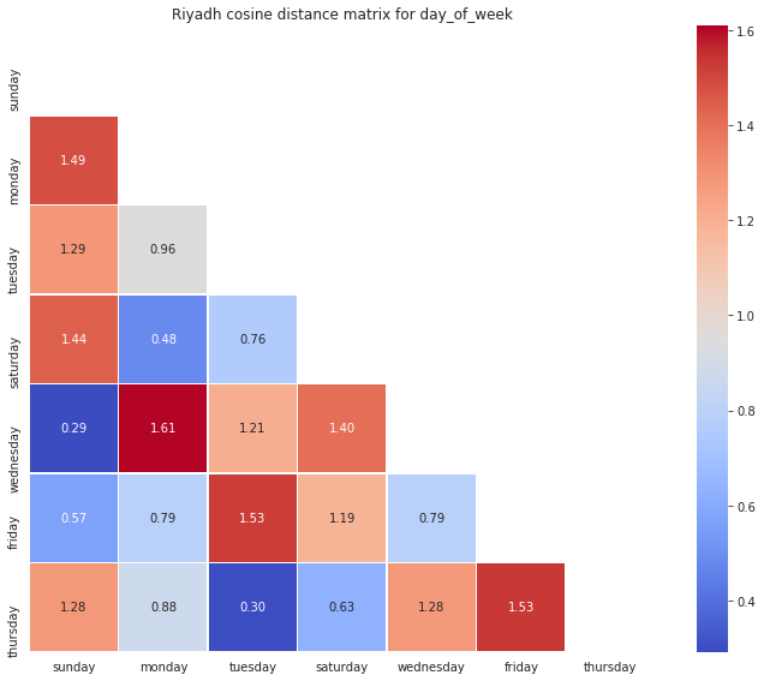


Figure 8.78: Riyadh Sector-10 heatmap of the cosine distance between the values of the variable Day of the Week.

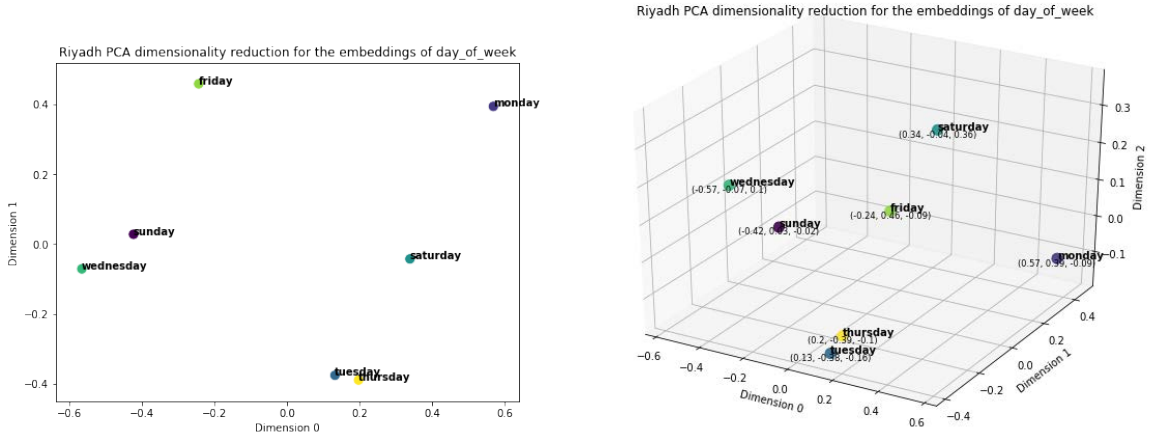


Figure 8.79: Riyadh Sector-10 scatterplots for the embeddings of the variable Day of the Week after PCA.

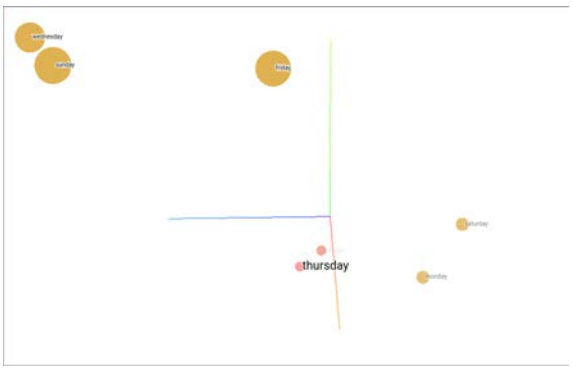


Figure 8.80: Riyadh Sector-10 scatterplots for the embeddings of the variable Day of the Week after tSNE.

considered, the only one that seems to have little impact is Precipitation Type, and this is due to the low variation in the weather conditions.

8.2.3 All the numerical variables

A model has been created using as input the following variables:

- Temperature.
- Cloud Cover.
- Dew Point.
- Humidity.
- Wind Bearing.
- Wind Speed.
- Wind Gust.
- UV Index.
- Visibility.
- Precipitation Intensity.
- Precipitation Accumulated.

The model consists of:

- Two GRU layers of 128 units each. A dense output layer with six neurons that takes the last output of the previous GRU layer and computes the parking occupancy for the next six time steps.
- Adam optimizer.
- 50 epochs with early stop.

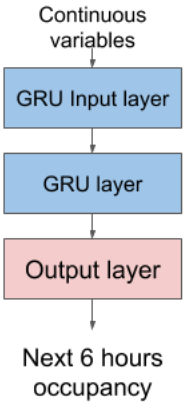


Figure 8.81: Schema of the architecture used to model all continuous variables (except parking occupancy) in order to forecast parking occupancy values.

- 128 as batch size.
- RMSE as loss.

As explained previously, all the continuous variables are preprocessed and therefore their range of values is between zero and one. Then, the time series are created as sequences of the twenty-four past values for each variable. The data obtained after preprocessing consists of a tensor of shape $(samples, sequence\,length, number\ exogenous\ variables)$. This type of data can be directly used in RNN layers, as they specialize in processing sequence data. Two RNN layers are used, as the first projects the inputs into a higher dimensional space (128 dimensions) from which the second GRU will extract useful information for providing forecasts of parking occupancy. In the same way as the endogenous models, an output layer of six neurons is used to provide the next six hours of parking occupancy.

A schema of the model is presented in Figure 8.81. The visualization summarizes what has been explained previously about the model architecture used.

Figure 8.82 presents the Antwerp Pelikaanstraat sector's next six hours of parking occupancy forecasts, which were provided by a model trained with all exogenous continuous variables. The forecasts show that the model has learnt that some relationship exists throughout the days (daytime and nighttime), but it is not capable of distinguishing weekend and weekeday patterns or hourly changes in the parking occupancy.

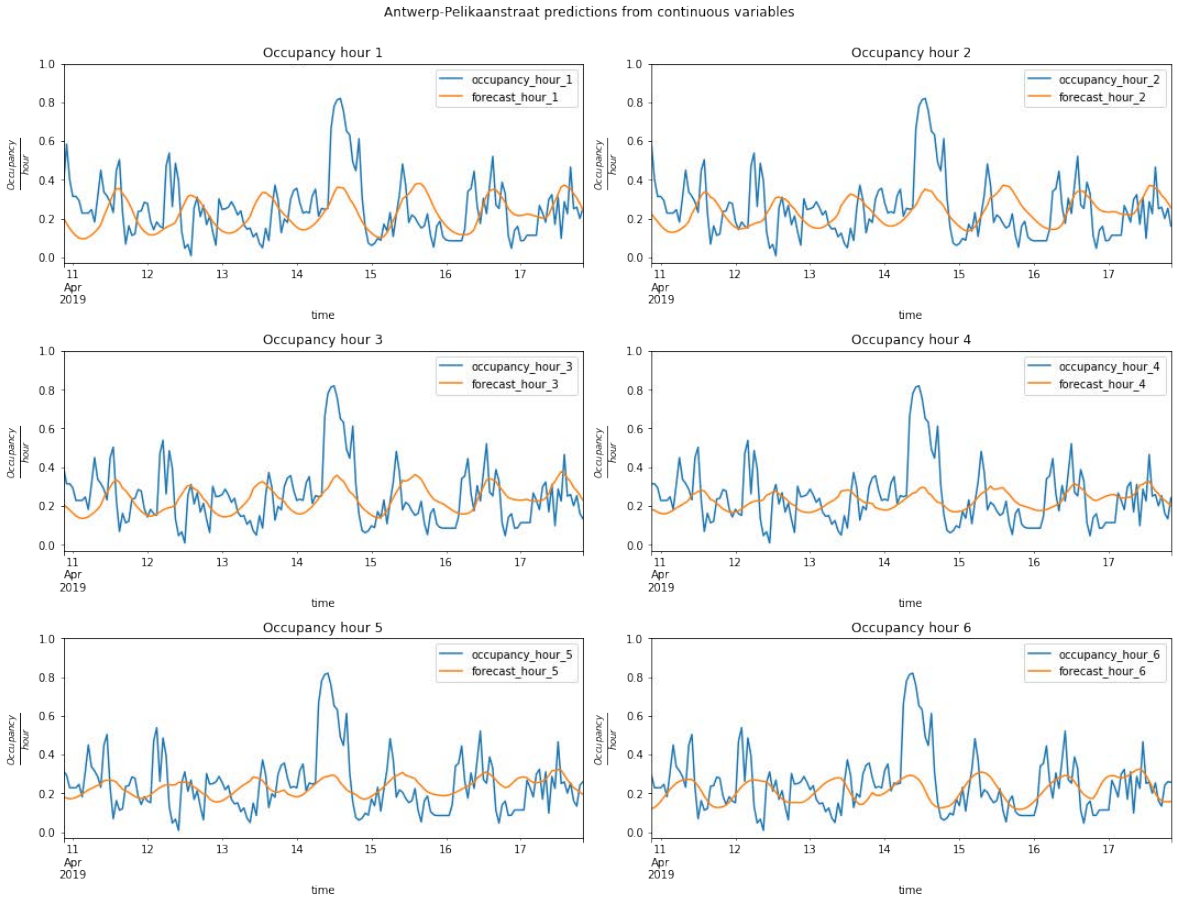


Figure 8.82: Antwerp forecasts with all the numerical variables.

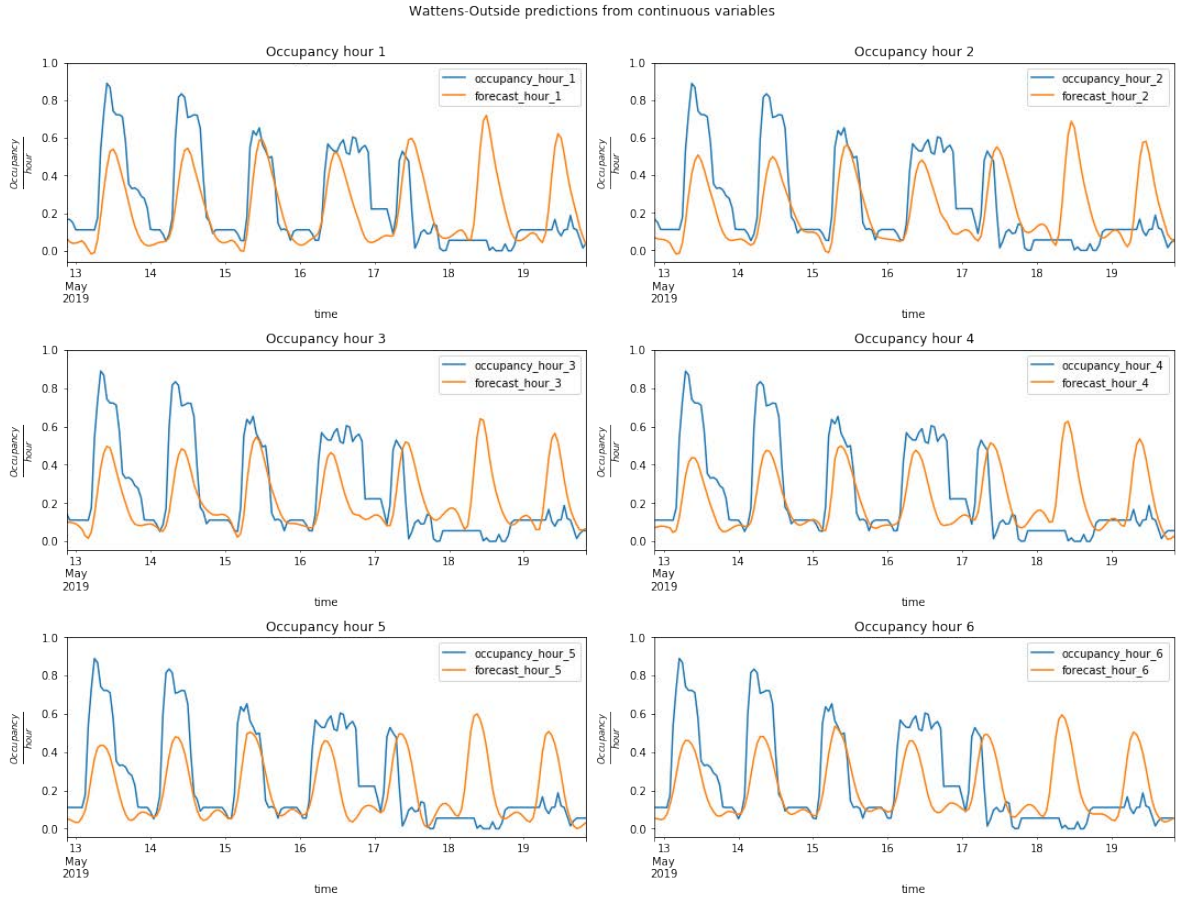


Figure 8.83: Wattens forecasts with all the numerical variables.

Forecasts for the Wattens scenario are presented in Figure 8.83. Similarly to the previous scenario, the forecasts show that the model knows that some relationship exists between the hours of the day and the parking occupancy, but it fails to recognize more complex patterns like those that distinguish weekends from weekdays.

Figure 8.84 presents the Los Angeles Standard sector's next six hours of parking occupancy forecasts, which were provided by a model trained with all exogenous continuous variables. The forecasts show that the models cannot make good predictions about the parking occupancy during the day, although it provides different values based on its inputs.

The last scenario is Sector-10 in Riyadh. The forecasts of the model trained with only exogenous continuous variables are presented in Figure 8.85. The regular patterns of parking occupancy make it easy for the model to learn the relationship between the inputs and the parking occupancies, while some different values among the days can be caused by weather information.

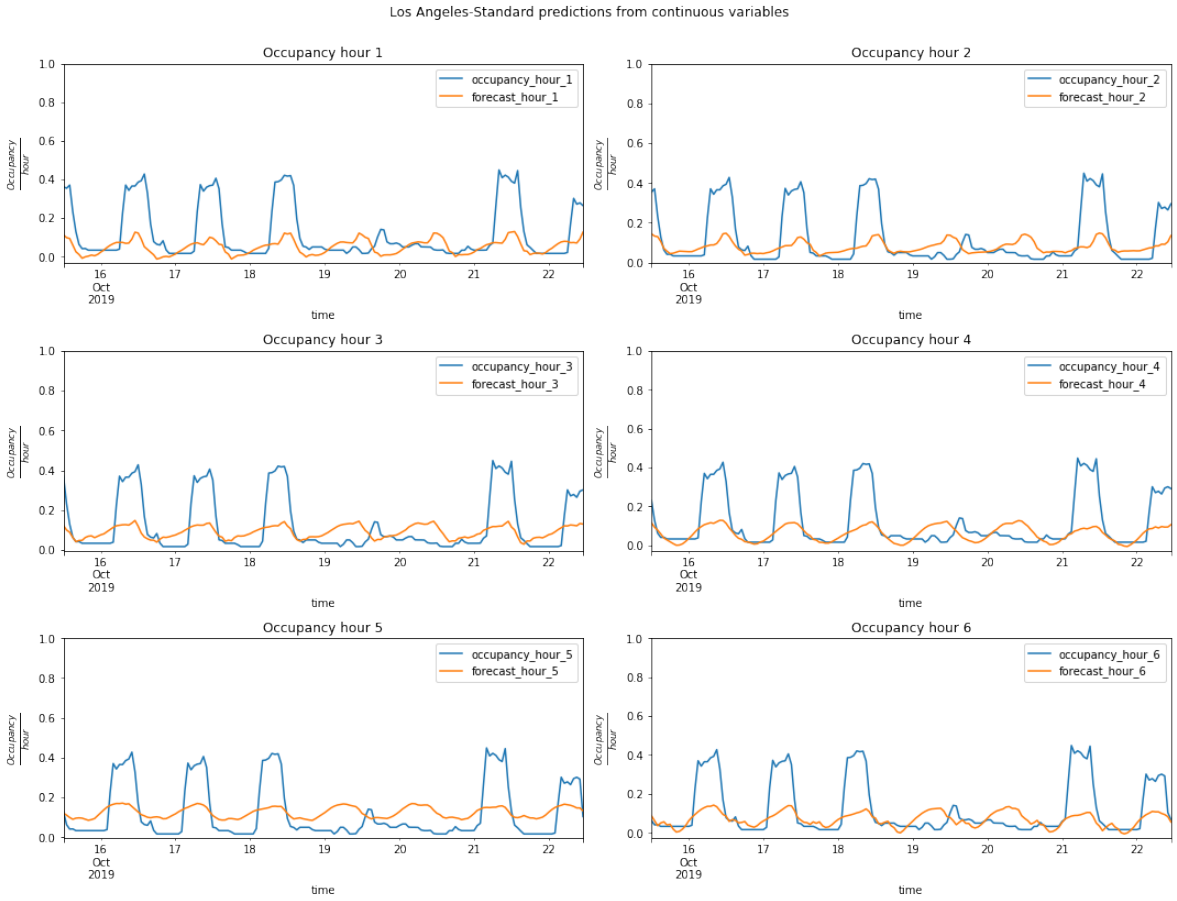


Figure 8.84: Los Angeles forecasts with all the numerical variables.

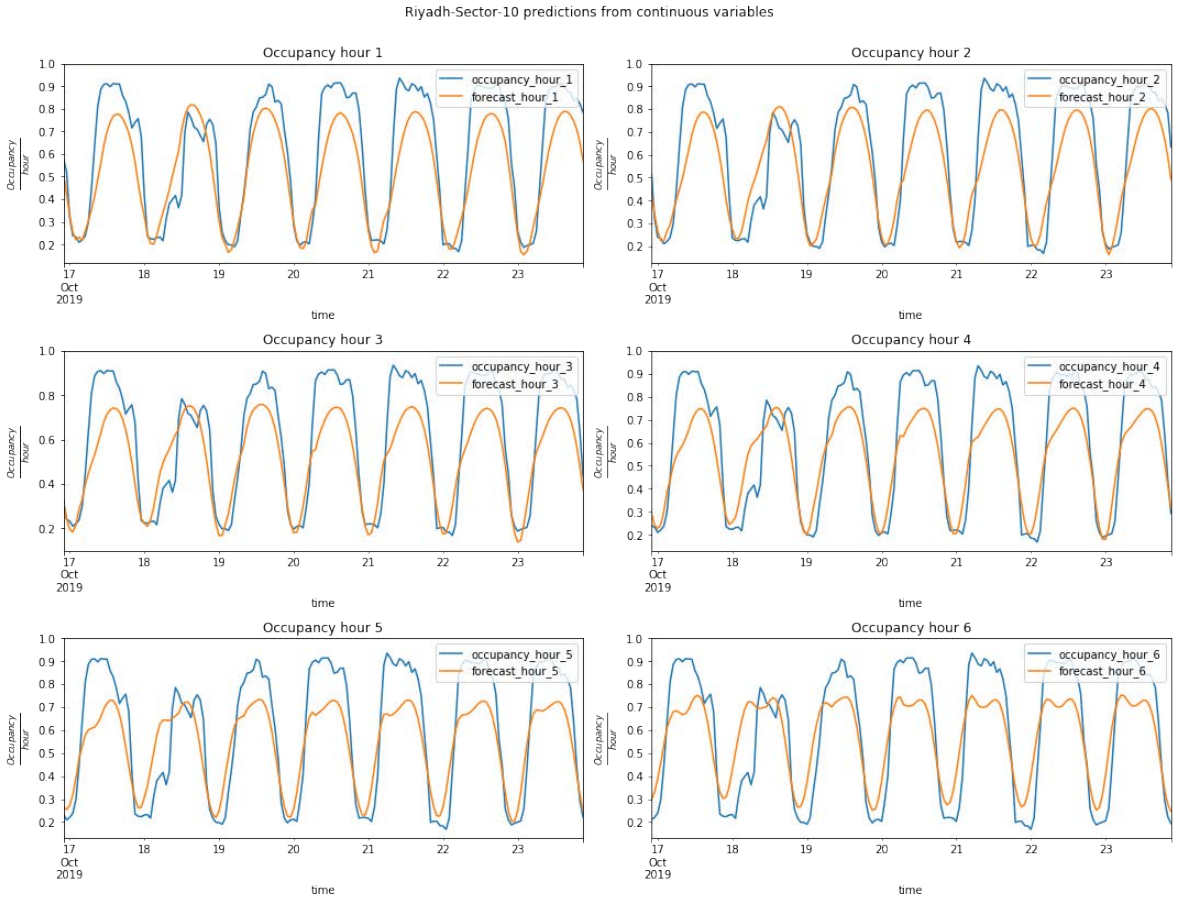


Figure 8.85: Riyadh forecasts with all the numerical variables.

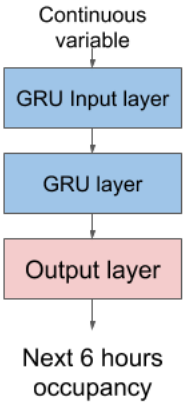


Figure 8.86: Architecture used for the models with only one numerical variable.

8.2.4 Numerical variables one by one

In order to gain insights into the relationship between the numerical exogenous variables and the parking occupancy, a model has been created for each of them in order to forecast the next six hours of parking occupancy.

The models consist of:

- Two GRU layers of 128 units each, followed by a dense layer with six neurons.
- Adam optimizer.
- 50 epochs with early stop.
- 128 as batch size.
- RMSE as loss function.

The architecture of the models with one numerical variable is presented in Figure 8.86. It is exactly the same as when all the numerical variables are used, with the exception that the input corresponds to sequences (time series) from just one feature.

Antwerp

The model using all continuous variables for the Antwerp scenario learns that a relationship exists between the hours of the day and the parking occupancy, but the values of the forecast were not good enough.

time	visibility	cloudCover	windBearing	uvIndex
2019-04-16 00:00:00	1.000000	0.04	0.267409	0.000
2019-04-16 01:00:00	1.000000	0.06	0.270195	0.000
2019-04-16 02:00:00	1.000000	0.09	0.261838	0.000
2019-04-16 03:00:00	1.000000	0.15	0.267409	0.000
2019-04-16 04:00:00	1.000000	0.23	0.267409	0.000
2019-04-16 05:00:00	1.000000	0.29	0.267409	0.000
2019-04-16 06:00:00	1.000000	0.30	0.267409	0.000
2019-04-16 07:00:00	1.000000	0.28	0.259053	0.000
2019-04-16 08:00:00	1.000000	0.31	0.247911	0.000
2019-04-16 09:00:00	1.000000	0.44	0.247911	0.125
2019-04-16 10:00:00	1.000000	0.61	0.256267	0.125
2019-04-16 11:00:00	1.000000	0.74	0.261838	0.250
2019-04-16 12:00:00	1.000000	0.79	0.278552	0.375
2019-04-16 13:00:00	1.000000	0.79	0.289694	0.375
2019-04-16 14:00:00	1.000000	0.81	0.295265	0.375
2019-04-16 15:00:00	1.000000	0.76	0.337047	0.375
2019-04-16 16:00:00	1.000000	0.81	0.376045	0.250
2019-04-16 17:00:00	1.000000	0.86	0.420613	0.250
2019-04-16 18:00:00	0.960412	1.00	0.462396	0.125
2019-04-16 19:00:00	0.504486	1.00	0.515320	0.000
2019-04-16 20:00:00	0.209988	1.00	0.520891	0.000
2019-04-16 21:00:00	0.220089	1.00	0.515320	0.000
2019-04-16 22:00:00	0.391179	1.00	0.509749	0.000
2019-04-16 23:00:00	0.561077	1.00	0.506964	0.000

Table 8.5: Antwerp Visibility, Cloud Cover, Wind Bearing and UV Index values for 16 April 2019.

As support for the analysis, Table 8.5 provides the numerical values for 16 April 2019.

Figure 8.88 presents the forecasts from the model trained with the Temperature variable. Similarly to when using UV Index and Humidity, the model uses the variable information as a proxy for the hours of the day. Therefore, it relates them to the parking occupancy but cannot discriminate days or learn other complex relationships.

The forecasts of the model trained with the exogenous variable Wind Gust are presented in Figure 8.89. Similarly to what happens with Wind Bearing, this variable's information makes sense when used with other variables related to wind. The forecasts of the model are more or less around the same value for the week of data visualized.

Table 8.6 provides auxiliary information about the values of the variables Wind Speed, Precipitation Accumulation and Dew Point.

The visualizations in Figure 8.90 present the six time step forecasts from the model

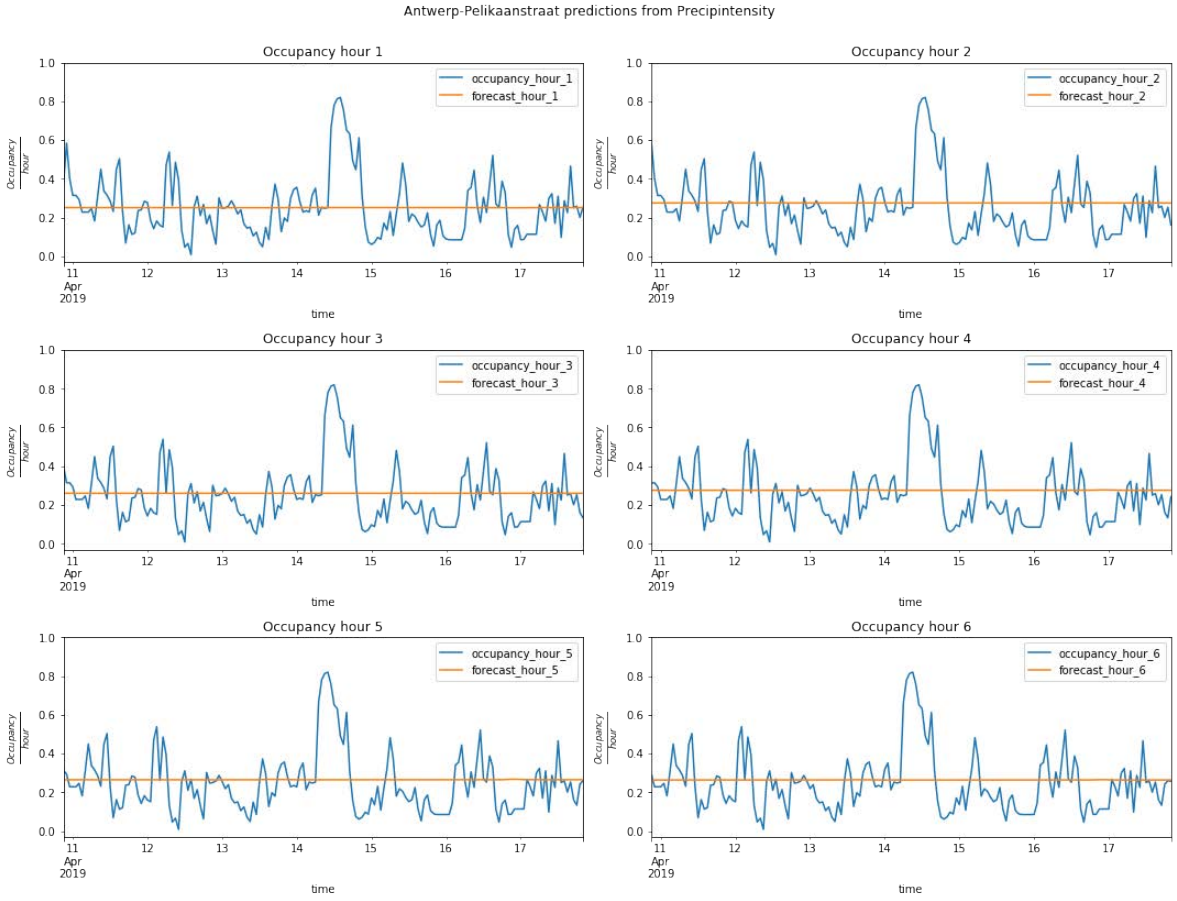


Figure 8.87: Six-hour parking occupancy forecasts from the model trained with Precipitation Intensity.

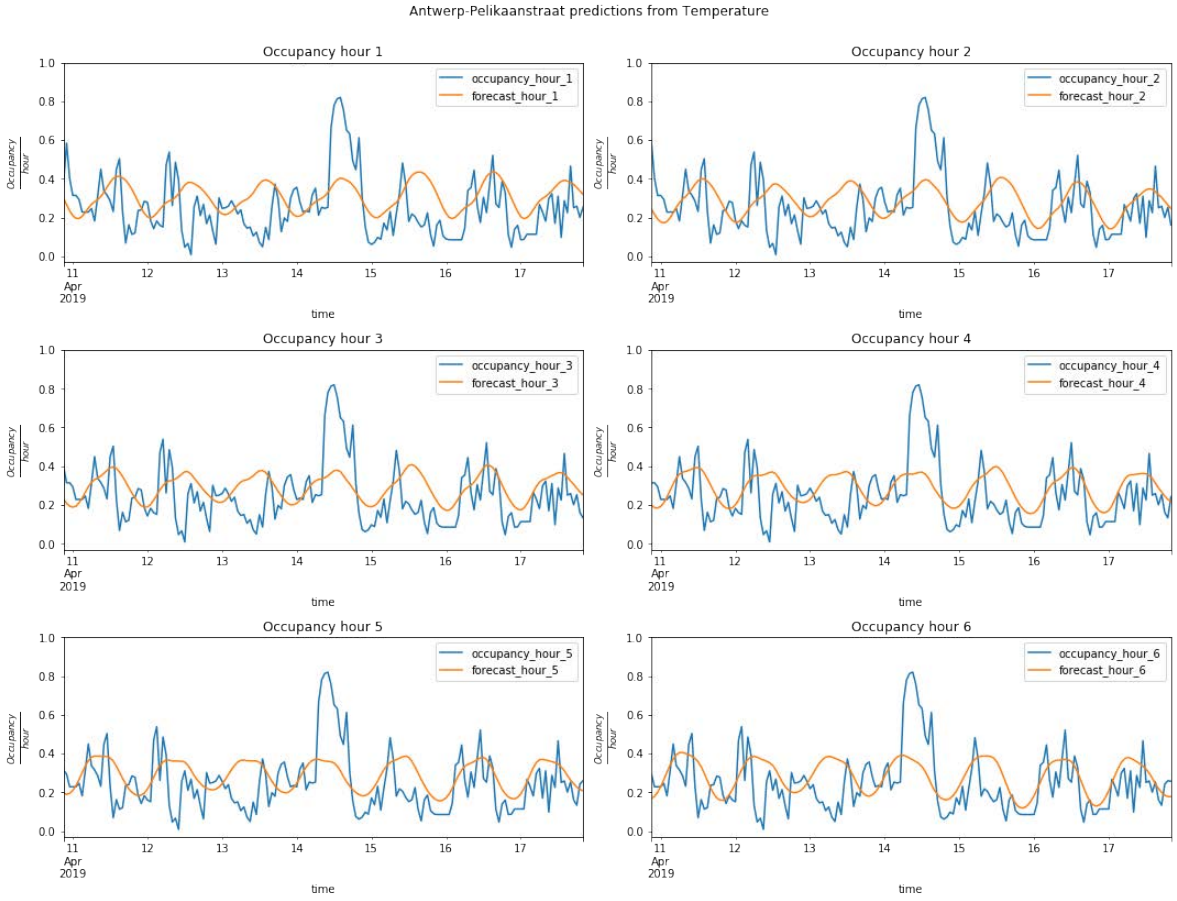


Figure 8.88: Antwerp scenario time series from parking occupancy (blue) and forecasts of the model trained with the Temperature variable (orange).

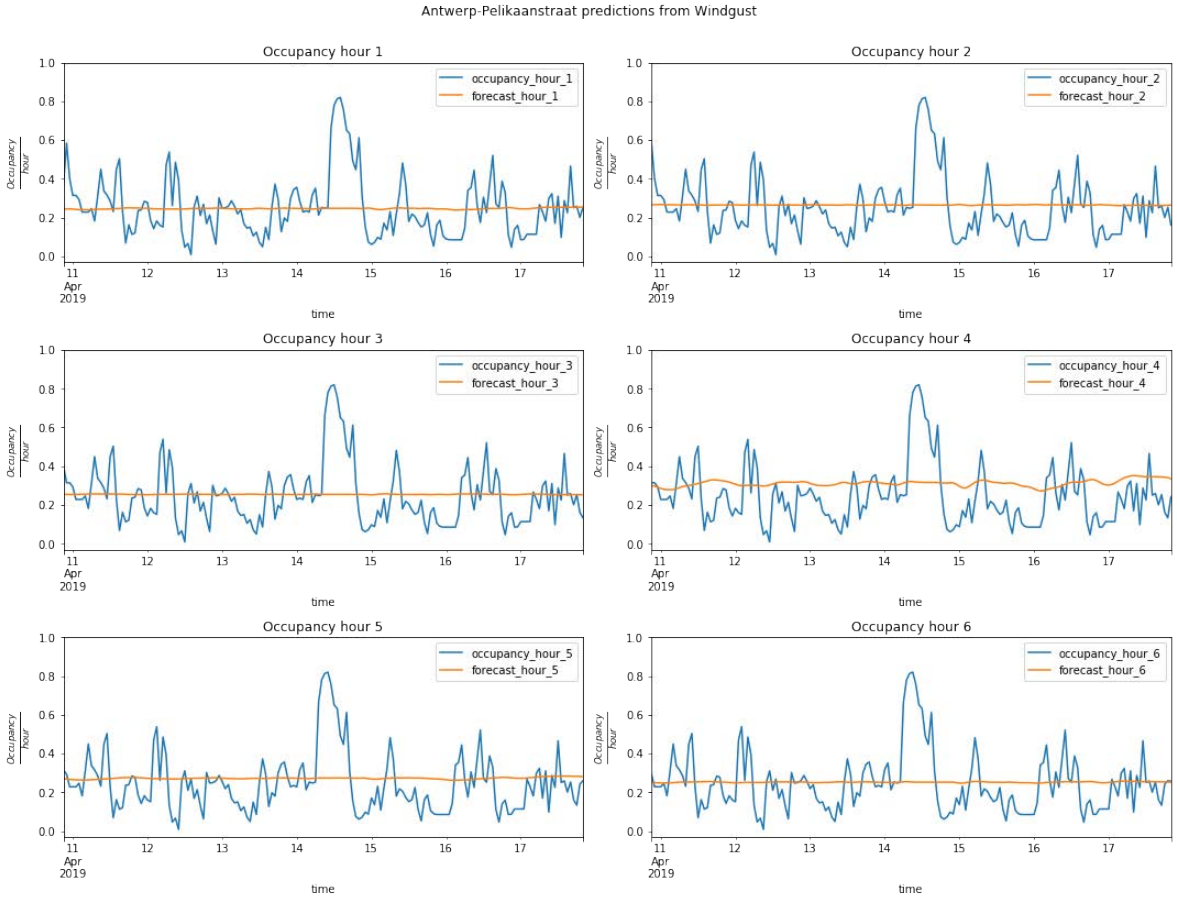


Figure 8.89: Time series visualizations of the forecasts from the model trained with the Wind Gust variable (orange) and the parking occupancy (blue).

time	dewPoint	precipAccumulation	windSpeed
2019-04-16 00:00:00	0.531883	0.0	0.336777
2019-04-16 01:00:00	0.520399	0.0	0.342975
2019-04-16 02:00:00	0.514053	0.0	0.310606
2019-04-16 03:00:00	0.498036	0.0	0.306474
2019-04-16 04:00:00	0.476881	0.0	0.293388
2019-04-16 05:00:00	0.465095	0.0	0.292011
2019-04-16 06:00:00	0.462375	0.0	0.284435
2019-04-16 07:00:00	0.455425	0.0	0.289256
2019-04-16 08:00:00	0.464491	0.0	0.267906
2019-04-16 09:00:00	0.492898	0.0	0.285813
2019-04-16 10:00:00	0.523723	0.0	0.280992
2019-04-16 11:00:00	0.561499	0.0	0.280303
2019-04-16 12:00:00	0.593835	0.0	0.272039
2019-04-16 13:00:00	0.607736	0.0	0.269972
2019-04-16 14:00:00	0.565428	0.0	0.230716
2019-04-16 15:00:00	0.572378	0.0	0.242424
2019-04-16 16:00:00	0.588395	0.0	0.225207
2019-04-16 17:00:00	0.602297	0.0	0.249311
2019-04-16 18:00:00	0.603808	0.0	0.245179
2019-04-16 19:00:00	0.619825	0.0	0.251377
2019-04-16 20:00:00	0.636144	0.0	0.253444
2019-04-16 21:00:00	0.652765	0.0	0.251377
2019-04-16 22:00:00	0.654276	0.0	0.235537
2019-04-16 23:00:00	0.652765	0.0	0.239669

Table 8.6: Antwerp Wind Speed, Precipitation Accumulation and Dew Point values for 16 April 2019.

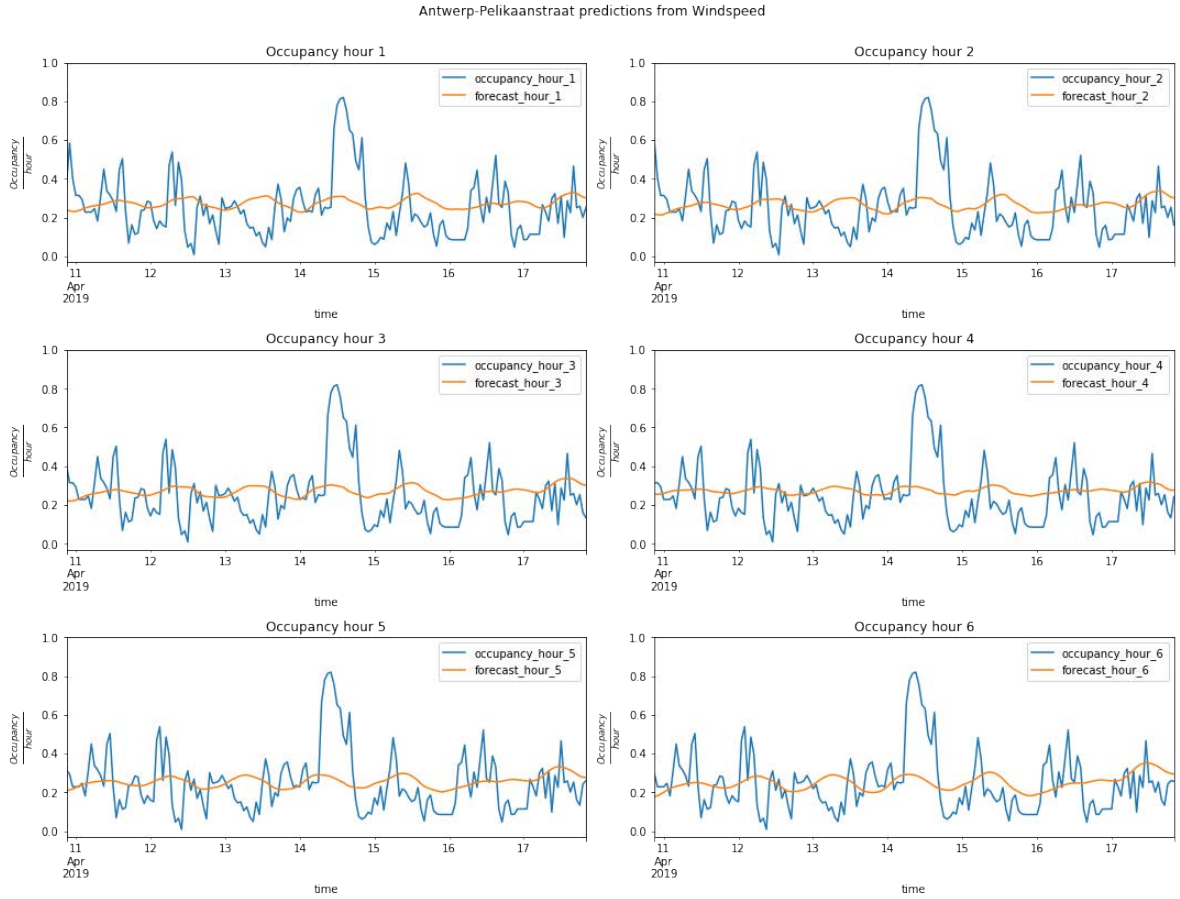


Figure 8.90: Forecasts (orange) and parking occupancy (blue) visualizations from the model trained with the variable Wind Speed.

trained with the Wind Speed variable. The variable offers information about the speed of the wind in miles per hour. Because there is a recurrent pattern during the days where Wind Speed is higher at night and lower during the day, the model uses this information to produce forecasts that follows the hours of the day.

The variable Precipitation Accumulation (related to accumulated snow) (forecasts shown in Figure 8.91) offers little information, as it is formed by very sparse time series.

Despite this, the model can use the information carried by the exogenous variable to eventually adjust the occupancy values, as can be seen in Figure 8.92, which is based on the information presented in Table 8.7. The visualizations correspond to a week with a huge amount of snow precipitation in Antwerp. More specifically, day 22 experienced a lot of snow accumulation. Therefore, it was not possible to travel by car. This anomaly is reflected as a huge occupancy level with low variability throughout the day. The model has learnt that, under certain conditions (high values for Precipitation Accumulation), the occupancy levels are higher than the normal levels.

Figure 8.93 presents the forecasts from the model trained with the exogenous variable

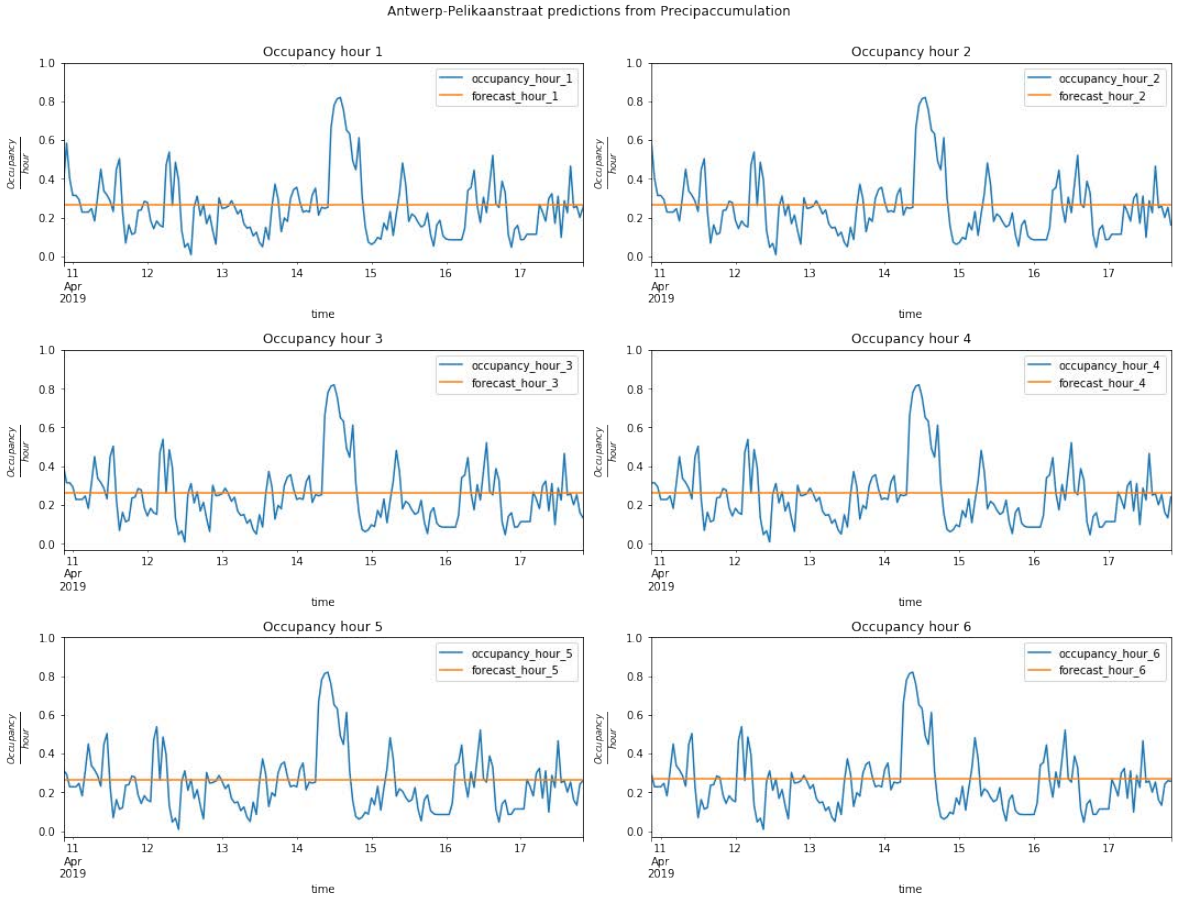


Figure 8.91: Pelikaanstraat (Antwerp) six-hour forecasts from the model trained using the continuous variable Precipitation Accumulation.

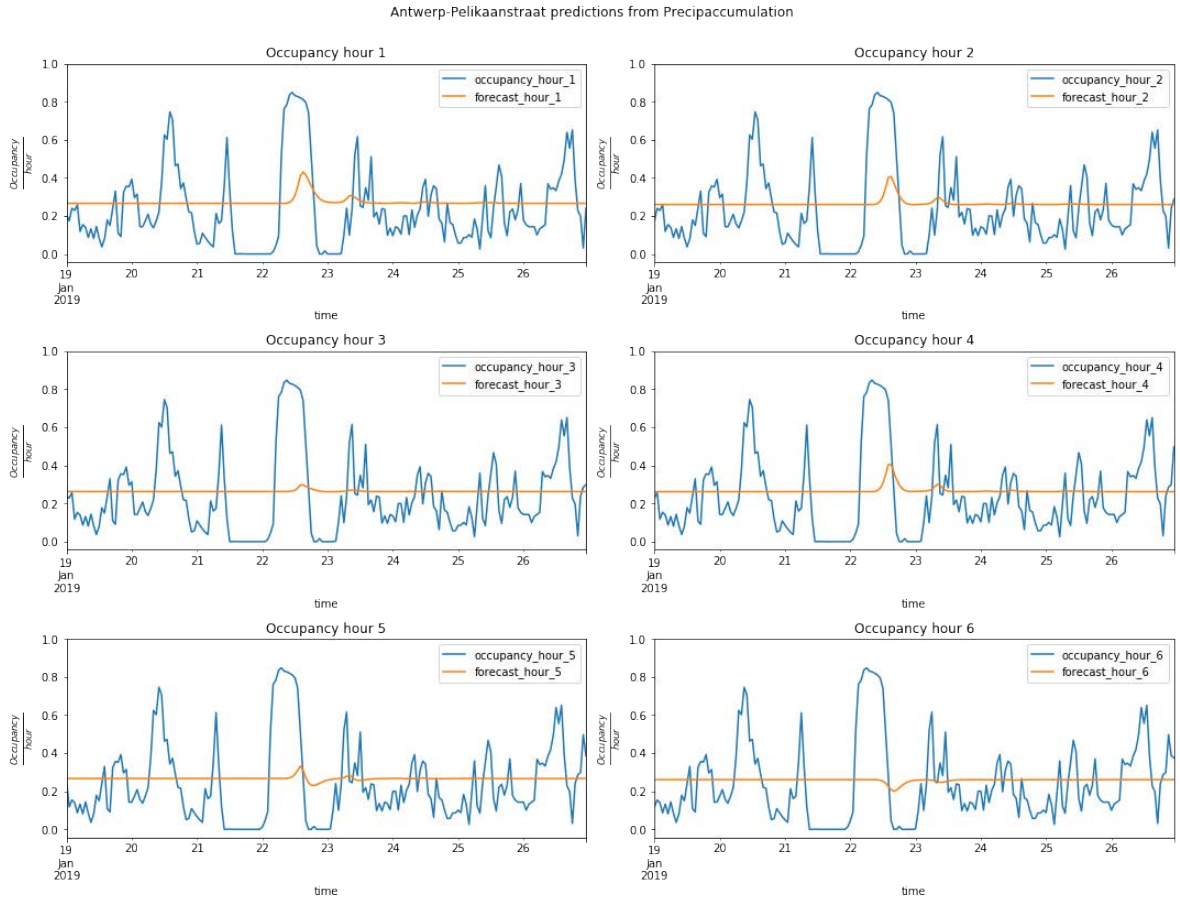


Figure 8.92: Pelikaanstraat (Antwerp) six-hour forecasts of the model trained using the continuous variable Precipitation Accumulation, focusing on snowy days (2019-01-20 to 2019-01-24).

time	precipAccumulation
2019-01-22 00:00:00	0.000000
2019-01-22 01:00:00	0.000000
2019-01-22 02:00:00	0.000000
2019-01-22 03:00:00	0.000000
2019-01-22 04:00:00	0.000000
2019-01-22 05:00:00	0.000000
2019-01-22 06:00:00	0.000000
2019-01-22 07:00:00	0.000000
2019-01-22 08:00:00	0.010651
2019-01-22 09:00:00	0.062591
2019-01-22 10:00:00	0.160928
2019-01-22 11:00:00	0.376131
2019-01-22 12:00:00	0.705282
2019-01-22 13:00:00	1.000000
2019-01-22 14:00:00	0.772833
2019-01-22 15:00:00	0.440473
2019-01-22 16:00:00	0.237963
2019-01-22 17:00:00	0.115115
2019-01-22 18:00:00	0.036475
2019-01-22 19:00:00	0.008900
2019-01-22 20:00:00	0.000000
2019-01-22 21:00:00	0.000000
2019-01-22 22:00:00	0.005398
2019-01-22 23:00:00	0.006274

Table 8.7: Antwerp Precipitation Accumulated values for the date 2019-01-22.

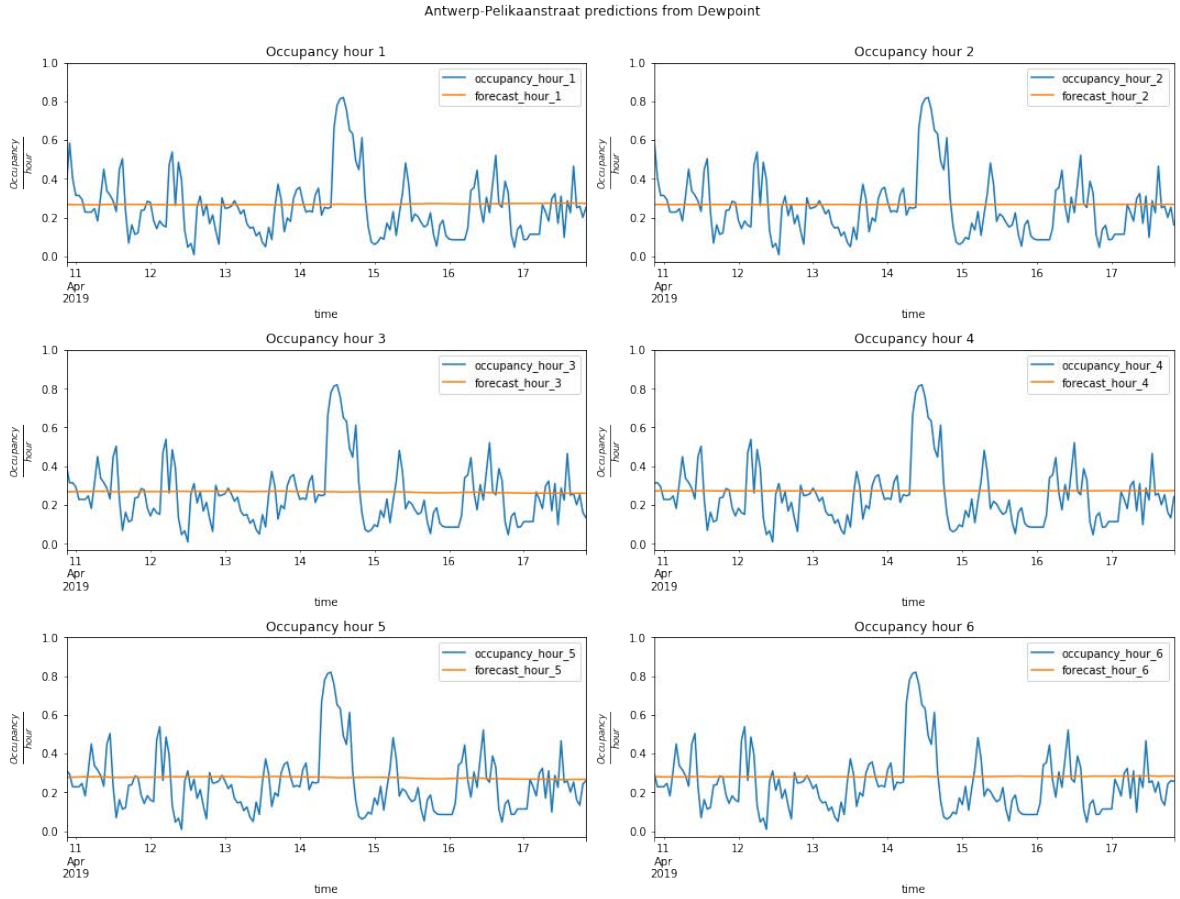


Figure 8.93: Pelikaanstraat (Antwerp) parking occupancy (blue) and forecasts from model trained with the exogenous variable Dew Point (orange).

Dew Point. The visualizations show that the model has not learnt any useful information from the input in order to produce forecasts.

From the models trained with each of the exogenous continuous variables for the Pelikaanstraat scenario, the following has been observed:

- The trained models use a subset of the exogenous variables as a proxy to map their values to parking occupancy, thus exploiting the fact that parking occupancy is a consequence of human activity. Those exogenous variables carry information that is both related to and a consequence of sunshine hours. The exogenous variables in which the models show this behaviour are: UV Index, Humidity and Temperature (perhaps also Wind Speed). The information from these variables could be redundant and can add noise to the models.
- A subset of the exogenous variables provides auxiliary information that can help the model improve the forecasts in isolated moments. Those variables are Precipitation Accumulation, Precipitation Intensity and Wind Gust (perhaps also Wind Speed).

time	visibility	cloudCover	windBearing	uvIndex
2019-05-14 00:00:00	0.685950	0.68	0.103064	0.000000
2019-05-14 01:00:00	0.759647	0.64	0.128134	0.000000
2019-05-14 02:00:00	0.702355	0.58	0.181058	0.000000
2019-05-14 03:00:00	0.869384	0.49	0.211699	0.000000
2019-05-14 04:00:00	1.000000	0.38	0.256267	0.000000
2019-05-14 05:00:00	1.000000	0.30	0.270195	0.000000
2019-05-14 06:00:00	1.000000	0.29	0.183844	0.000000
2019-05-14 07:00:00	0.875536	0.31	0.097493	0.000000
2019-05-14 08:00:00 1	0.755670	0.32	0.052925	0.11111
2019-05-14 09:00:00	0.900081	0.33	0.038997	0.222222
2019-05-14 10:00:00	1.000000	0.34	0.038997	0.333333
2019-05-14 11:00:00	1.000000	0.36	0.038997	0.444444
2019-05-14 12:00:00	1.000000	0.40	0.030641	0.555556
2019-05-14 13:00:00	0.671286	0.45	0.019499	0.555556
2019-05-14 14:00:00	0.185422	0.53	0.863510	0.555556
2019-05-14 15:00:00	0.088113	0.67	0.899721	0.444444
2019-05-14 16:00:00	0.100354	0.84	0.949861	0.333333
2019-05-14 17:00:00	0.117070	0.96	0.994429	0.222222
2019-05-14 18:00:00	0.102343	0.98	0.997214	0.111111
2019-05-14 19:00:00	0.091965	0.97	0.994429	0.000000
2019-05-14 20:00:00	0.140869	0.97	0.963788	0.000000
2019-05-14 21:00:00	0.120549	0.98	0.974930	0.000000
2019-05-14 22:00:00	0.090226	0.98	0.983287	0.000000
2019-05-14 23:00:00	0.076431	0.99	0.986072	0.000000

Table 8.8: Wattens Visibility, Cloud Cover, Wind Bearing and UV Index values for 14 May 2019.

- Some exogenous variables do not offer any useful information, namely Dew Point, Wind Bearing and Visibility.

Wattens

As support for the analysis, Table 8.8 provides the numerical values for 14 May 2019.

Figure 8.94 presents the forecasts from the model trained with the Visibility variable. The visualizations show that the model has not learnt any useful information for forecasting parking occupancy.

Figure 8.95 presents the forecasts from the model trained with the exogenous numerical variable Cloud Cover. The visualizations show that the model change its forecasts using the variable's information and by carefully inspecting the input and forecast. However, it is hard to discern any pattern.

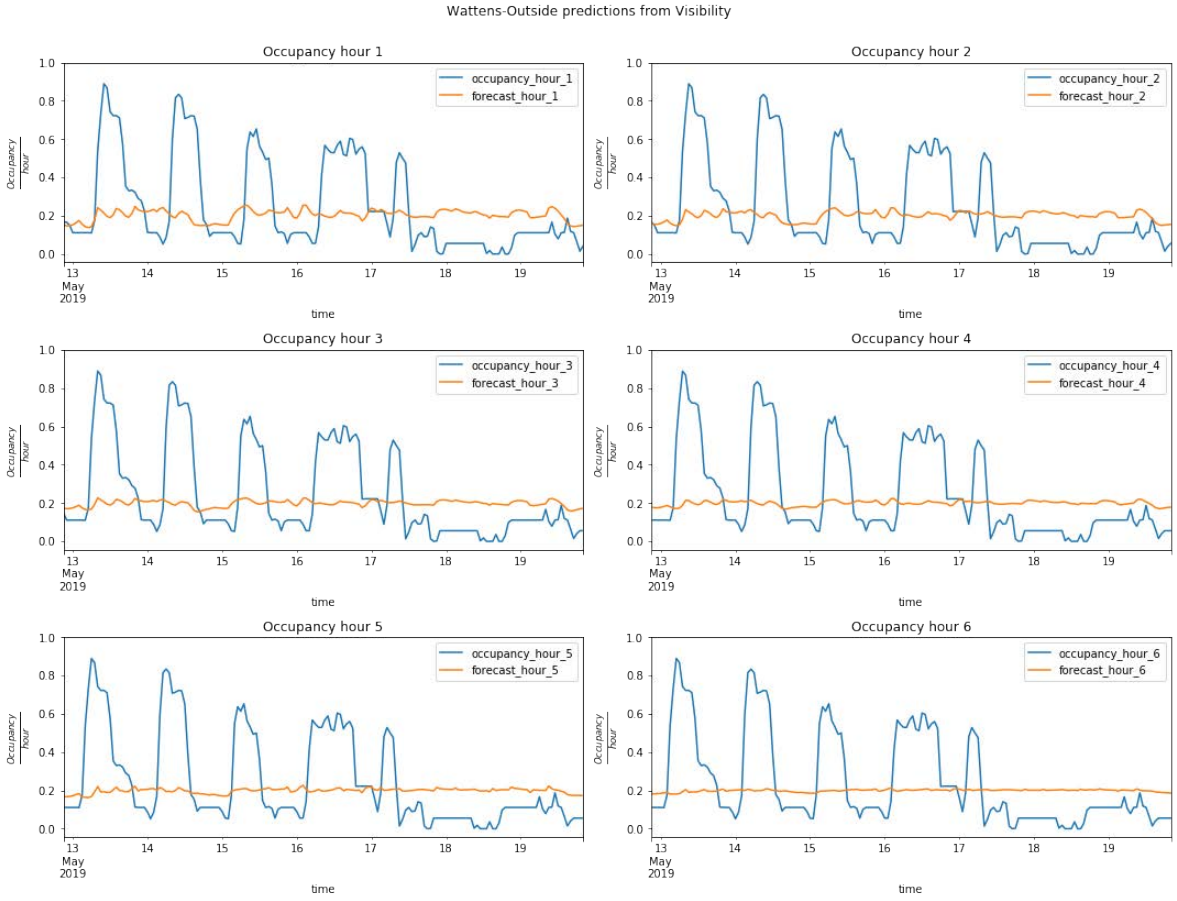


Figure 8.94: Outside sector in Wattens forecast visualizations from a model trained using the continuous variable Visibility.

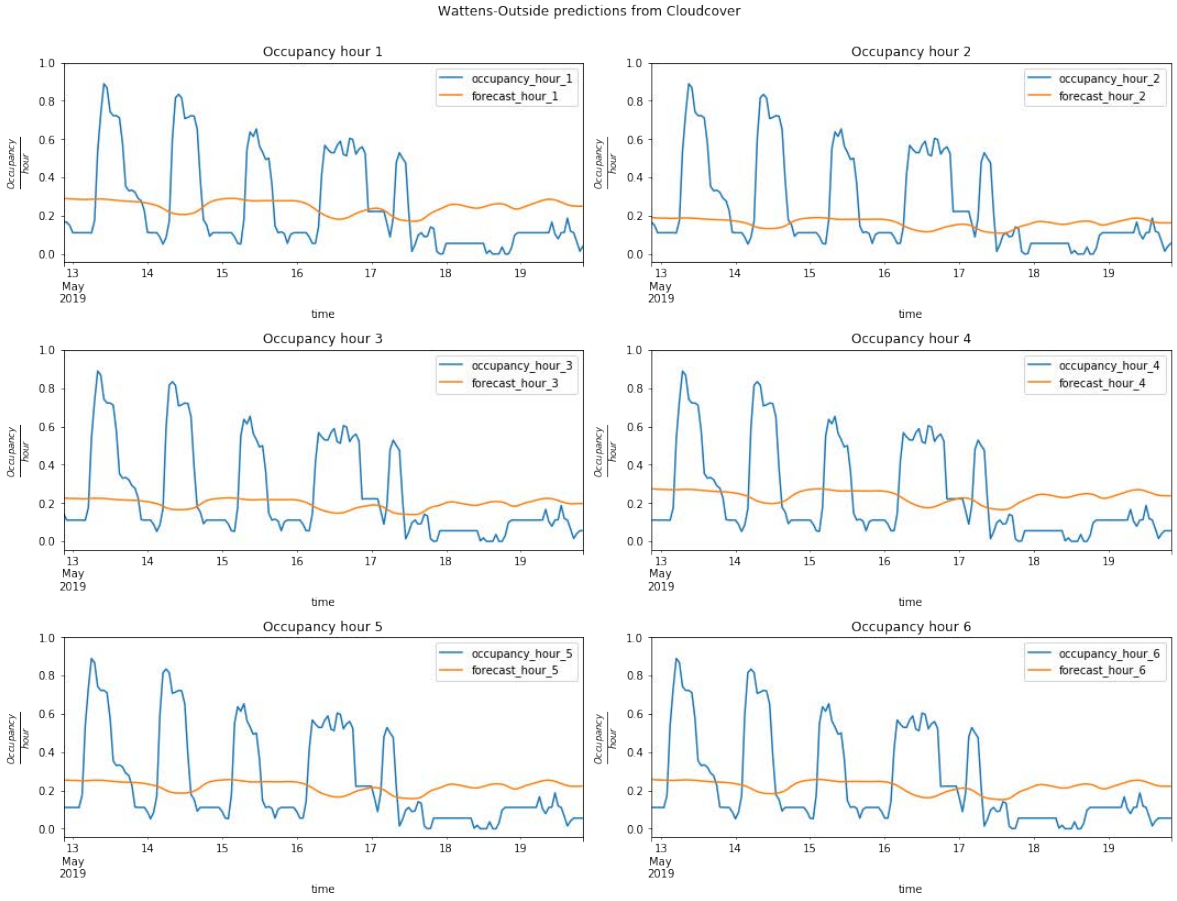


Figure 8.95: Outside sector parking occupancy (blue) and forecasts (orange) from a model trained using the continuous variable Cloud Cover.

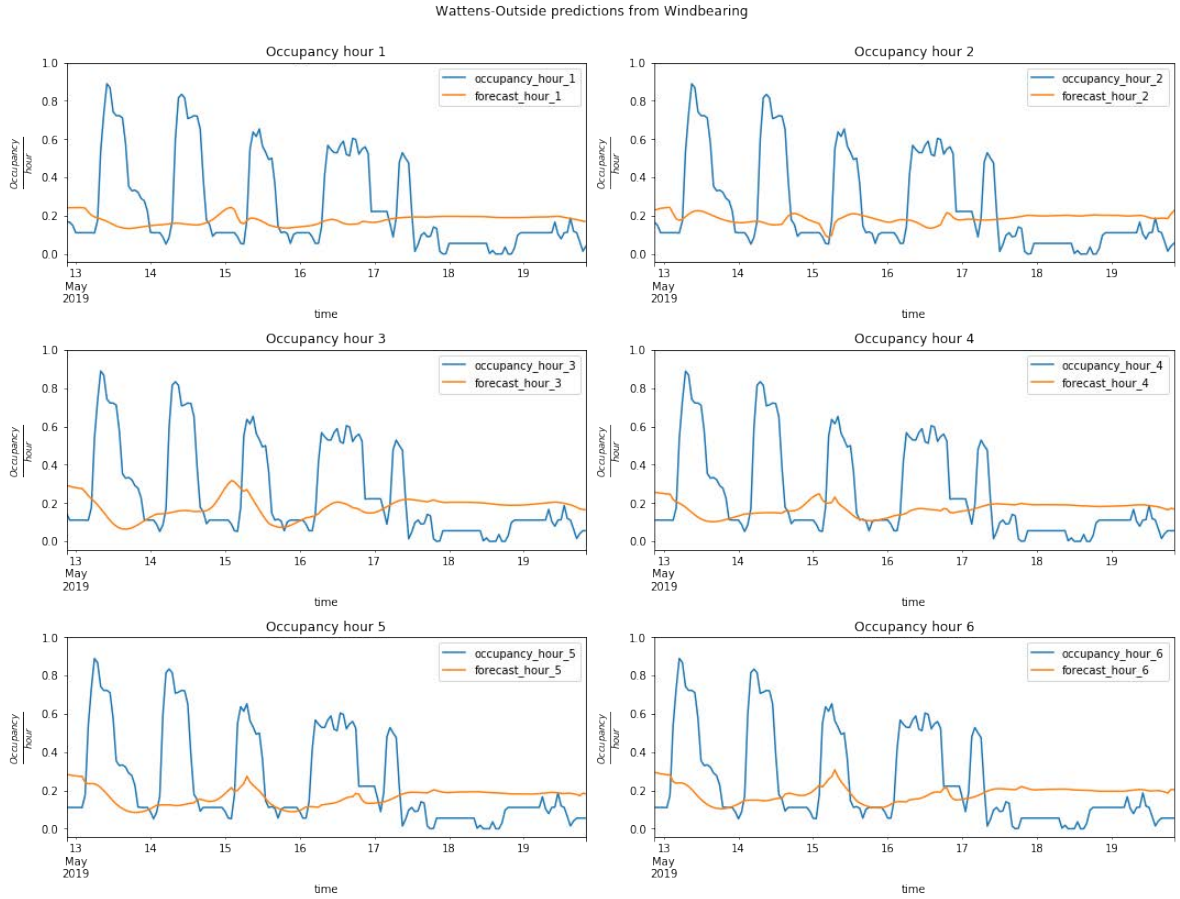


Figure 8.96: Real parking occupancy (blue) and model forecasts (orange) for the Outside sector in Wattens. The model has been trained using the exogenous variable Wind Bearing.

Figure 8.96 presents the forecasts from the model trained with the Wind Bearing variable. The model fails to provide correct forecast values when using the Wind Bearing information.

Figure 8.97 presents the forecasts from the model trained with the UV Index variable. Similarly to what was observed in Antwerp, the model learns from the variable which are the sunshine hours of the day and, based on that, it provides the occupancy levels. This information can be obtained from better sources like the categorical variable Day of the Week, as it also allows learning behaviours on special days (Friday, in the case of Wattens) and weekend patterns.

Information about the next numerical variables (Humidity, Precipitation Intensity, Temperature and Wind Gust) is offered in Table 8.9 as support for the text.

The visualizations in Figure 8.98 present the six time step forecasts from the model trained with the Humidity variable. The interpretation of the visualizations is the same as in the Antwerp scenario, as the model uses the humidity values throughout the day as a

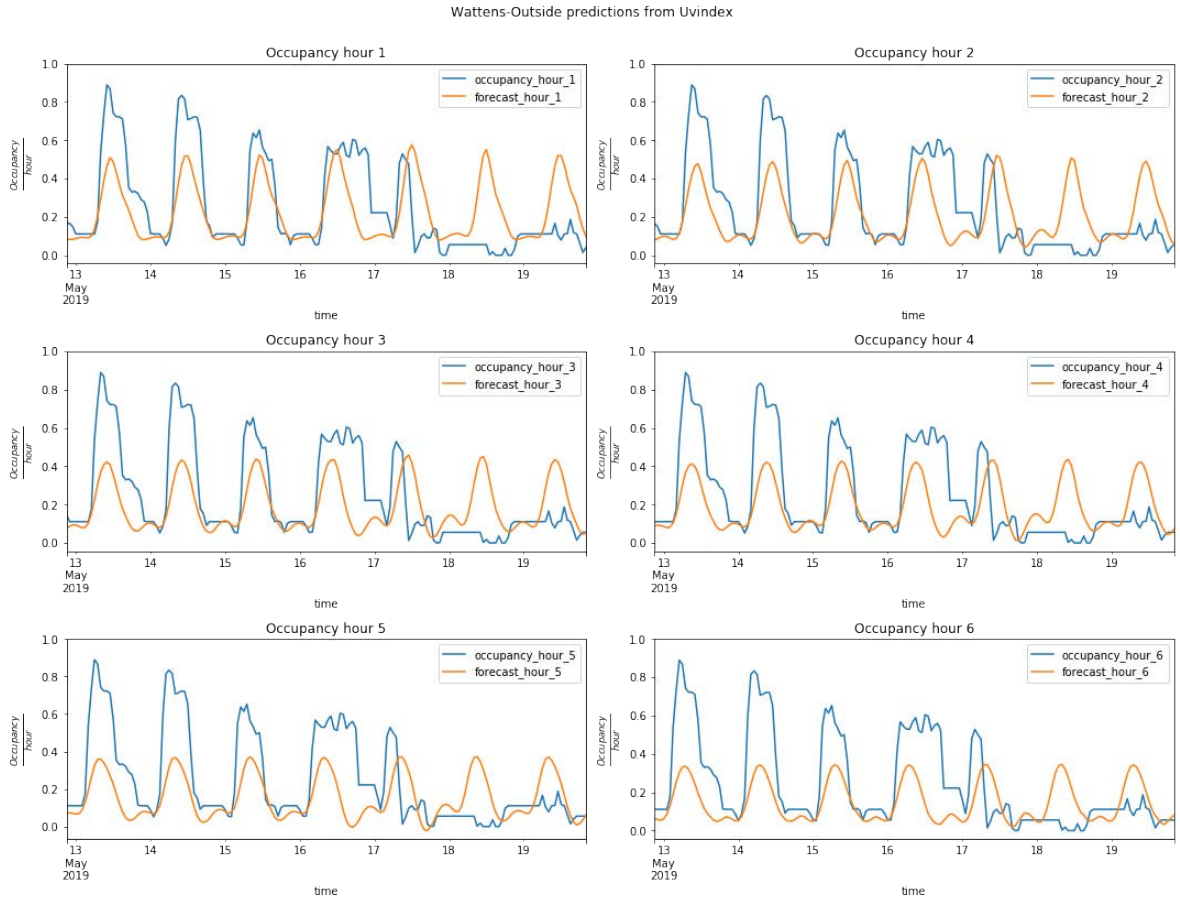


Figure 8.97: Real parking occupancy (blue) and model forecasts (orange) for the Outside sector in Wattens. The model has been trained using the exogenous variable UV Index.

time	humidity	precipIntensity	temperature	windGust
2019-05-15 00:00:00	0.722892	0.041774	0.281328	0.135561
2019-05-15 01:00:00	0.722892	0.040495	0.282465	0.139379
2019-05-15 02:00:00	0.698795	0.038515	0.291790	0.187589
2019-05-15 03:00:00	0.698795	0.034089	0.297930	0.183771
2019-05-15 04:00:00	0.710843 8	0.029211	0.300205	0.17183
2019-05-15 05:00:00	0.843373	0.0248433	0.257676	0.16372
2019-05-15 06:00:00	0.843373	0.020293	0.259950	0.165155
2019-05-15 07:00:00	0.819277	0.016366	0.269047	0.169928
2019-05-15 08:00:00	0.783133	0.013819 0	.282010	0.164678
2019-05-15 09:00:00	0.746988	0.012891	0.299068	0.173270
2019-05-15 10:00:00	0.710843	0.013072	0.319309	0.185203
2019-05-15 11:00:00	0.650602	0.012042	0.337958	0.192840
2019-05-15 12:00:00	0.614458	0.008477	0.362065	0.192363
2019-05-15 13:00:00	0.590361	0.005783	0.385718	0.188067
2019-05-15 14:00:00	0.602410	0.0048103	0.358199	0.17804
2019-05-15 15:00:00	0.530120	0.003814	0.398681	0.167542
2019-05-15 16:00:00	0.493976	0.002829	0.438026	0.154177
2019-05-15 17:00:00	0.626506 3	0.002456	0.384808	0.14224
2019-05-15 18:00:00	0.626506	0.002479	0.379577	0.134129
2019-05-15 19:00:00	0.626506	0.003169 9	0.371162	0.12696
2019-05-15 20:00:00	0.626506	0.003622	0.357744	0.075418
2019-05-15 21:00:00	0.650602	0.002705	0.345690	0.054415
2019-05-15 22:00:00	0.674699	0.001709	0.331817	0.035800
2019-05-15 23:00:00	0.710843	0.001245	0.315897	0.025298

Table 8.9: Wattens Humidity, Precipitation Intensity, Temperature and Wind Gust values for 15 May 2019

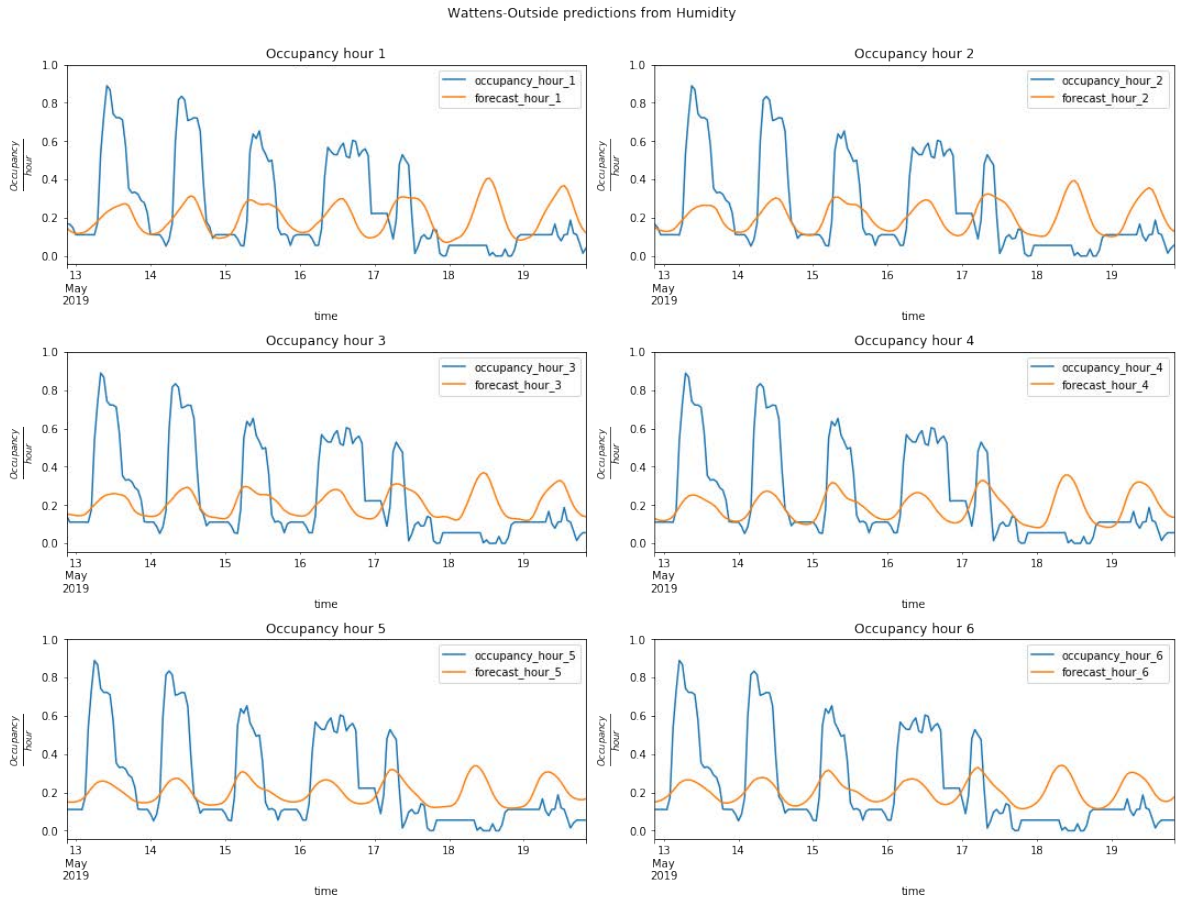


Figure 8.98: Real parking occupancy (blue) and model forecasts (orange) for the Outside sector in Wattens. The model has been trained using the exogenous variable Humidity.

proxy variable to learn sunshine hours, which it then uses to predict parking occupancies. Because of this, the model fails to predict the occupancy levels for Fridays and weekends.

Figure 8.99 presents the forecasts from the model trained with the exogenous numerical variable Precipitation Intensity. As commented on in the Antwerp scenario, this variable could be useful for adjusting some occasional forecasts. The visualizations show that the model forecasts lower occupancy levels when the Precipitation Intensity value increases, meaning that there could be a relationship.

Figure 8.100 presents the forecasts from the model trained with the Temperature variable. As commented on in the Antwerp scenario, the model uses the variable information to learn the relationship between sunshine hours and parking occupancy. Despite this, the model's Wattens forecasts seem to capture some more complex patterns, as the parking occupancy for day 15 is different when compared to the other days. Furthermore, the model's forecasts are different, which could mean that the lower temperatures on that day are interpreted by the model as lower parking occupancies (lower temperatures

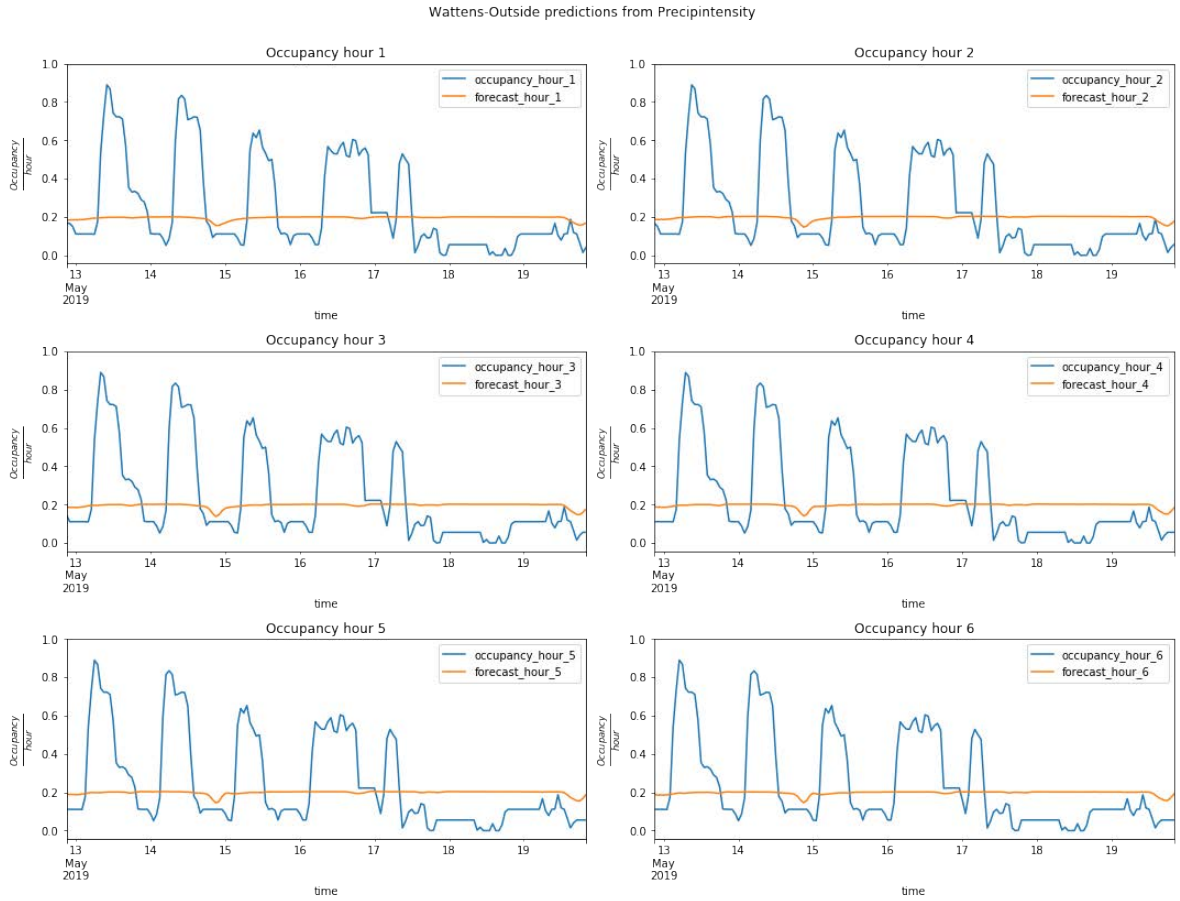


Figure 8.99: Time series visualizations of the real parking occupancy from Outside sector (Wattens) (blue) and forecasts (orange). The forecasts are provided by a model trained with the exogenous variable Precipitation Intensity.

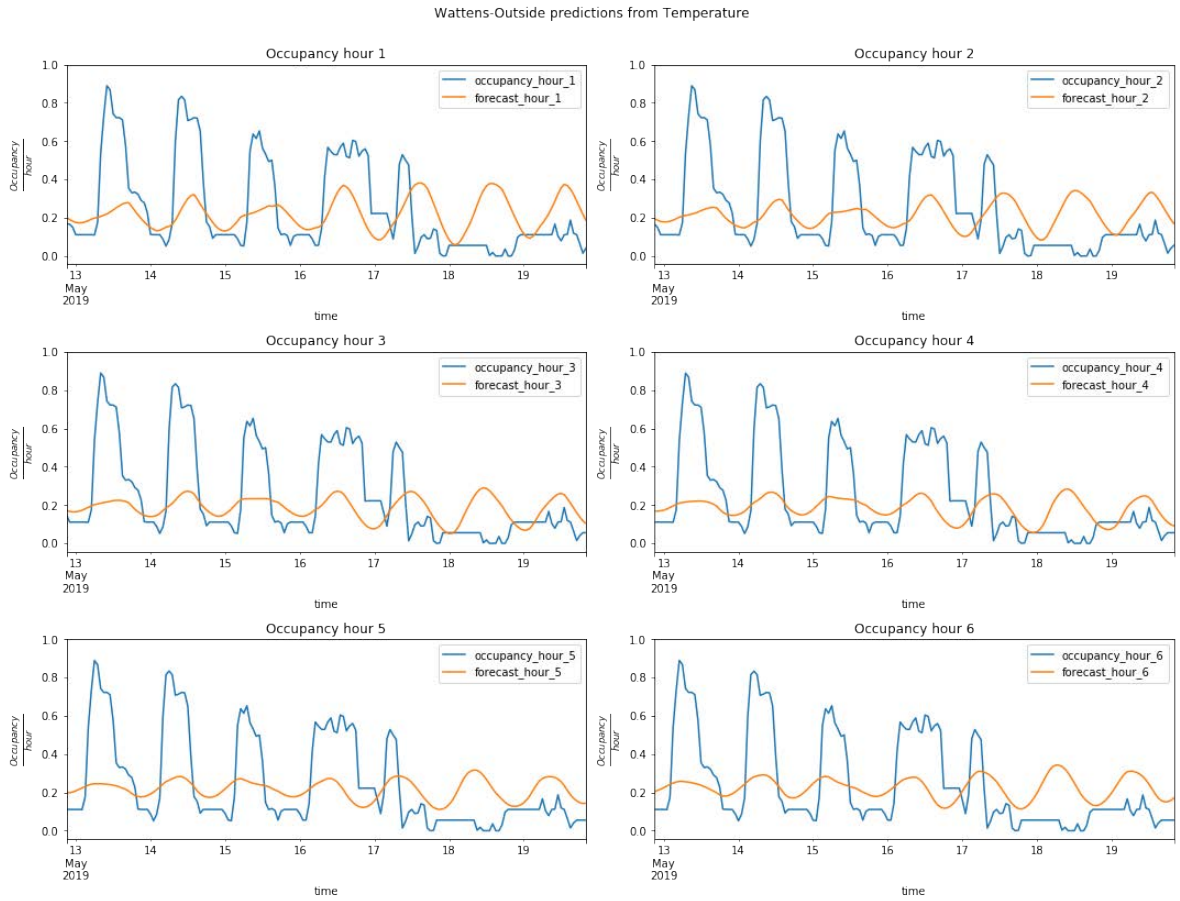


Figure 8.100: Time series visualizations of the parking occupancy (blue) and forecasts (orange). The forecasts are provided by a model trained with the exogenous variable Temperature.

were caused by precipitation). The problem is that temperature depends on the season and therefore this variable alone is not enough for providing solid forecasts, although it can help improve them.

The forecasts of the model trained with the exogenous variable Wind Gust are presented in Figure 8.101. The visualizations show that the model does not learn any useful information from the values of Wind Gust, although it is possible that it provides some useful information about weather intensity for occasional hours.

Table 8.10 provides auxiliary information about the values of the variables Wind Speed, Precipitation Accumulation and Dew Point.

The visualizations in Figure 8.102 present the six time step forecasts from the model trained with the Wind Speed variable. The information carried by this exogenous variable in the Wattens scenario seems to be insufficient for producing forecasts (remember that Wind Speed in the Antwerp scenario provided some information correlated with the sunshine hours, which the model learnt in order to produce forecasts). It is possible

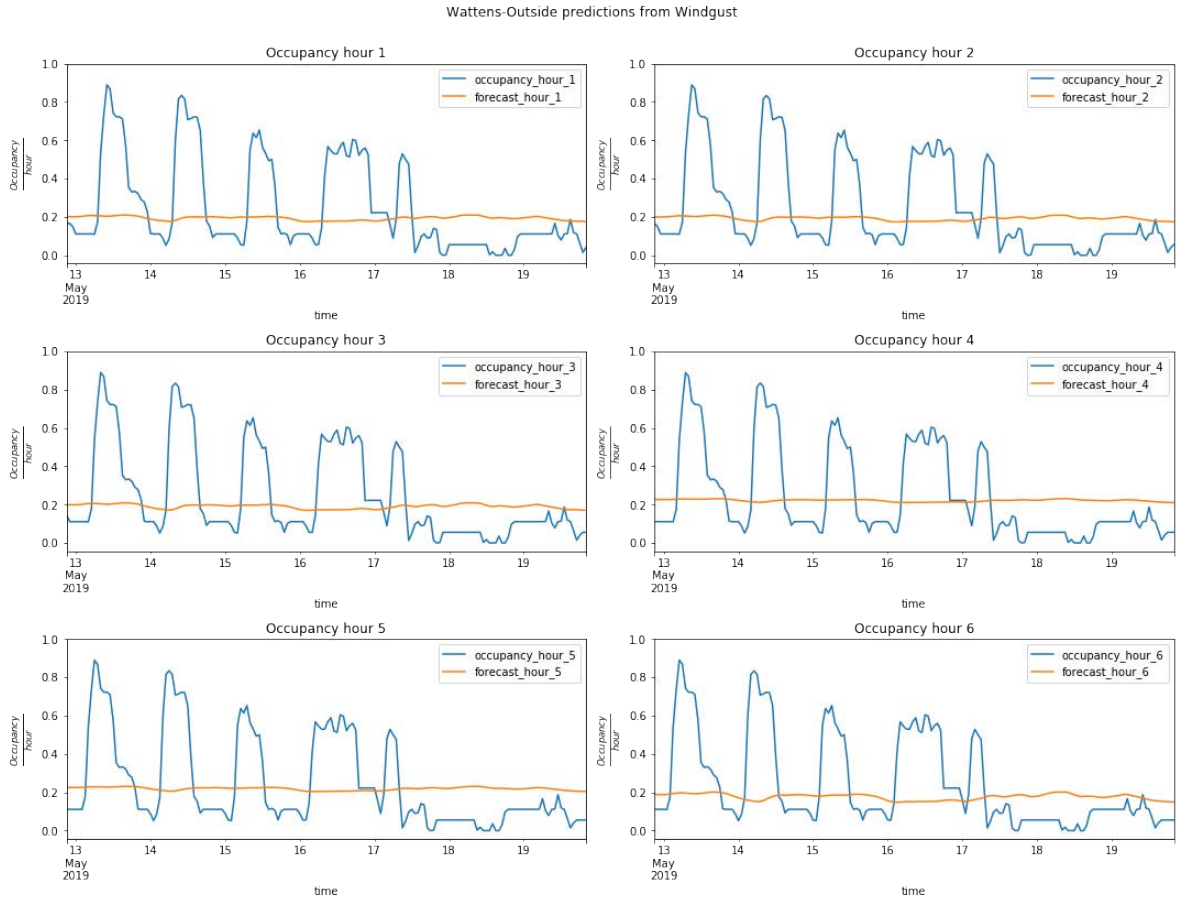


Figure 8.101: Time series visualizations of the parking occupancy (blue) and forecasts (orange) for the Outside sector in Wattens. The forecasts are provided by a model trained with the exogenous variable Wind Gust.

time	windSpeed	precipAccumulation	dewPoint
2019-05-15 00:00:00	0.091044	0.0	0.459580
2019-05-15 01:00:00	0.088823	0.0	0.458005
2019-05-15 02:00:00	0.080681	0.0	0.460367
2019-05-15 03:00:00	0.084382	0.0	0.465092
2019-05-15 04:00:00	0.094745	0.0	0.472178
2019-05-15 05:00:00	0.100666	0.0	0.475591
2019-05-15 06:00:00	0.100666	0.0	0.476903
2019-05-15 07:00:00	0.100666	0.0	0.477165
2019-05-15 08:00:00	0.099186	0.0	0.481102
2019-05-15 09:00:00	0.107328	0.0	0.487402
2019-05-15 10:00:00	0.119911	0.0	0.495801
2019-05-15 11:00:00	0.128793	0.0	0.493701
2019-05-15 12:00:00	0.131754	0.0	0.504199
2019-05-15 13:00:00	0.132494	0.0	0.514961
2019-05-15 14:00:00	0.142117	0.0	0.491601
2019-05-15 15:00:00	0.135455	0.0	0.503150
2019-05-15 16:00:00	0.124352	0.0	0.525722
2019-05-15 17:00:00	0.113990	0.0	0.531759
2019-05-15 18:00:00	0.105848	0.0	0.529134
2019-05-15 19:00:00	0.096965	0.0	0.516535
2019-05-15 20:00:00	0.070318	0.0	0.503412
2019-05-15 21:00:00	0.059215	0.0	0.498425
2019-05-15 22:00:00	0.051073	0.0	0.495276
2019-05-15 23:00:00	0.045152	0.0	0.489764

Table 8.10: Wattens Wind Speed, Precipitation Accumulation and Dew Point values for 17 May 2019.

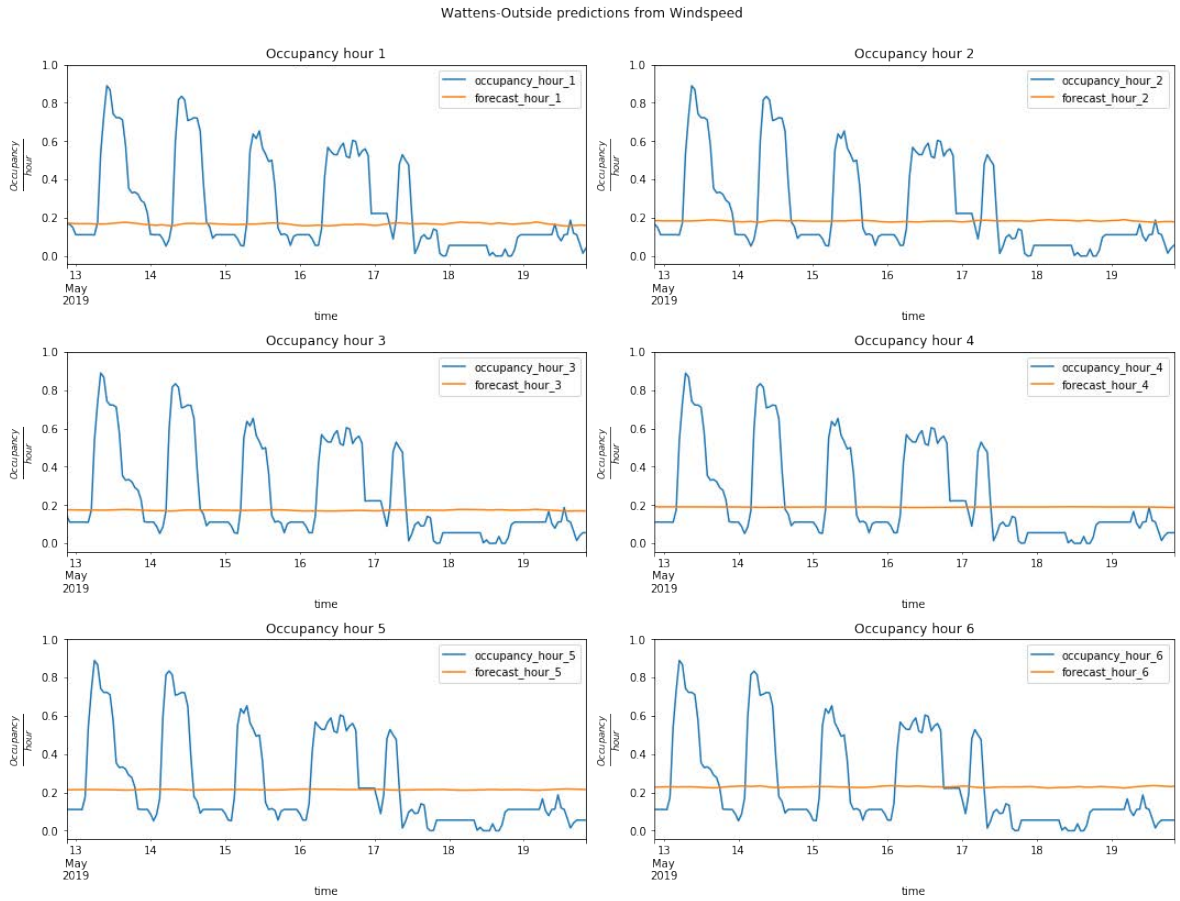


Figure 8.102: Outside sector (Wattens) parking occupancy (blue) and forecasts from a model trained with the exogenous variable Wind Speed (orange).

that this variable could offer some occasional information when severe weather happens.

Forecasts of the variable Precipitation Accumulation are shown in Figure 8.103. As shown in Table 8.11, snow accumulated on day 23, and the model used this information to predict that the occupancy will be lower in the upcoming hours.

Figure 8.104 presents the forecasts from the model trained with the exogenous variable Dew Point. Similarly to what has been observed in Antwerp, this exogenous variable does not provide any useful information.

From the model forecasts using each of the numerical variables as input to produce parking occupancy forecasts, it can be concluded that:

- The usefulness of the exogenous variable values is dependent on the scenario, as observed with the Wind Speed variable.
- The subset of proxy variables (UV Index and especially Temperature and Humidity) could provide extra information, which was provided only by the sunshine hours in the Antwerp observations.

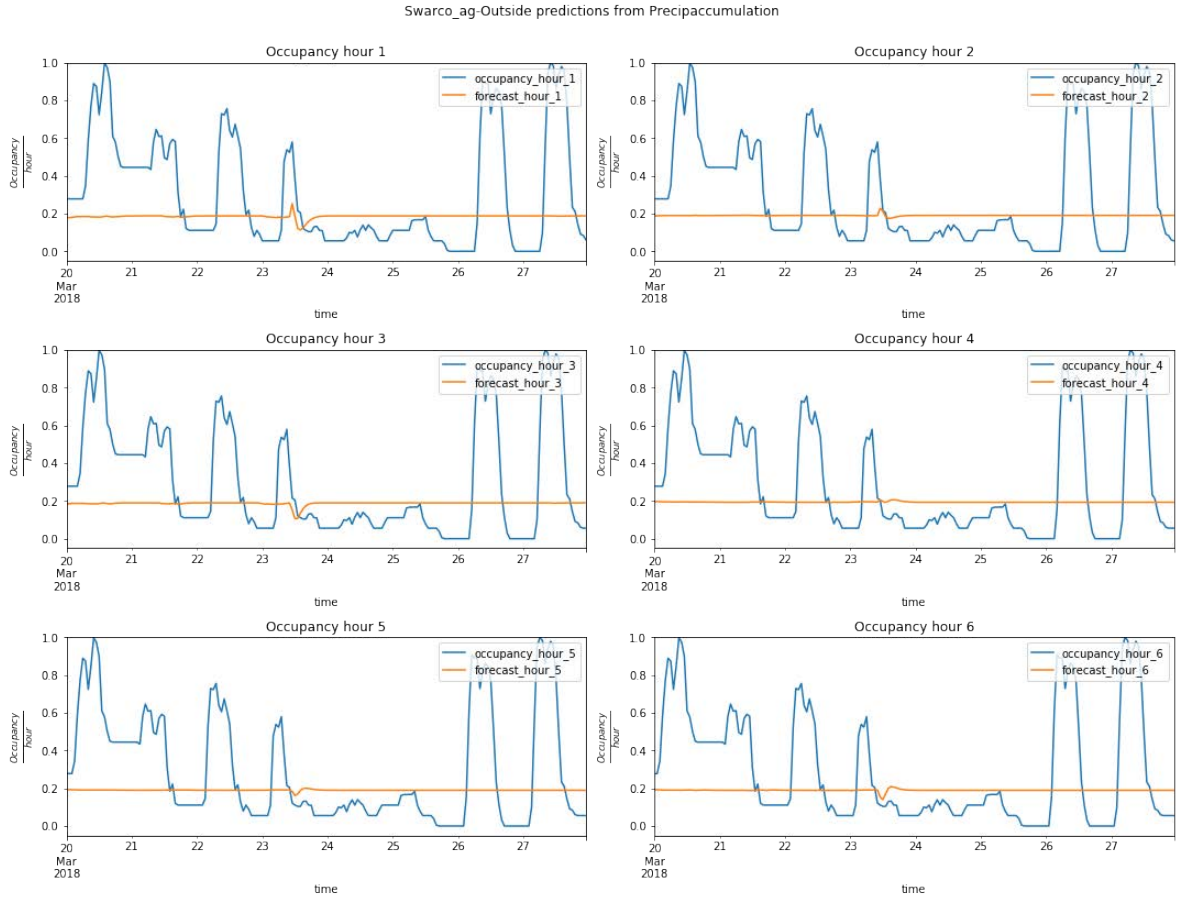


Figure 8.103: Outside sector (Wattens) parking occupancy (blue) and forecasts from a model trained with the exogenous variable Precipitation Accumulation (orange).

time	precAccumulation
2018-03-23 00:00:00	0.046424
2018-03-23 01:00:00	0.016262
2018-03-23 02:00:00	0.057355
2018-03-23 03:00:00	0.038664
2018-03-23 04:00:00	0.011066
2018-03-23 05:00:00	0.049663
2018-03-23 06:00:00	0.023887
2018-03-23 07:00:00	0.005398
2018-03-23 08:00:00	0.000000
2018-03-23 09:00:00	0.000000
2018-03-23 10:00:00	1.000000
2018-03-23 11:00:00	1.000000
2018-03-23 12:00:00	0.000000
2018-03-23 13:00:00	0.000000
2018-03-23 14:00:00	0.000000
2018-03-23 15:00:00	0.000000
2018-03-23 16:00:00	0.000000
2018-03-23 17:00:00	0.000000
2018-03-23 18:00:00	0.000000
2018-03-23 19:00:00	0.000000
2018-03-23 20:00:00	0.000000
2018-03-23 21:00:00	0.000000
2018-03-23 22:00:00	0.000000
2018-03-23 23:00:00	0.000000

Table 8.11: Wattens Precipitation Accumulated values for the date 2019-01-22.

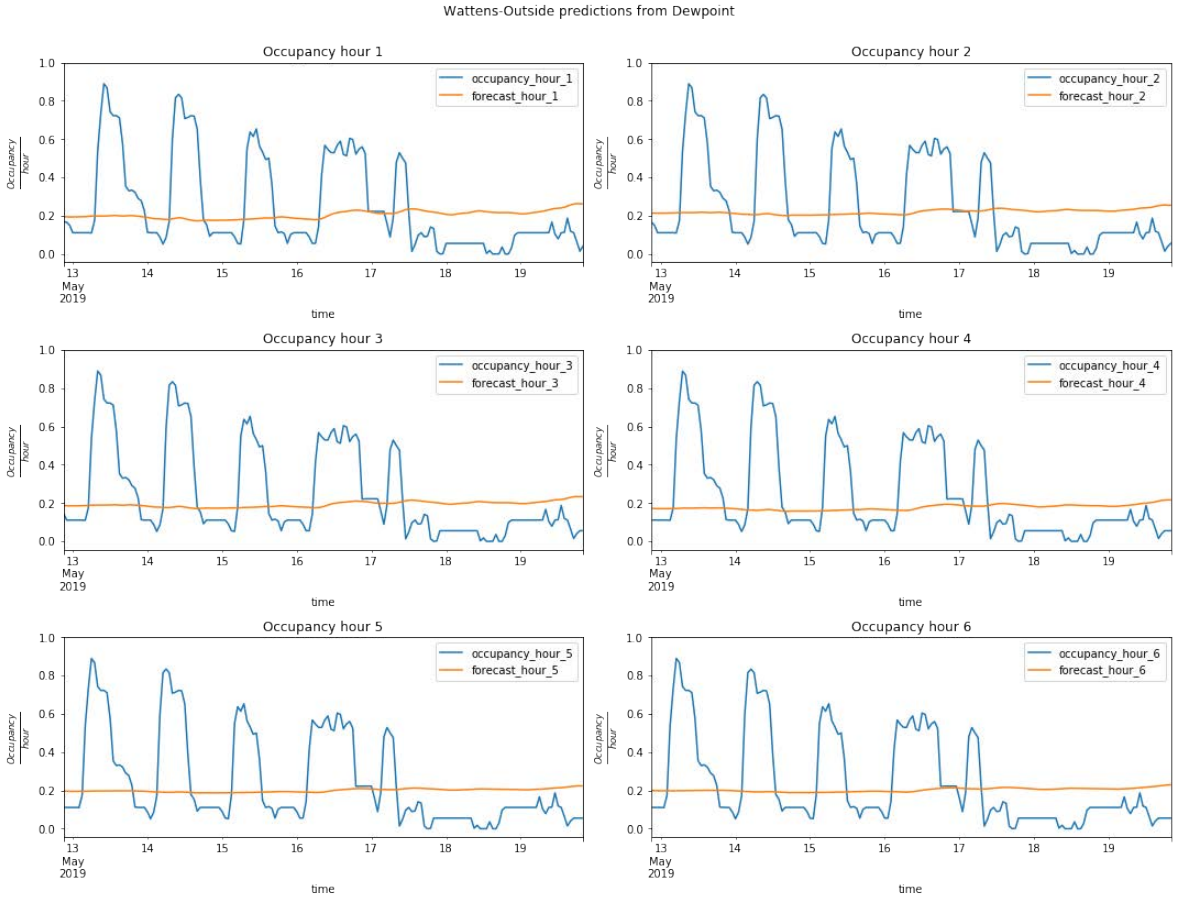


Figure 8.104: Wattens scenario parking occupation (blue) and forecasts (orange). The forecasts are provided by a model trained with the exogenous variable Dew Point.

time	visibility	cloudCover	windBearing	uvIndex
2019-10-17 00:00:00	1.000000	0.09	0.506964	0.000000
2019-10-17 01:00:00	1.000000	0.09	0.493036	0.000000
2019-10-17 02:00:00	1.000000	0.13	0.420613	0.000000
2019-10-17 03:00:00	1.000000	0.13	0.295265	0.000000
2019-10-17 04:00:00	1.000000	0.25	0.342618	0.000000
2019-10-17 05:00:00	1.000000	0.33	0.353760	0.000000
2019-10-17 06:00:00	1.000000	0.50	0.295265	0.000000
2019-10-17 07:00:00	1.000000	0.53	0.239554	0.000000
2019-10-17 08:00:00	0.979306	0.53	0.236769	0.000000
2019-10-17 09:00:00	1.000000	0.55	0.267409	0.090909
2019-10-17 10:00:00	1.000000	0.55	0.339833	0.272727
2019-10-17 11:00:00	1.000000	0.48	0.412256	0.363636
2019-10-17 12:00:00	1.000000	0.41	0.417827	0.454545
2019-10-17 13:00:00	1.000000	0.20	0.412256	0.454545
2019-10-17 14:00:00	1.000000	0.21	0.448468	0.454545
2019-10-17 15:00:00	1.000000	0.24	0.532033	0.272727
2019-10-17 16:00:00	1.000000	0.28	0.604457	0.181818
2019-10-17 17:00:00	1.000000	0.34	0.662953	0.000000
2019-10-17 18:00:00	1.000000	0.26	0.632312	0.000000
2019-10-17 19:00:00	1.000000	0.23	0.793872	0.000000
2019-10-17 20:00:00	1.000000	0.19	0.679666	0.000000
2019-10-17 21:00:00 0	1.000000	0.17	0.545961	0.000000
2019-10-17 22:00:00	1.000000	0.16	0.590529	0.000000
2019-10-17 23:00:00	1.000000	0.14	0.584958	0.000000

Table 8.12: Los Angeles Visibility, Cloud Cover, Wind Bearing and UV Index values for 17 October 2019.

- The variables that do not provide useful forecasts are Dew Point, Cloud Cover and Wind Bearing.

Los Angeles

As support for the analysis, Table 8.12 provides the numerical values for 17 October 2019.

Figure 8.105 presents the forecasts from the model trained with the Visibility variable. The visualizations show that the model does not learn any useful information for forecasting parking occupancy. Moreover, it could not be used to improve or support other exogenous variables.

The model trained with the numerical exogenous variable Cloud Cover is not capable of forecasting parking occupancy, as can be seen in Figure 8.106.

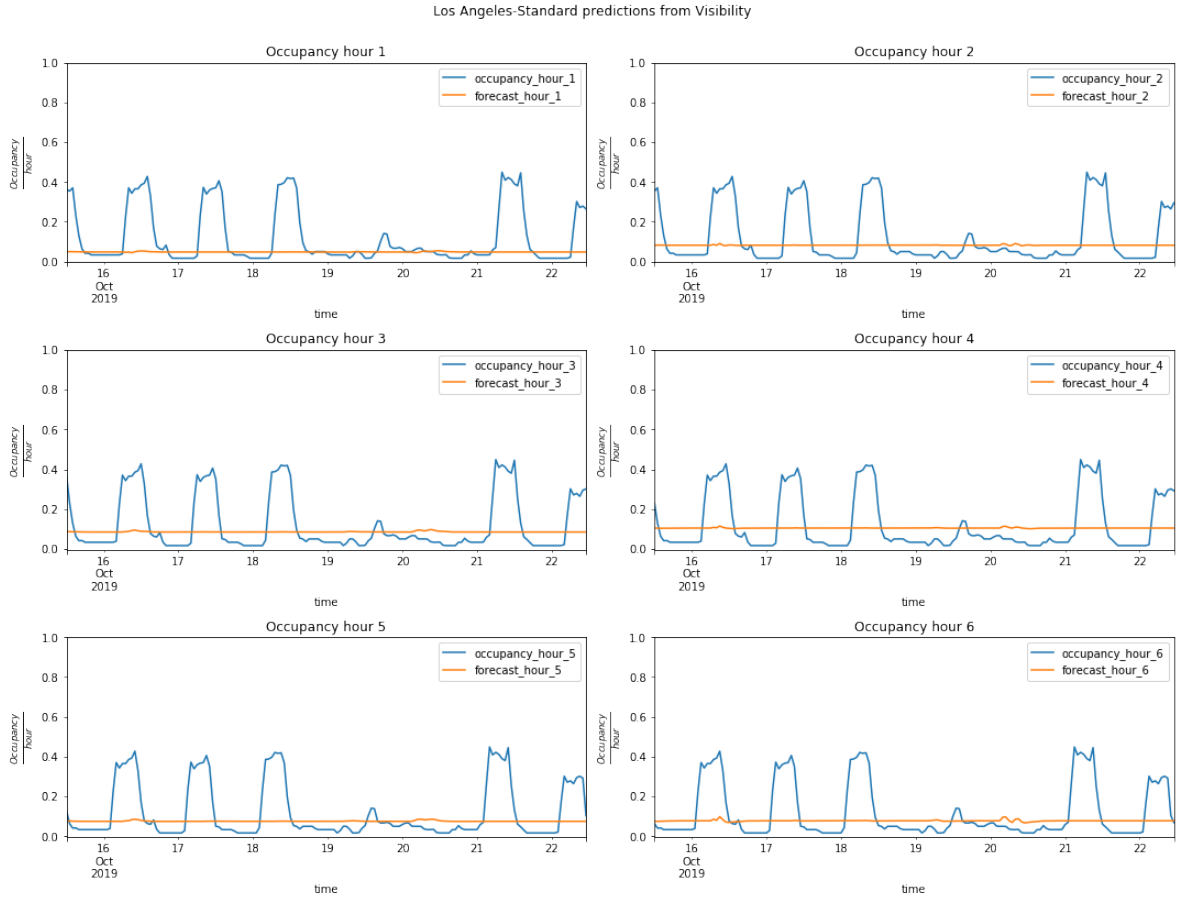


Figure 8.105: Time series from Los Angeles Standard sector parking occupancy (blue) and forecasts from a model trained with the exogenous variable Visibility (orange).

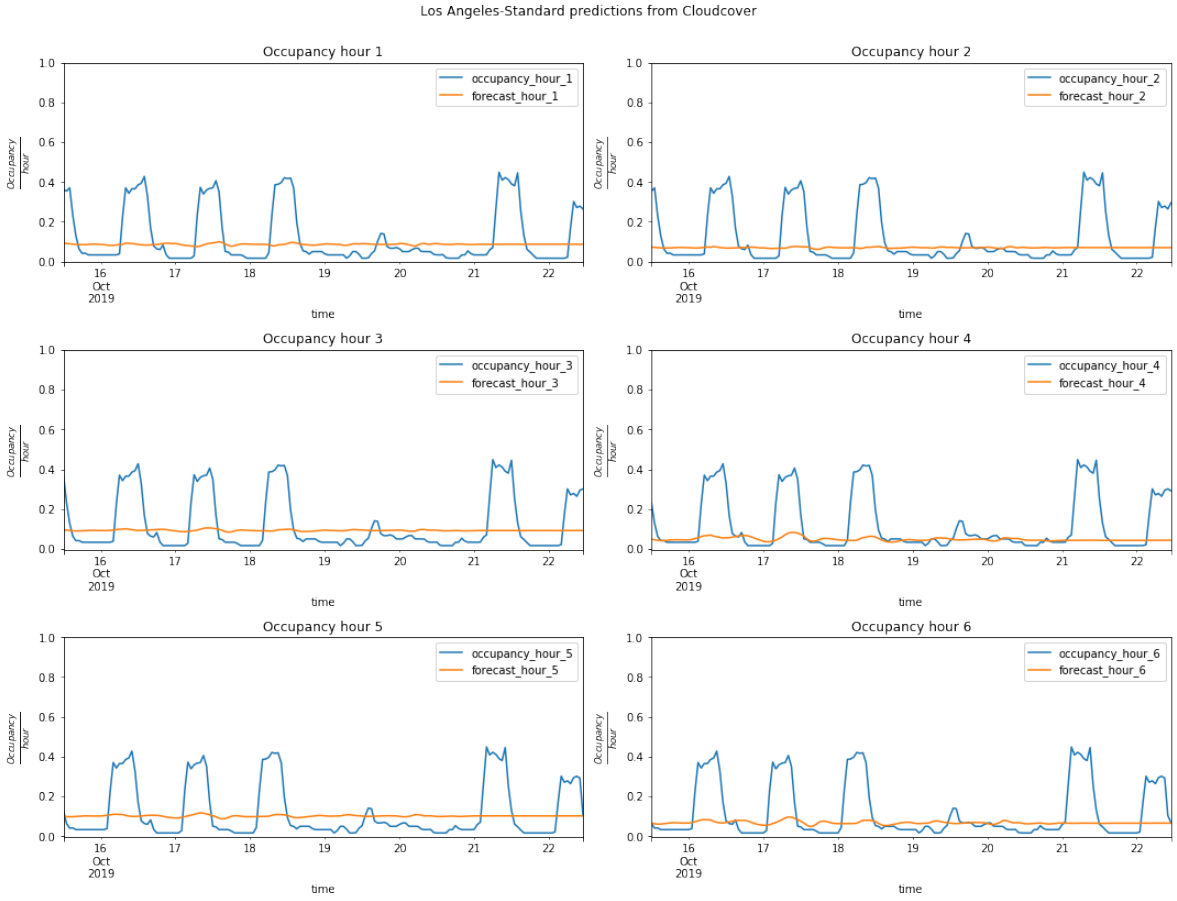


Figure 8.106: Visualizations of the parking occupancy in Los Angeles Standard sector (blue) and the forecasts of a model trained with the Cloud Cover variable (orange).

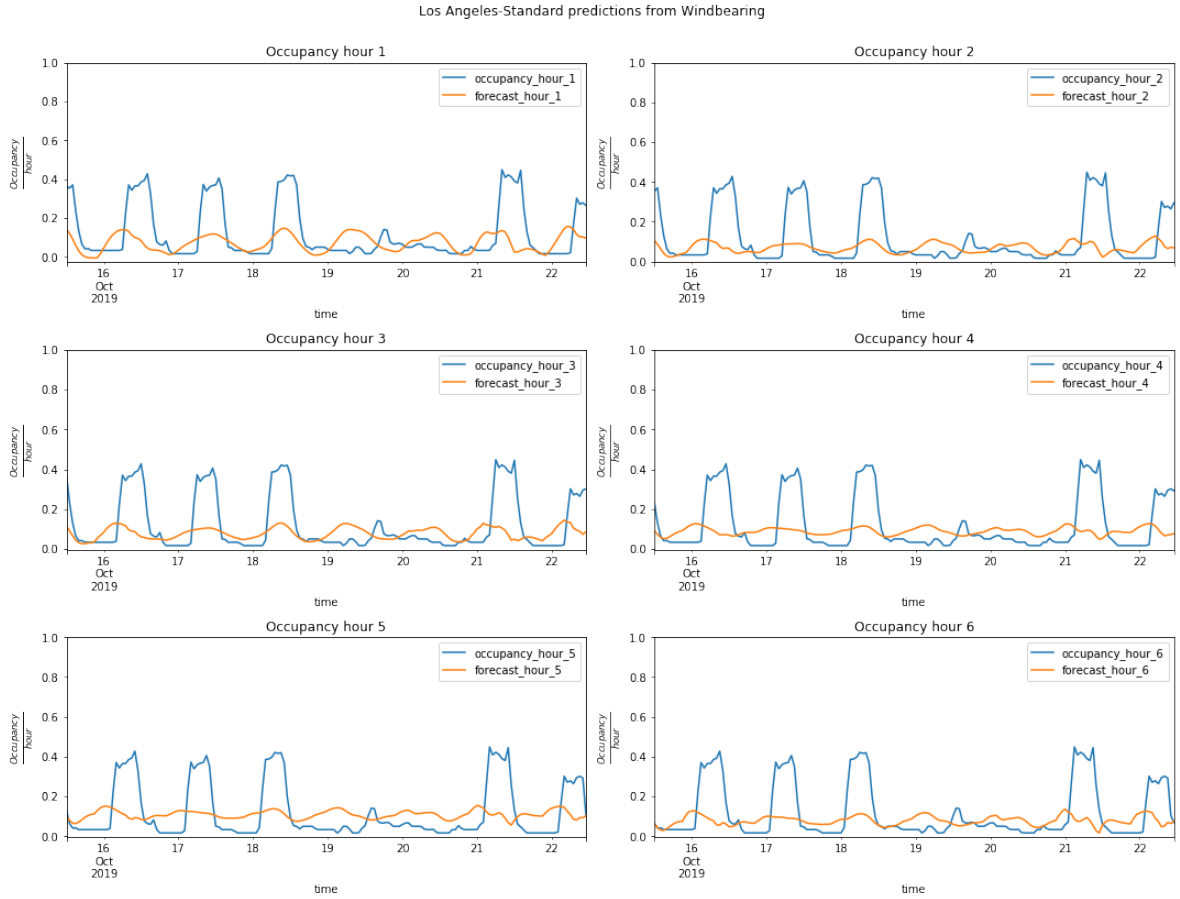


Figure 8.107: Time series of the real parking occupancy of the Standard sector and the values forecasted by a model trained with the exogenous variable Wind Bearing.

Figure 8.107 presents the forecasts from the model trained with the Wind Bearing variable. The model is somehow able to provide forecasts with different values of parking occupancy, but no relationship could be obtained from the visualization. There seems to be some spurious correlation between the values of the Wind Bearing variable and those of the parking occupancy.

Figure 8.108 presents the forecasts from the model trained with the UV Index variable. Similarly to other scenarios, the model learns the relationship between sunshine hours and parking occupancy. However, it fails to distinguish weekdays from weekends or the correct levels of occupancy.

Information about the next numerical variables (Humidity, Precipitation Intensity, Temperature and Wind Gust) is offered in Table 8.13 as support for the text.

The visualizations in Figure 8.109 present the six time step forecasts from the model trained with the Humidity variable. In the same way as in Antwerp, the model learns the sunshine hours and relates them to the parking occupancy based on the cyclical behaviour of the relative humidity. It can be observed that the model for 18 October

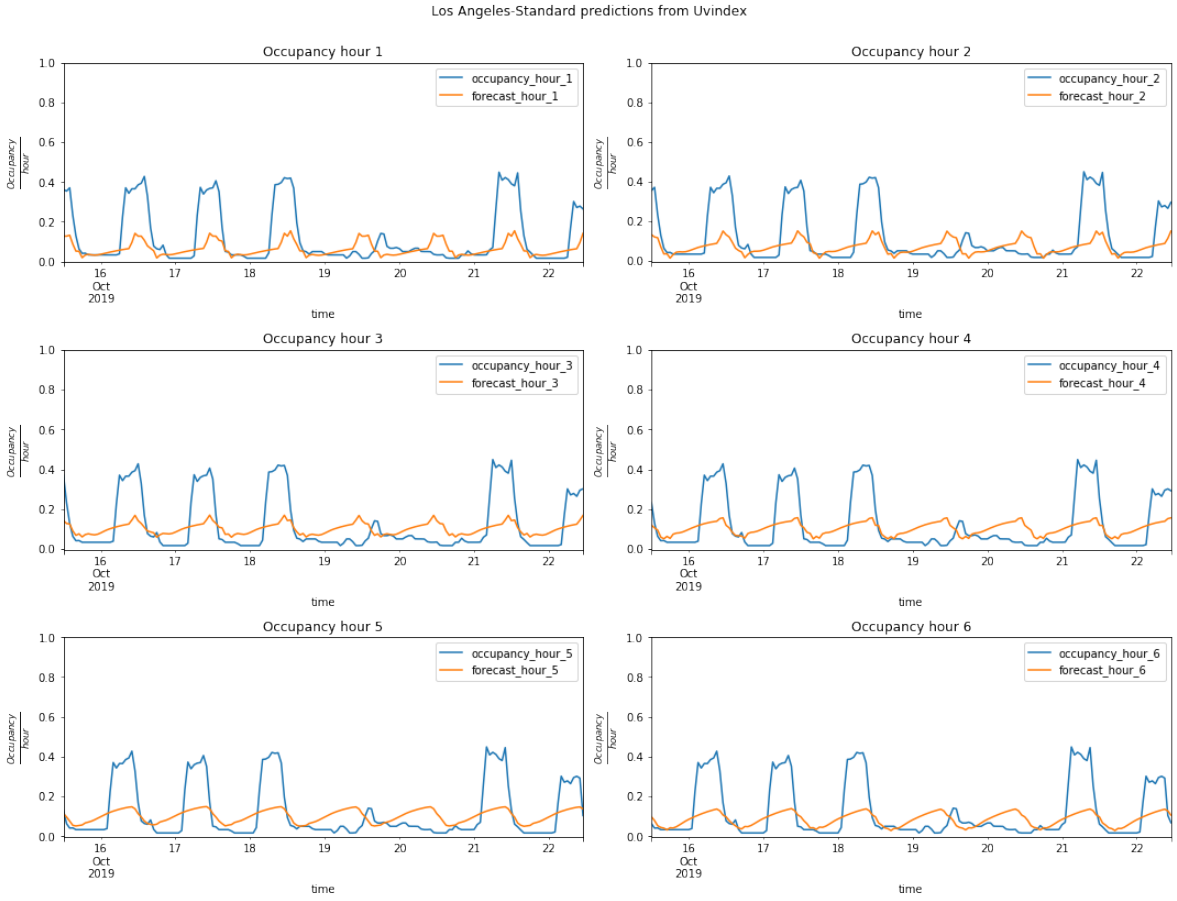


Figure 8.108: Los Angeles six-hour forecasts from a model trained using the continuous variable UV Index.

time	humidity	precipIntensity	temperature	windGust
2019-10-17 00:00:00	0.773196	0.000000	0.393186	0.137492
2019-10-17 01:00:00	0.814433	0.000000	0.374797	0.136150
2019-10-17 02:00:00	0.804124	0.000000	0.353164	0.122736
2019-10-17 03:00:00	0.845361	0.000000	0.350730	0.097250
2019-10-17 04:00:00	0.855670	0.000000	0.346944	0.11670
2019-10-17 05:00:00	0.865979	0.000000	0.355868	0.133467
2019-10-17 06:00:00	0.876289	0.000000	0.357761	0.105298
2019-10-17 07:00:00	0.896907	0.000000	0.357491	0.158954
2019-10-17 08:00:00	0.855670	0.000000	0.376420	0.218645
2019-10-17 09:00:00	0.793814	0.000000	0.405895	0.213280
2019-10-17 10:00:00	0.752577	0.000000	0.435911	0.241449
2019-10-17 11:00:00	0.721649	0.000000	0.467550	0.171697
2019-10-17 12:00:00	0.690722	0.000000	0.482693	0.193830
2019-10-17 13:00:00	0.6597946	0.002220	0.514332	0.248820
2019-10-17 14:00:00	0.628866	0.002337	0.532180	0.209926
2019-10-17 15:00:00	0.639175	0.001319	0.525419	0.283700
2019-10-17 16:00:00	0.659794	0.000000	0.505679	0.264252
2019-10-17 17:00:00	0.731959	0.000000	0.470525	0.336016
2019-10-17 18:00:00	0.742268	0.000914	0.450784	0.200537
2019-10-17 19:00:00	0.762887	0.000000	0.422390	0.179074
2019-10-17 20:00:00	0.773196	0.000000	0.406436	0.138833
2019-10-17 21:00:00	0.752577	0.002102	0.396971	0.135480
2019-10-17 22:00:00	0.742268	0.001580	0.383721	0.156940
2019-10-17 23:00:00	0.731959	0.001737	0.378042	0.154930

Table 8.13: Los Angeles Humidity, Precipitation Intensity, Temperature and Wind Gust values for 17 October 2019

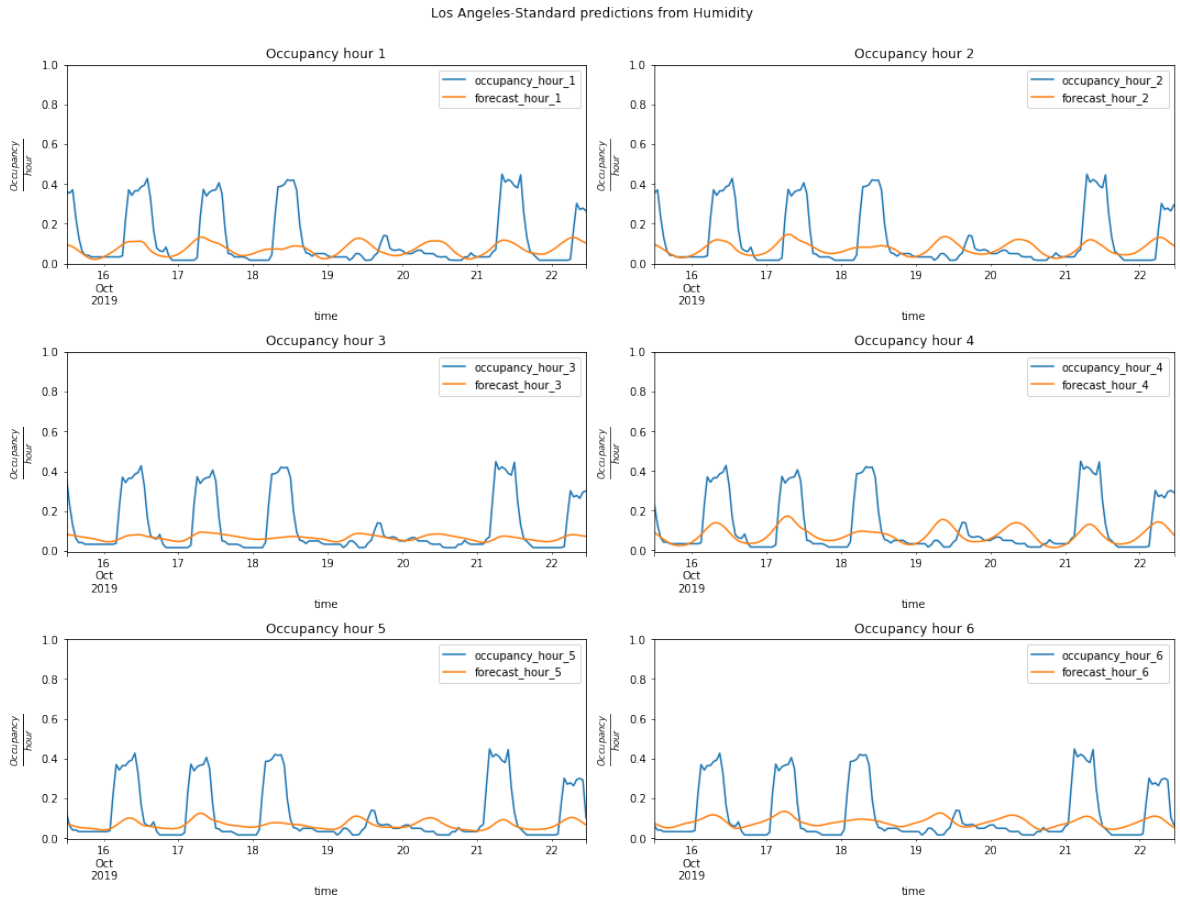


Figure 8.109: Time series visualizations for the second week of April 2019 in the Los Angeles Standard sector. The orange line shows the six-hour forecasts obtained from a model trained using the continuous variable Humidity.

forecasts lower levels of parking occupancy, but the true occupancy did not change. Therefore, it is possible to interpret that Humidity is not providing any useful information.

Figure 8.110 presents the forecasts from the model trained with the exogenous numerical variable Precipitation Intensity. The effects of the information provided by the variable are barely noticeable in the forecasts of the model (for example, there is a small change in the forecasted values on day 19). Despite this, this variable could serve as an auxiliary for improving occasional forecasts.

The visualizations from the forecasts of the model trained with the exogenous variable Temperature to provide multiple forecasts for the Los Angeles scenario are presented in Figure 8.111. Although the model relates sunshine hours with parking occupancy, it is similar to other observations in that the values provided are not good, as they are insufficient for establishing patterns according to the day of the week (for example, weekends).

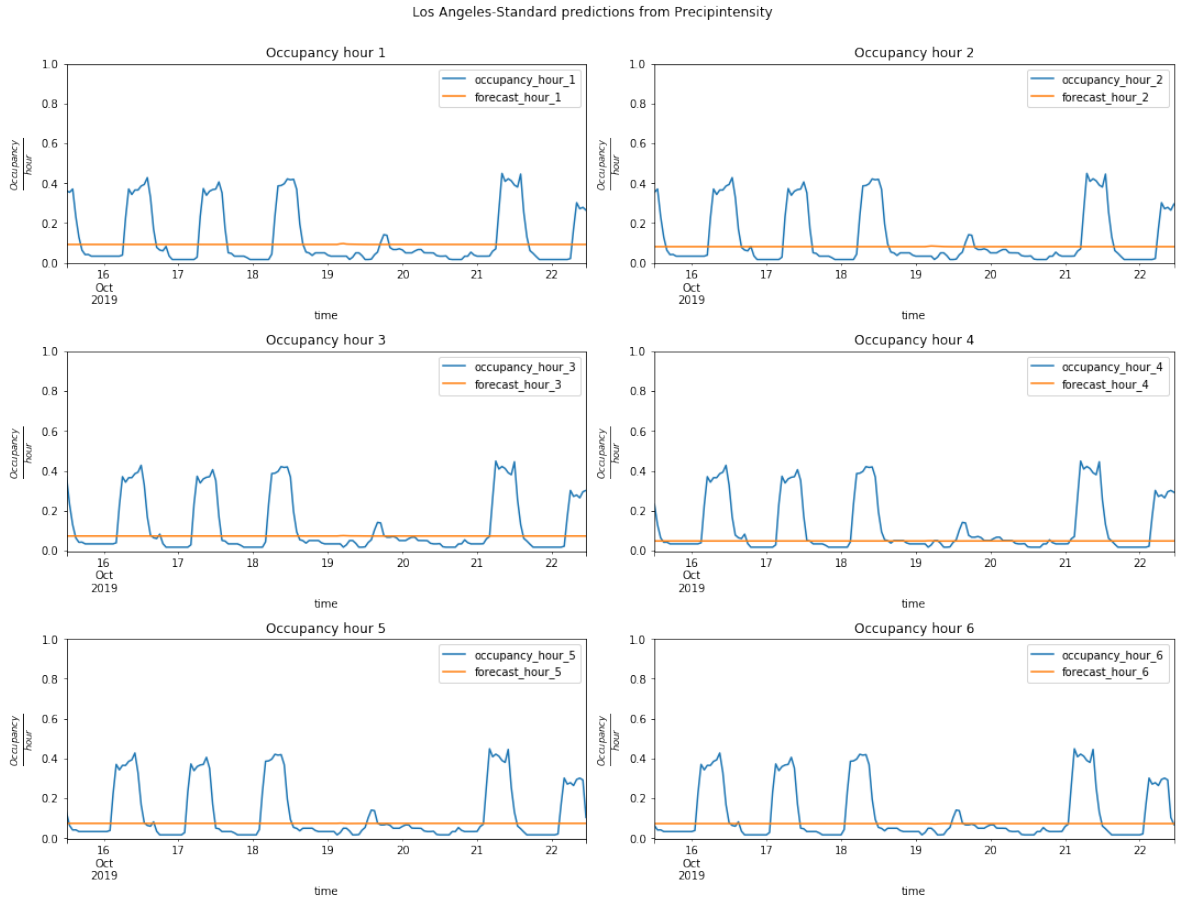


Figure 8.110: Visualizations at different time steps (from one to six) from a model trained with the exogenous variable Precipitation Intensity in order to provide occupancy forecasts for the Los Angeles scenario.

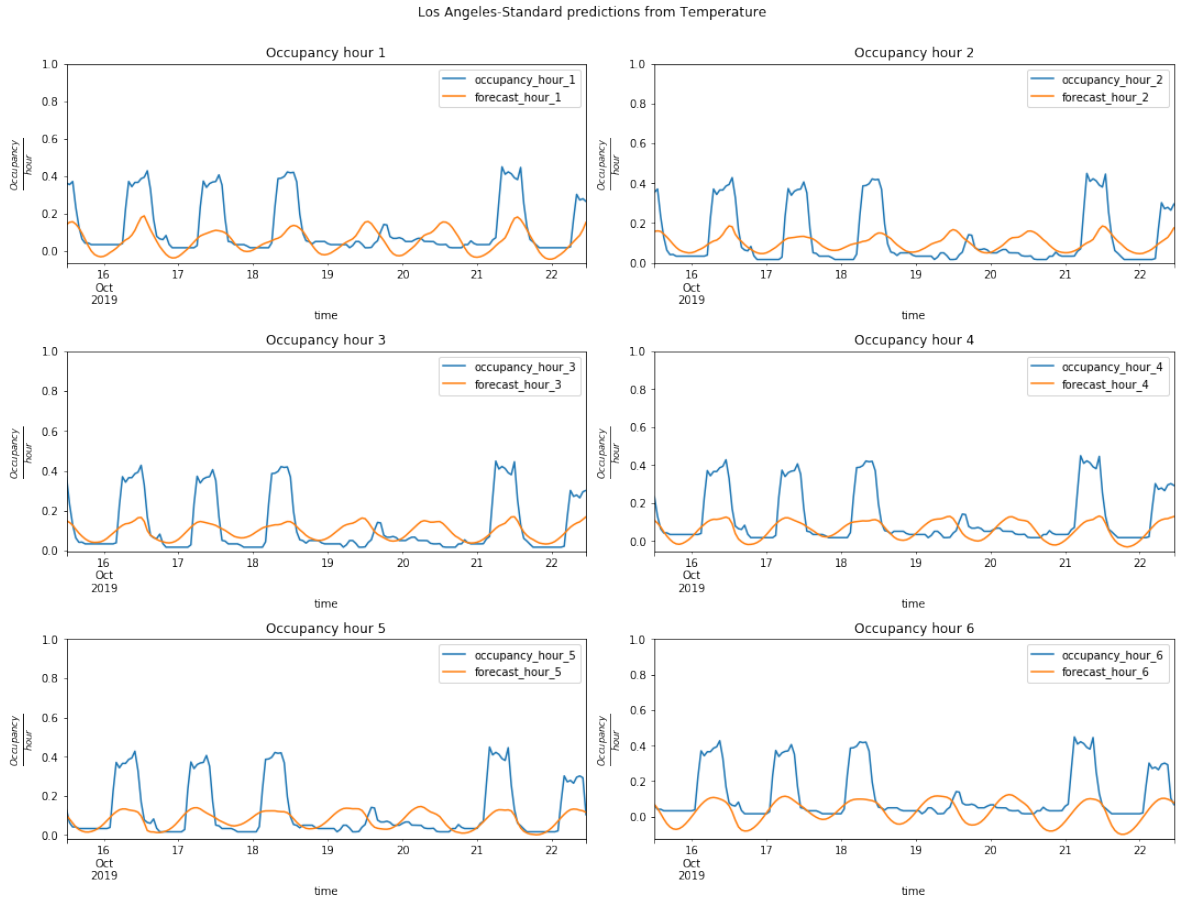


Figure 8.111: Forecasts for the next six hours from a model trained using the continuous variable Temperature in the Los Angeles scenario. The data interval is 15 to 23 October 2019.

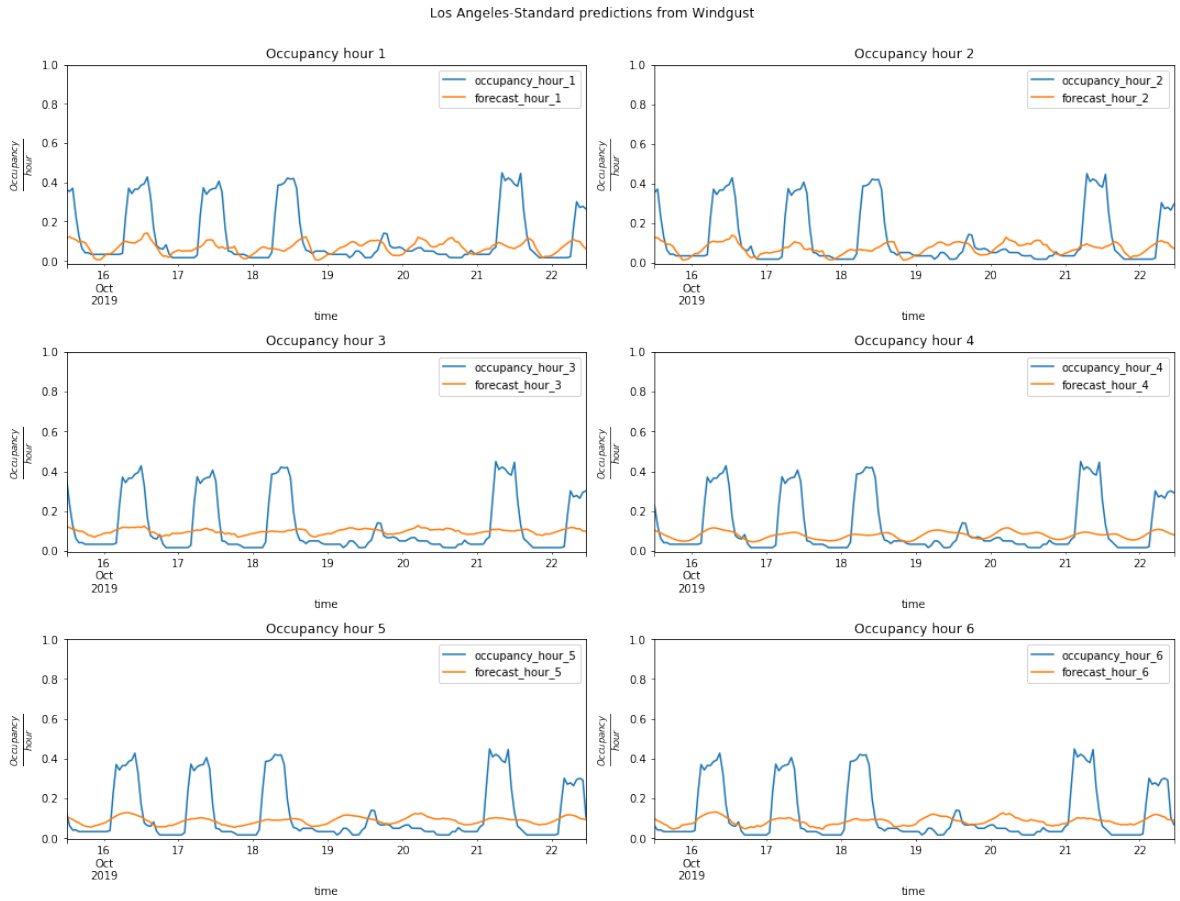


Figure 8.112: Los Angeles Standard sector six-hour forecasts from a model trained using the continuous variable Wind Gust.

The forecasts of the model trained with the exogenous variable Wind Gust are presented in Figure 8.112. From the visualizations of the forecasts, it is not clear if there is an effect between the Wind Gust values and the parking occupancies, although the model is able to follow cyclical daily patterns, possibly because the Wind Gust values are also cyclical throughout the days.

Table 8.14 provides auxiliary information about the values of the variables Wind Speed, Precipitation Accumulation and Dew Point.

Figure 8.113 presents the six time step forecasts from the model trained with the Wind Speed variable. As the values of the variable seem to have a cyclical pattern throughout the days, the model learns to map those to the parking occupancy values.However, the variable alone is clearly not enough for providing useful forecasts or interpretable information.

The variable Precipitation Accumulation does not provide any satisfactory information, as it is always zero for the Los Angeles scenario.

In the same way as previous scenarios, the Dew Point variable does not allow any useful

time	windSpeed	precipAccumulation	dewPoint
2019-10-17 00:00:00	0.193878	0.0	0.837220
2019-10-17 01:00:00	0.240363	0.0	0.839865
2019-10-17 02:00:00	0.181406	0.0	0.817264
2019-10-17 03:00:00	0.121315	0.0	0.830248
2019-10-17 04:00:00	0.119048	0.0	0.830729
2019-10-17 05:00:00	0.138322	0.0	0.842991
2019-10-17 06:00:00	0.136054	0.0	0.851166
2019-10-17 07:00:00	0.265306	0.0	0.857418
2019-10-17 08:00:00	0.361678	0.0	0.859101
2019-10-17 09:00:00	0.329932	0.0	0.858379
2019-10-17 10:00:00	0.383220	0.0	0.865112
2019-10-17 11:00:00	0.259637	0.0	0.874249
2019-10-17 12:00:00	0.311791	0.0	0.870642
2019-10-17 13:00:00	0.340136	0.0	0.882664
2019-10-17 14:00:00	0.299320	0.0	0.879057
2019-10-17 15:00:00	0.402494	0.0	0.881943
2019-10-17 16:00:00	0.354875	0.0	0.876172
2019-10-17 17:00:00	0.478458	0.0	0.881943
2019-10-17 18:00:00	0.286848	0.0	0.871844
2019-10-17 19:00:00	0.243764	0.0	0.858379
2019-10-17 20:00:00	0.181406	0.0	0.845396
2019-10-17 21:00:00 0	0.193878	0.0	0.827843
2019-10-17 22:00:00	0.219955	0.0	0.814619
2019-10-17 23:00:00	0.219955	0.0	0.802837

Table 8.14: Los Angeles Wind Speed, Precipitation Accumulation and Dew Point values for 17 October 2019

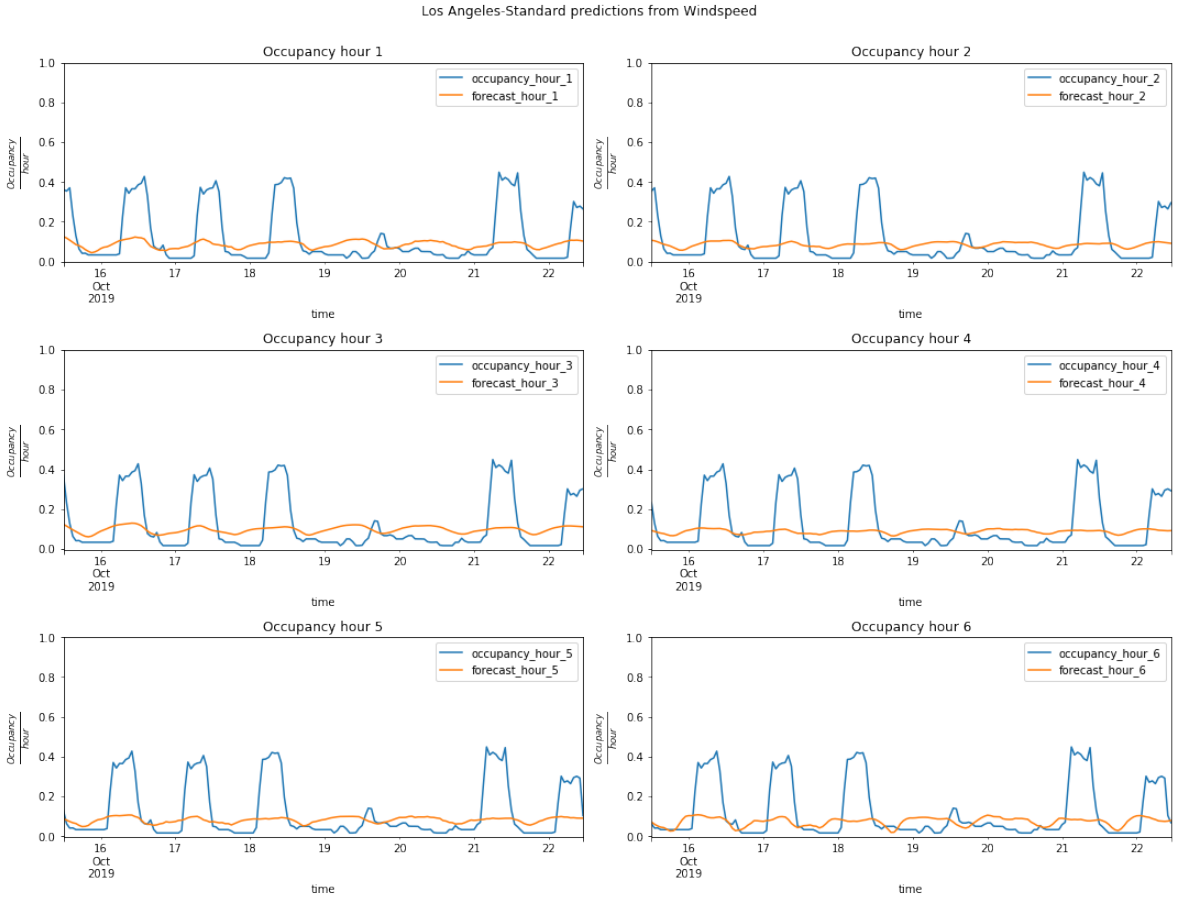


Figure 8.113: Parking occupancy forecasts (next six hours) using a model trained with Wind Speed information from the Los Angeles scenario.

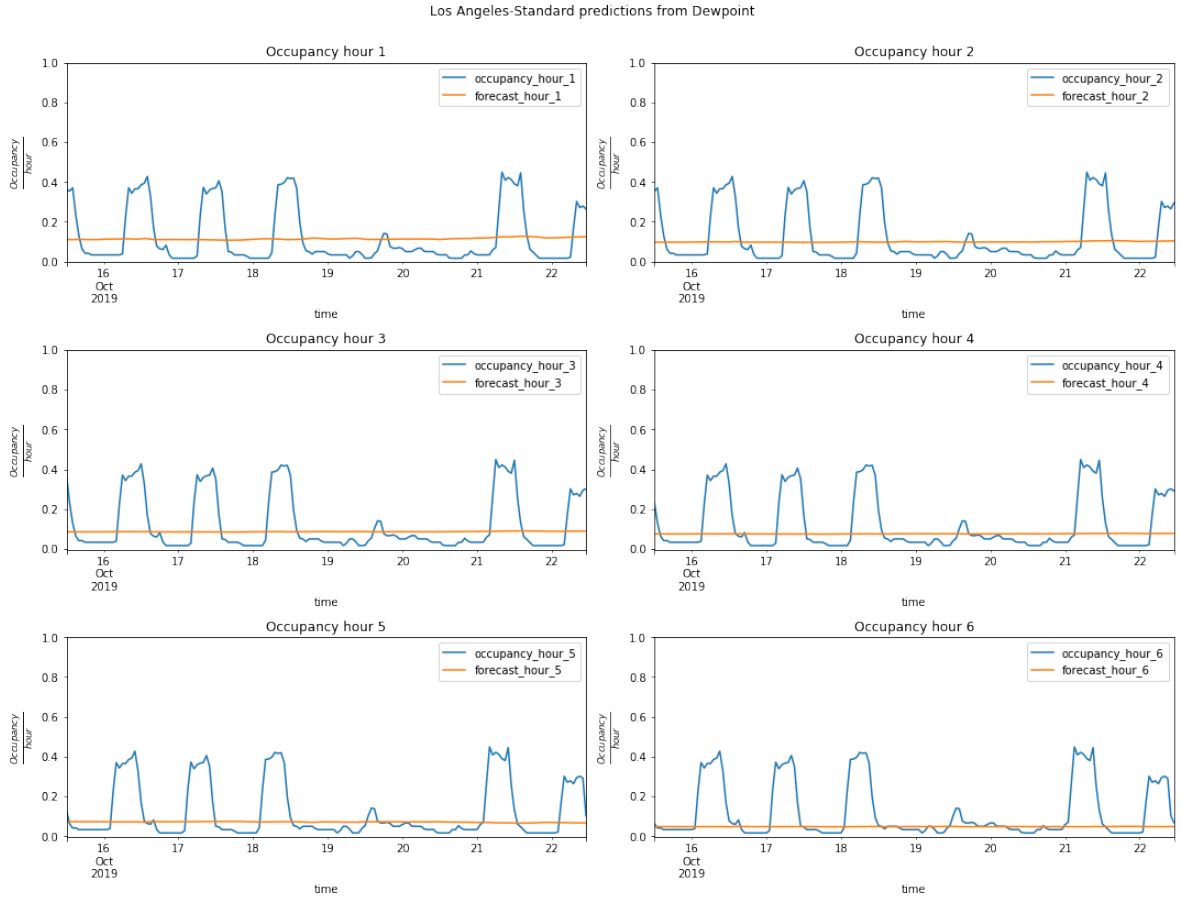


Figure 8.114: Time series visualizations from a model trained with the exogenous variable Dew Point to produce parking occupancy forecasts for the Los Angeles Standard sector. The blue line denotes the real occupancy values, and orange the forecasted ones.

interpretation about any relationship that may exist between Dew Point and parking occupancy, as can be seen in the visualizations in Figure 8.114.

The analysis of the exogenous numerical variables conducted for the city of Los Angeles (Standard sector) does not add any extra information to what has been observed previously in the Wattens and Antwerp scenarios.

Riyadh

The last scenario corresponds to Riyadh. Its singular weather conditions could have a different effect on parking occupancy when compared to the previous scenarios. As support for the analysis, Table 8.15 provides the numerical values for October 18 2019.

Figure 8.115 presents the forecasts from the model trained with the Visibility variable. Little information is obtained from the variable, as it nearly always has a value of 1, and

time	visibility	cloudCover	windBearing	uvIndex
2019-10-18 00:00:00	1.0	1.00	0.852368	0.000000
2019-10-18 01:00:00	1.0	1.00	0.883008	0.000000
2019-10-18 02:00:00	1.0	1.00	0.994429	0.000000
2019-10-18 03:00:00	1.0	1.00	0.860724	0.000000
2019-10-18 04:00:00	1.0	0.98	0.813370	0.000000
2019-10-18 05:00:00	1.0	0.96	0.857939	0.000000
2019-10-18 06:00:00	1.0	0.94	0.871866	0.000000
2019-10-18 07:00:00	1.0	0.93	0.883008	0.000000
2019-10-18 08:00:00	1.0	0.92	0.913649	0.153846
2019-10-18 09:00:00	1.0	0.83	0.935933	0.230769
2019-10-18 10:00:00	1.0	0.57	0.977716	0.384615
2019-10-18 11:00:00	1.0	0.24	0.025070	0.615385
2019-10-18 12:00:00	1.0 8	0.00	0.061281	0.69230
2019-10-18 13:00:00	1.0	0.005	0.077994	0.61538
2019-10-18 14:00:00	1.0	0.00	0.080780	0.384615
2019-10-18 15:00:00	1.0	0.03 0	.086351	0.230769
2019-10-18 16:00:00	1.0	0.11	0.097493	0.076923
2019-10-18 17:00:00	1.0	0.22	0.105850	0.000000
2019-10-18 18:00:00	1.0	0.32	0.105850	0.000000
2019-10-18 19:00:00	1.0	0.44	0.114206	0.000000
2019-10-18 20:00:00	1.0	0.51	0.125348	0.000000
2019-10-18 21:00:00	1.0	0.58	0.128134	0.000000
2019-10-18 22:00:00	1.0	0.48	0.114206	0.000000
2019-10-18 23:00:00	1.0	0.38 0	0.089136	0.000000

Table 8.15: Riyadh Visibility, Cloud Cover, Wind Bearing and UV Index values for 15 May 2019.

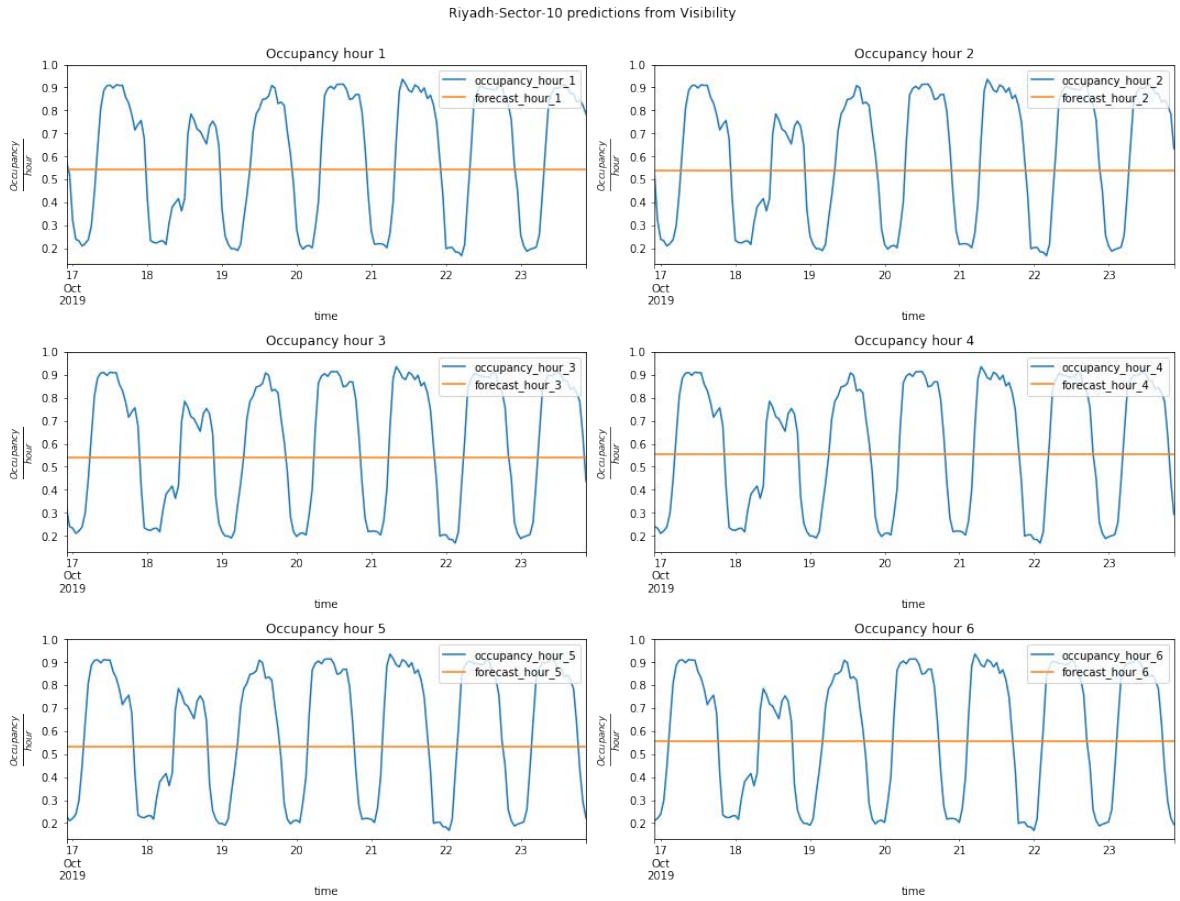


Figure 8.115: Sector-10 in Riyadh six-hour forecasts of the parking occupancy from a model trained with only the Visibility exogenous variable.

this is reflected in the forecasts with a constant forecasted value.

The time series visualizations presented in Figure 8.116 show the forecasts at six time steps of a model trained with the exogenous variable Cloud Cover to predict parking occupancies for Sector-10 in Riyadh. From the visualizations, there is no clear effect or usefulness of the variable values for the model to exploit.

Figure 8.117 presents visualizations of the forecasts from the model trained with the exogenous variable Wind Bearing. In contrast to other scenarios, it seems that the direction of the wind in Riyadh has some cyclical behaviour that the model learns in order to provide parking occupancy forecasts. Despite this, it is more likely that the correlation between Wind Bearing and parking occupancy is spurious.

Similar observations to those of the Wind Bearing variable can be made from the visualizations of the time series in Figure 8.118. The forecasts of the model with all the numerical variables is very similar to the one presented here. Therefore, it is possible that UV Index is the variable with more information about the parking occupancy patterns throughout the day. However, it is not true that UV Index is related to the

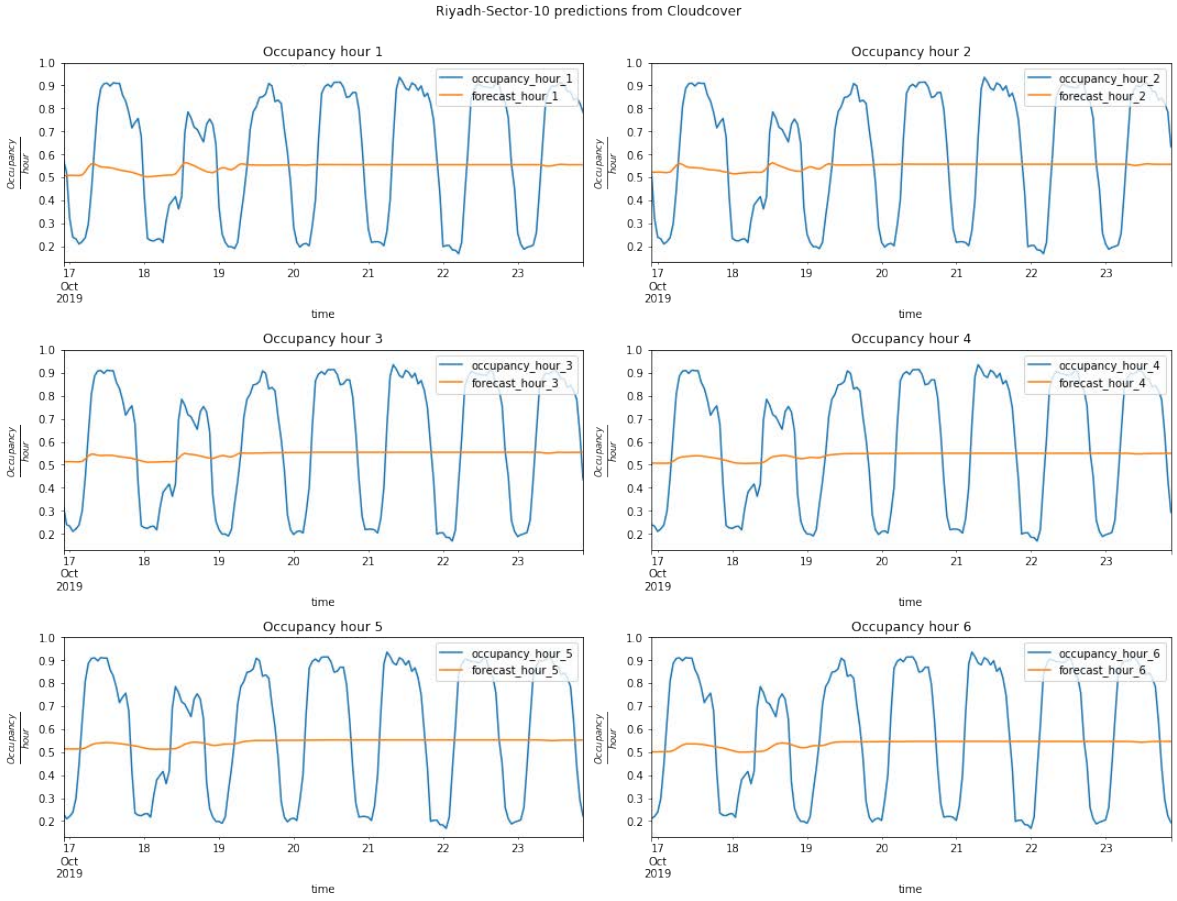


Figure 8.116: Riyadh forecasts from a model trained with the Cloud Cover exogenous variable. Blue denotes the real parking occupancy and orange the forecasted ones.

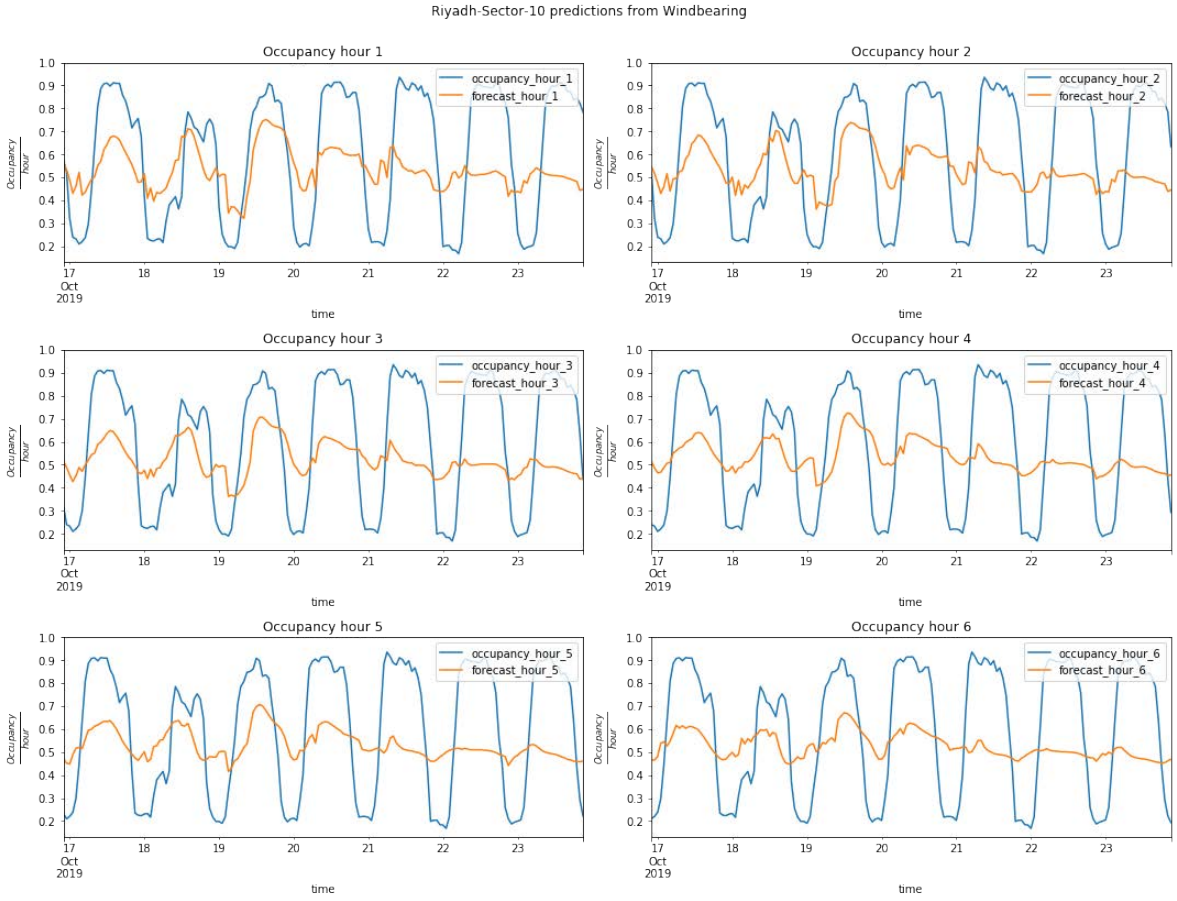


Figure 8.117: Time series visualizations from a model trained using only the exogenous variable Wind Bearing to forecast parking occupancies in Sector-10 in Riyadh.

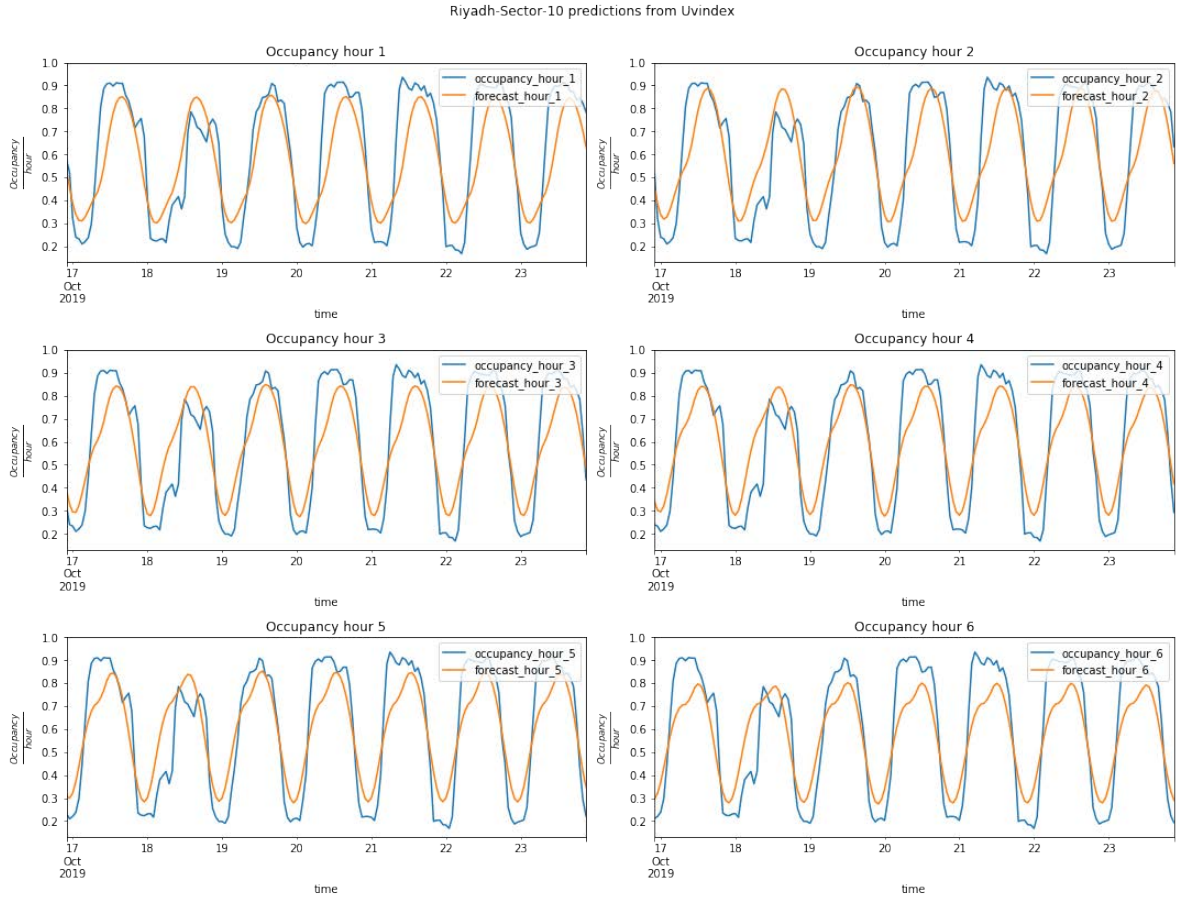


Figure 8.118: Riyadh Sector-10 time series visualizations from a model trained with the categorical variable UV Index. Although it seems that the model is able to discern parking occupancy from the UV Index, the reality is that UV Index measurements are related to the sunshine hours in a similar way as human activities, thus affecting parking occupancy levels similarly.

parking occupancy because parking occupancy is actually a consequence of human activities. Therefore, as those start to happen early in the day (for example, people going to work), the parking occupancy in Sector-10 increases while it decreases at the end of the day as human activities also decrease. Therefore, there are no correlations between UV Index and parking occupancy, and the model merely exploits that both variables have similar patterns throughout the day, thus learning a spurious correlation. Variables like Day of the Week provide more useful information that can help the model learn parking occupancy, as human activities are conditioned by the day of the week. Therefore, the correlation is more plausible.

Information about the next numerical variables (Humidity, Precipitation Intensity, Temperature and Wind Gust) is offered in Table 8.16 as support for the text.

The visualizations in Figure 8.119 present the six time step forecasts from the model

time	humidity	precipIntensity	temperature	windGust
2019-10-18 00:00:00	0.195876	0.0	0.437995	0.022199
2019-10-18 01:00:00	0.206186	0.0	0.424009	0.025899
2019-10-18 02:00:00	0.216495	0.0	0.408625	0.030655
2019-10-18 03:00:00	0.216495	0.0	0.410490	0.038584
2019-10-18 04:00:00	0.226804	0.0	0.390676	0.080867
2019-10-18 05:00:00	0.247423	0.0	0.370163	0.104651
2019-10-18 06:00:00	0.237113	0.0	0.382051	0.129493
2019-10-18 07:00:00	0.206186	0.0	0.441958	0.128436
2019-10-18 08:00:00	0.164948	0.0	0.521445	0.115222
2019-10-18 09:00:00	0.134021	0.0	0.595338	0.113636
2019-10-18 10:00:00	0.113402	0.0	0.651049	0.135835
2019-10-18 11:00:00	0.092784	0.0	0.694406	0.169133
2019-10-18 12:00:00	0.082474	0.0	0.728671	0.198732
2019-10-18 13:00:00	0.072165	0.0	0.749650	0.218288
2019-10-18 14:00:00	0.061856	0.0	0.762937	0.233615
2019-10-18 15:00:00	0.082474	0.0	0.739161	0.250000
2019-10-18 16:00:00	0.082474	0.0	0.718415	0.271670
2019-10-18 17:00:00	0.092784	0.0	0.681119	0.294926
2019-10-18 18:00:00	0.103093	0.0	0.634033	0.309197
2019-10-18 19:00:00	0.123711	0.0	0.600000	0.313425
2019-10-18 20:00:00	0.144330	0.0	0.562005	0.313425
2019-10-18 21:00:00	0.164948	0.0	0.527273	0.273784
2019-10-18 22:00:00	0.195876	0.0	0.498601	0.255285
2019-10-18 23:00:00	0.206186	0.0	0.468065	0.115751

Table 8.16: Riyadh Humidity, Precipitation Intensity, Temperature and Wind Gust values for 18 October 2019

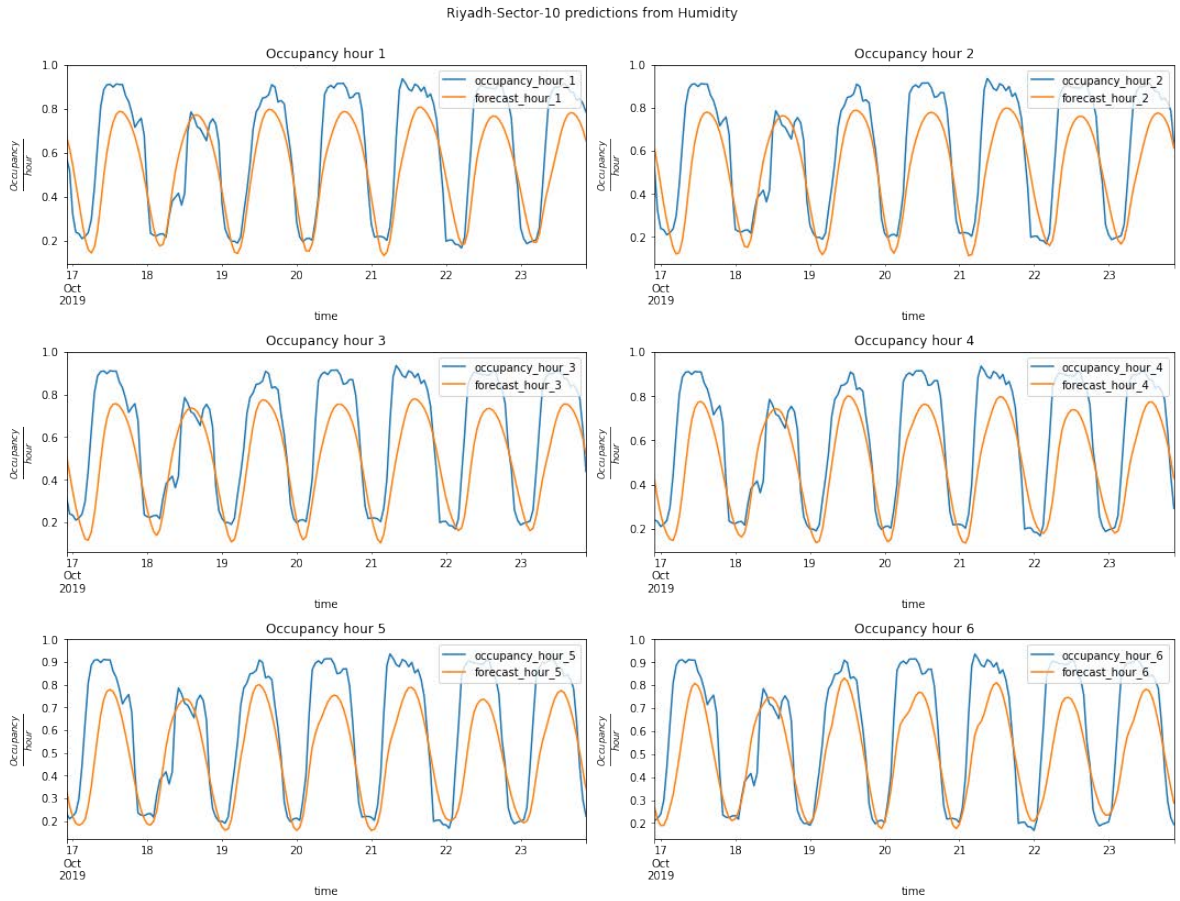


Figure 8.119: Riyadh Sector-10 scenario forecasts from the model trained with the exogenous variable Humidity. Real parking occupancy in blue and forecasted in orange.

trained with the Humidity variable. Again, it is possible that the model has learnt a spurious correlation between Humidity and parking occupancy. The provided forecasts (in orange) are a bit better than those obtained with the UV Index model, although they are the same for all days.

Figure 8.120 presents the forecasts from the model trained with the exogenous numerical variable Precipitation Intensity. The visualization shows that, although the time series usually has the same value (zero) when precipitation occurs, the intensity can affect parking occupancy. Thus, the information from the variable Precipitation Intensity becomes useful. Table 8.17 provides the values of the variable for 11 April 2019.

Figure 8.121 shows the time series of the Riyadh Sector-10 parking occupancy from a model trained with only the numerical variable Temperature. The obtained results are similar to those of IV Index and Humidity. The model is able to provide accurate forecasts because both variables have similar values throughout the day. However, inspecting each day reveals that the shapes of the forecasted values are always the same

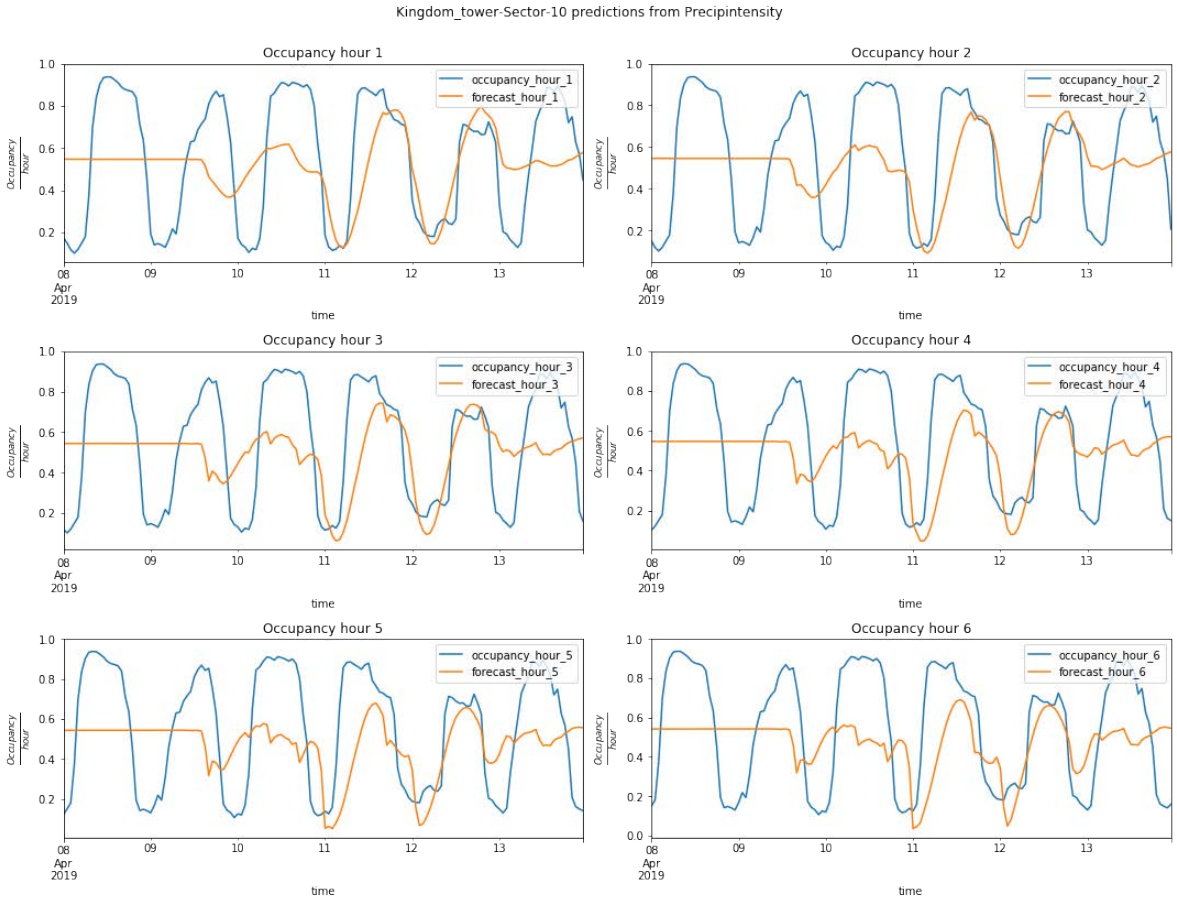


Figure 8.120: Riyadh scenario parking occupancy forecasts from the model trained with the exogenous variable Precipitation Intensity. Real occupancy in blue and forecasted in orange.

time	precipIntensity
2019-04-11 00:00:00	0.366658
2019-04-11 01:00:00	0.234663
2019-04-11 02:00:00	0.121081
2019-04-11 03:00:00	0.078550
2019-04-11 04:00:00	0.036199
2019-04-11 05:00:00	0.007410
2019-04-11 06:00:00	0.000000
2019-04-11 07:00:00	0.000000
2019-04-11 08:00:00	0.000000
2019-04-11 09:00:00	0.000000
2019-04-11 10:00:00	0.000000
2019-04-11 11:00:00	0.000000
2019-04-11 12:00:00	0.006871
2019-04-11 13:00:00	0.005255
2019-04-11 14:00:00	0.066020
2019-04-11 15:00:00	0.079448
2019-04-11 16:00:00	0.364951
2019-04-11 17:00:00	0.000000
2019-04-11 18:00:00	0.000000
2019-04-11 19:00:00	0.000000
2019-04-11 20:00:00	0.011003
2019-04-11 21:00:00	0.041543
2019-04-11 22:00:00	0.154002
2019-04-11 23:00:00	0.510689

Table 8.17: 11 April 2019 Precipitation Intensity values from Riyadh.

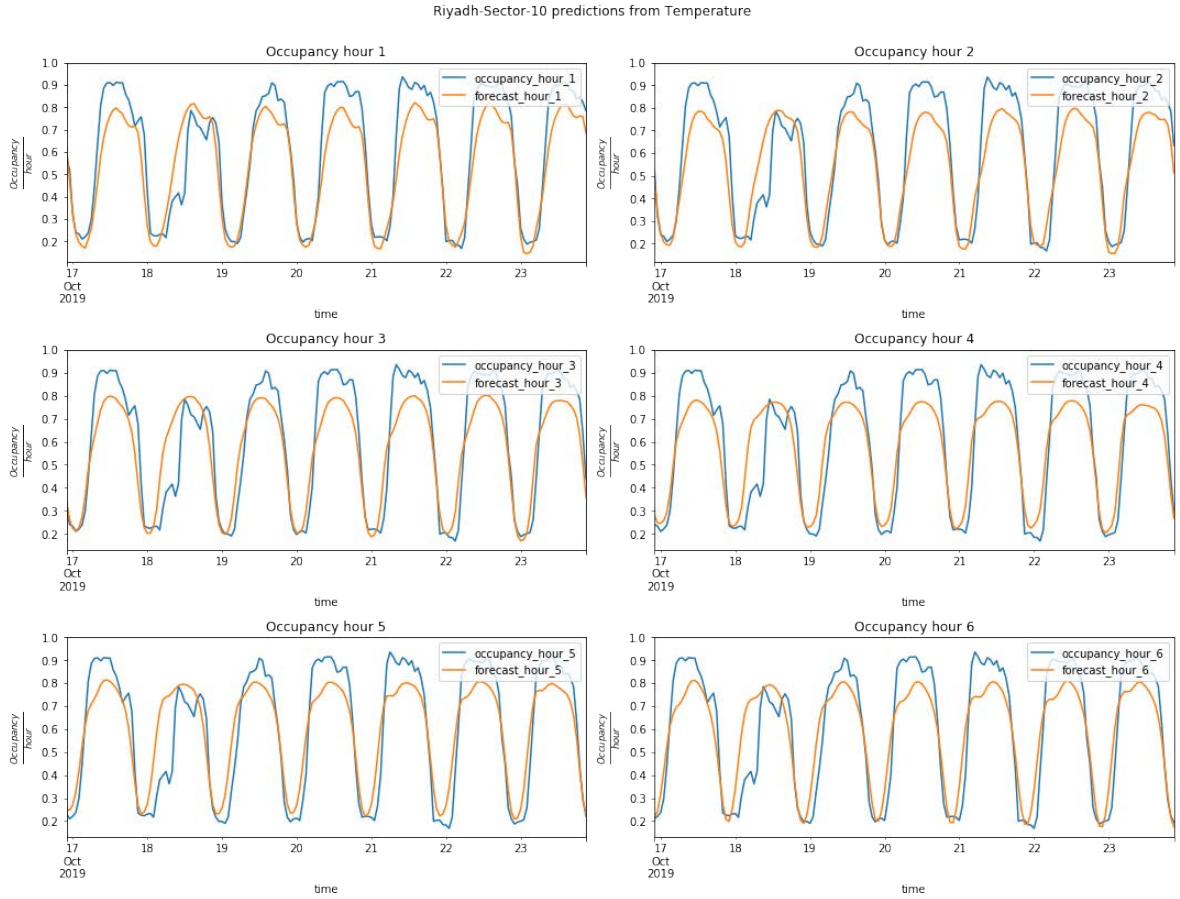


Figure 8.121: Forecasts from the model trained with the exogenous variable Temperature for predicting Sector-10 parking occupancy. Real occupancy in blue and forecasts in orange.

and do not make any distinction between days.

Figure 8.122 shows the time series from a model trained using only the exogenous variable Wind Gust. The visualizations show that there weather conditions could lead to some relationship between Wind Gust values and parking occupancy, but it is hard to discern this.

Table 8.18 provides information about the values of the variables Wind Speed, Precipitation Accumulation and Dew Point to support some of the explanations given.

Figure 8.123 shows the time series of the parking occupancy (real and forecasted) from the model trained with the exogenous variable Wind Speed for the Riyadh scenario. Although some relationship could exist between parking occupancy and Wind Speed under severe weather conditions, a possible interpretation of the visualizations is that the model is projecting the cyclical patterns of the wind in order to forecast the parking occupancy. Therefore, it is hard to discern how useful the information provided by the exogenous variable can be.

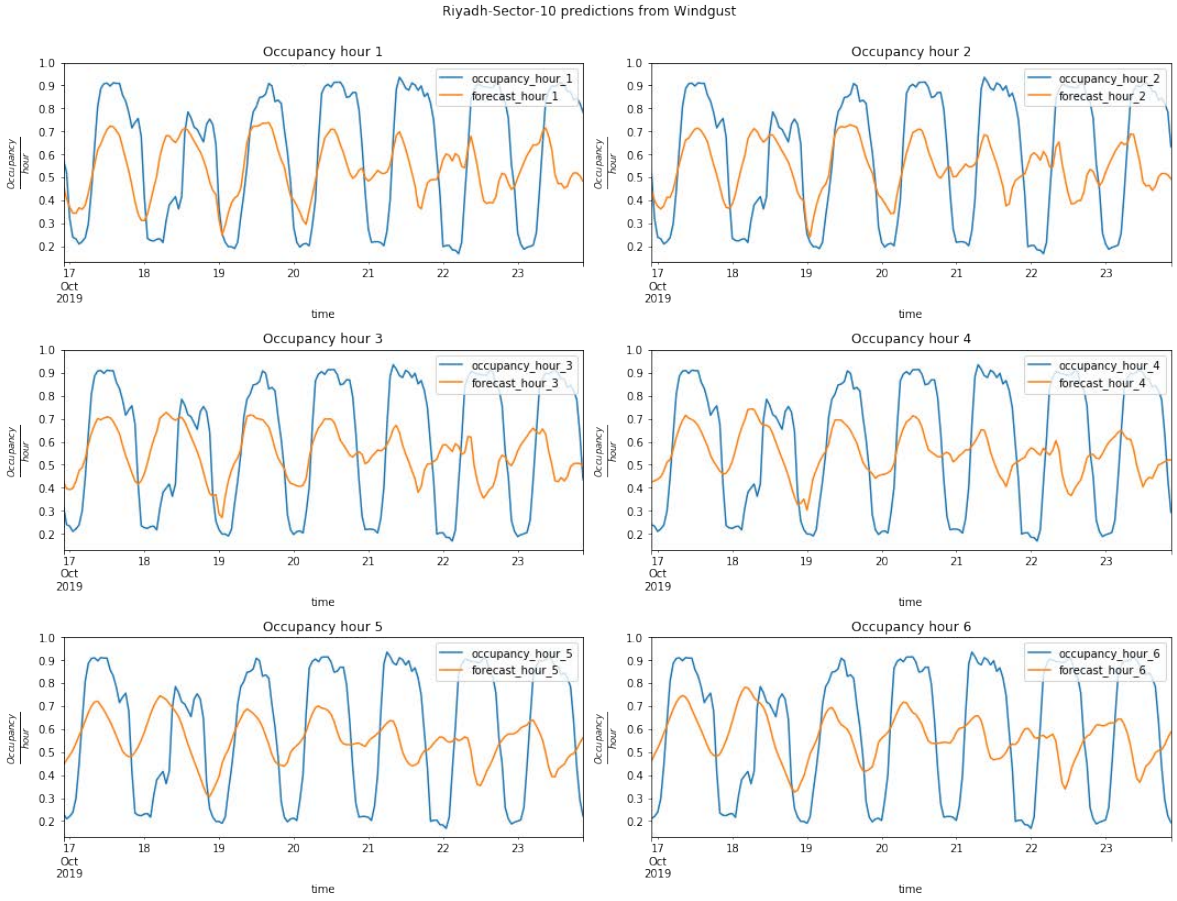


Figure 8.122: Sector-10 (Riyadh) time series of the parking occupancy (blue) and the forecasts (orange) of a model trained with the exogenous variable Wind Gust.

time	windSpeed	precipAccumulation	dewPoint
2019-10-18 00:00:00	0.071369	0.0	0.493378
2019-10-18 01:00:00	0.075519	0.0	0.498916
2019-10-18 02:00:00	0.081328	0.0	0.501324
2019-10-18 03:00:00	0.098755	0.0	0.496749
2019-10-18 04:00:00	0.170954	0.0	0.493378
2019-10-18 05:00:00	0.164315	0.0	0.499157
2019-10-18 06:00:00	0.160166	0.0	0.505659
2019-10-18 07:00:00	0.163485	0.0	0.516735
2019-10-18 08:00:00	0.169295	0.0	0.524199
2019-10-18 09:00:00	0.176763	0.0	0.532627
2019-10-18 10:00:00	0.185892	0.0	0.529497
2019-10-18 11:00:00	0.199170	0.0	0.515531
2019-10-18 12:00:00	0.228216	0.0	0.497231
2019-10-18 13:00:00	0.371784	0.0	0.481580
2019-10-18 14:00:00	0.371784	0.0	0.481580
2019-10-18 15:00:00	0.423237	0.0	0.498916
2019-10-18 16:00:00	0.421577	0.0	0.503973
2019-10-18 17:00:00	0.394191	0.0	0.503732
2019-10-18 18:00:00	0.321162	0.0	0.490970
2019-10-18 19:00:00	0.302075	0.0	0.498676
2019-10-18 20:00:00	0.271369	0.0	0.518420
2019-10-18 21:00:00	0.243983	0.0	0.531423
2019-10-18 22:00:00	0.215768	0.0	0.541055
2019-10-18 23:00:00 7	0.121162	0.0	0.53864

Table 8.18: Riyadh Wind Speed, Precipitation Accumulation and Dew Point values for 18 October 2019

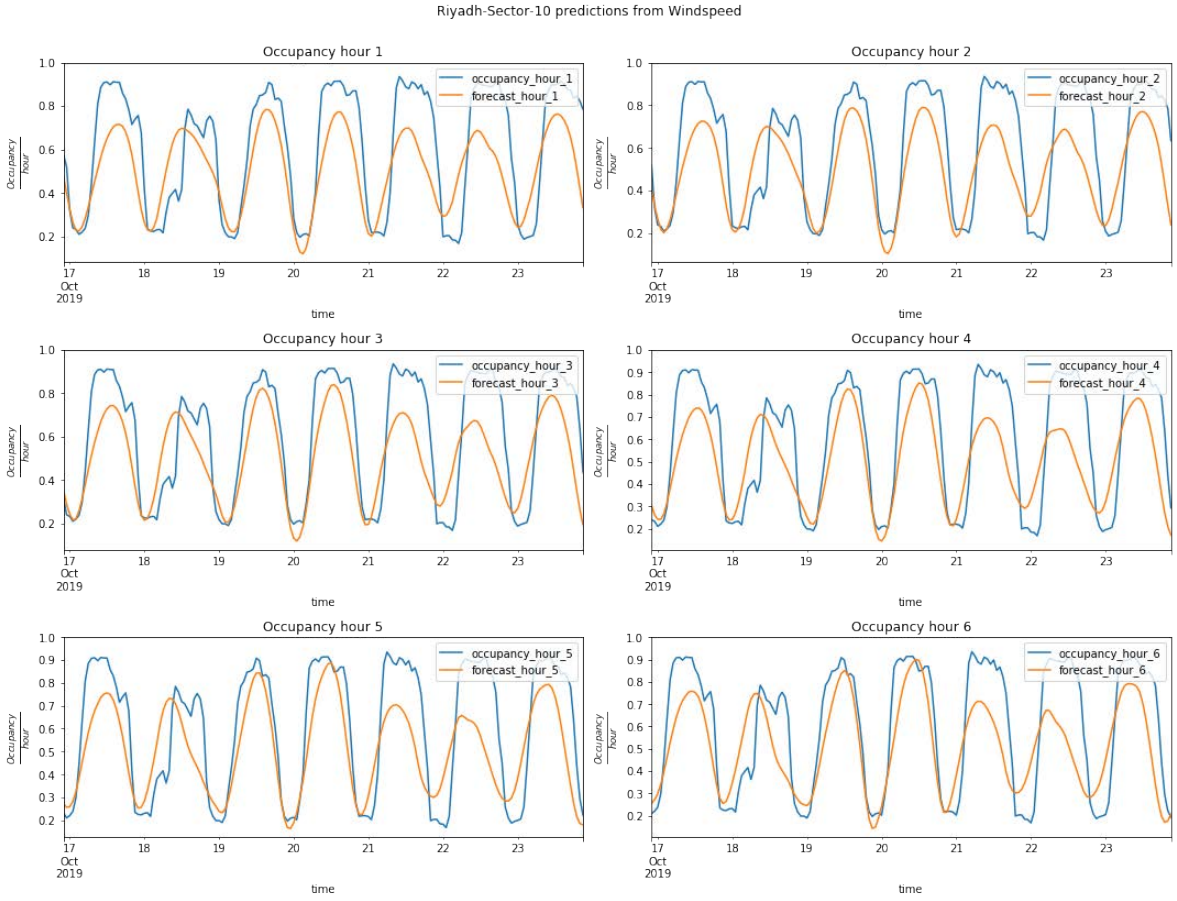


Figure 8.123: Sector-10 (Riyadh) six-hour forecasts of the model trained using the continuous variable Wind Speed.

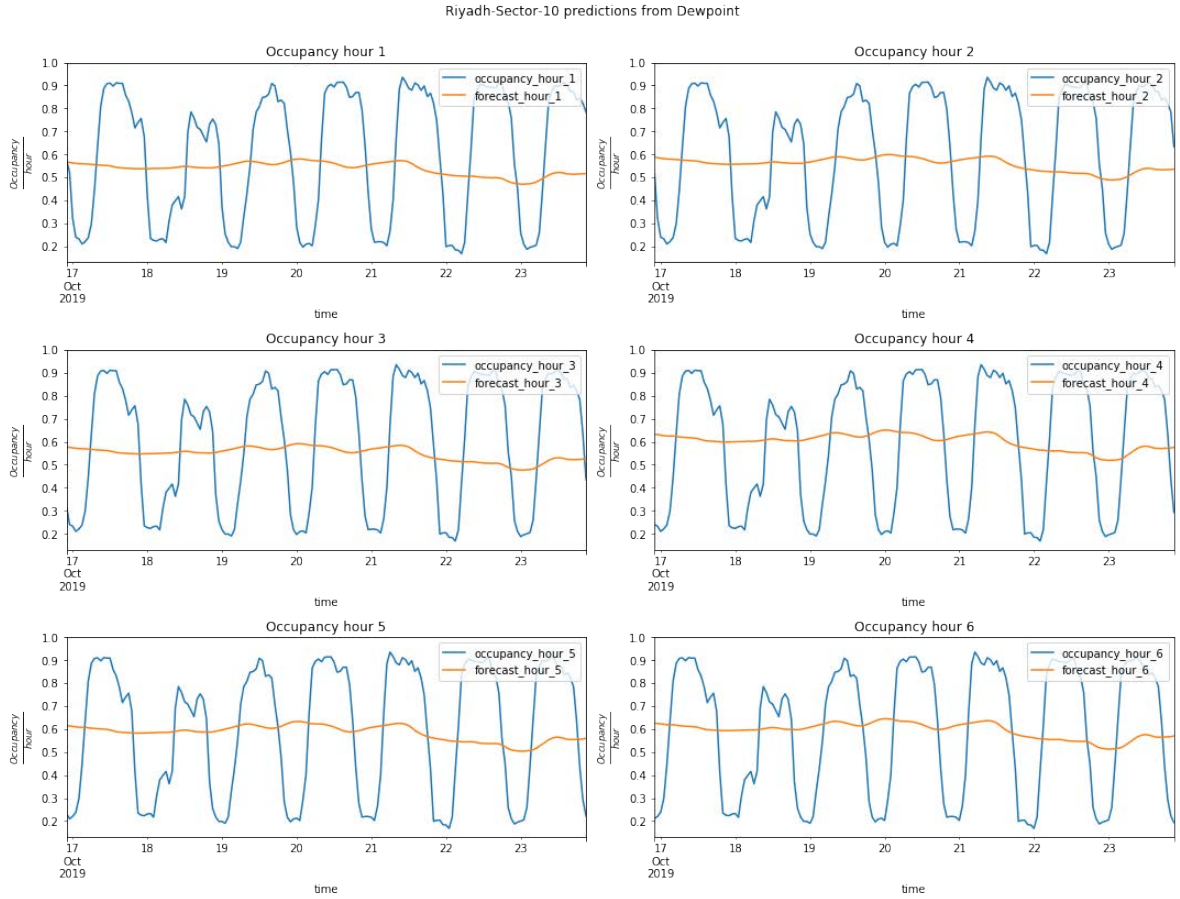


Figure 8.124: Plots of the parking occupancy real and forecasted values from Sector-10 in Riyadh. The model has been trained with the numerical variable Dew Point.

The model trained with Precipitation Accumulation does not offer any useful information, as the variable always has the same value (zero) and therefore the model cannot learn anything.

Similarly to the previous scenario, the exogenous variable Dew Pointis not related to parking occupancy levels, as can be seen in the model's time series of the parking occupation and forecasts in Figure 8.124.

From the models trained with each numerical exogenous variable to predict Riyadh Sector-10 parking occupancy, the following observations and conclusions have been extracted:

- In previous scenarios, a subset of variables was identified as a proxy between sunshine hours and parking occupancy. Those proxy variables provide visualizations that may lead one to think that the variables are correlated. However, those correlations are in fact spurious. In Riyadh, these variables are: Visibility, Cloud Cover, Wind Bearing, UV Index, Humidity and Temperature.
- The variable Precipitation Intensity can occasionally provide information to im-

prove some forecasts, while the variable Precipitation Accumulated does not offer any useful information because its values are always 0 (snow generally never accumulates in Saudi Arabia, for obvious reasons).

- It is hard to conclude if Wind Speed and Wind Gust have any effect on parking occupancy, as their values can be cyclical, just like the proxy variables, thus leading to forecasts based on spurious correlations.
- Dew Point offered no useful information, just as in previous scenarios.

From all the scenarios, the following conclusions have been made about each of the exogenous numerical variables:

- Visibility: No useful interpretation can be made from the visualizations; and when the values are similar to the real occupancy values, the forecasts may possibly be based on spurious correlations.
- Cloud Cover: The same conclusions as for Visibility.
- Wind Bearing: Their forecasts do not seem (visually) correlated or provide useful information for parking forecasts.
- UV Index: It has been shown that models use UV Index information to provide parking occupancy forecasts by exploiting the fact that both variables have similar cycles during the hours of the day. The most possible explanation is that it is a spurious correlation.
- Humidity: The same conclusions as for UV Index. However, changes in the Humidity values could also be related to weather conditions. Therefore, they can carry some useful information.
- Precipitation Intensity: It has been shown that these values have an effect on parking occupancy.
- Precipitation Accumulated: Similar to Precipitation Intensity, but it is not always useful (as in the case of Riyadh).
- Temperature: A spurious correlation that can mask some useful information about the weather conditions.
- Wind Gust: It has been shown that, in some cases, this could provide a spurious correlation, while in other cases it may provide some useful information for parking occupancy.
- Dew Point: Not useful in any of the scenarios.

8.3 Proposed architectures

In order to be able to use all the information provided by the exogenous variables, different architectures are proposed for each of the NN methods because they are constrained

by their nature. The goal is to obtain models that forecast the next six hours of parking occupancy by making use of the presented exogenous variables.

In general, the architectures have the following schema:

- Multiple input layers specialized in handling categorical and continuous data. This could be interpreted as having sub-networks inside the model, which specialize in handling specific types of data. One of the sub-networks specializes in handling the numerical information represented as time series data, while the other sub-network specializes in handling categorical data.
- Each sub-network performs its own computations before merging its outputs. In this way, the models transform the input data into useful representations for later use. Once the outputs of each sub-network are merged (concatenated by the feature dimension), extra computations are made in a new sub-network that has the same structure as the sub-network for the numerical inputs, because all the information is numerical after merging.
- The output layer consists of a dense layer of six neurons, which output the next six hours of parking occupancy.

A random search strategy has been performed in order to decide on the number of hidden layers, neurons and activation functions for the sub-network with the numerical inputs. For the sub-network with the categorical data, no random search is made, as it consists mainly of one embedding layer for each categorical variable, followed by some computational layer that transforms the embeddings so that they can be used in subsequent layers. Other hyperparameters like the optimizer are the same as those obtained in the endogenous models.

8.3.1 MLP proposed architecture

The addition of exogenous variables prevents the MLP architecture from being a good candidate for modeling the data, because it is designed to handle two dimensional tensors, and the input data has become a three-dimensional tensor. One solution is to have as many inputs in the model as there are features in the data. This means that the MLP model has 12 different inputs for the numerical variables (numerical sub-network), and another 8 inputs for the categorical variables (categorical sub-network).

The categorical variables are handled first by embedding layers, then by a Global Average Pooling layer that averages the information across the temporal dimension. This pooling layer also transforms the three-dimensional tensors provided by the embedding layers into two dimensional tensors, which can be concatenated along the feature dimension with the output tensors from the numerical sub-network.

Each of the numerical variables is handled by a subset of dense layers (each variable has its unique layers). The number of these layers and its number of neurons are decided by random search.

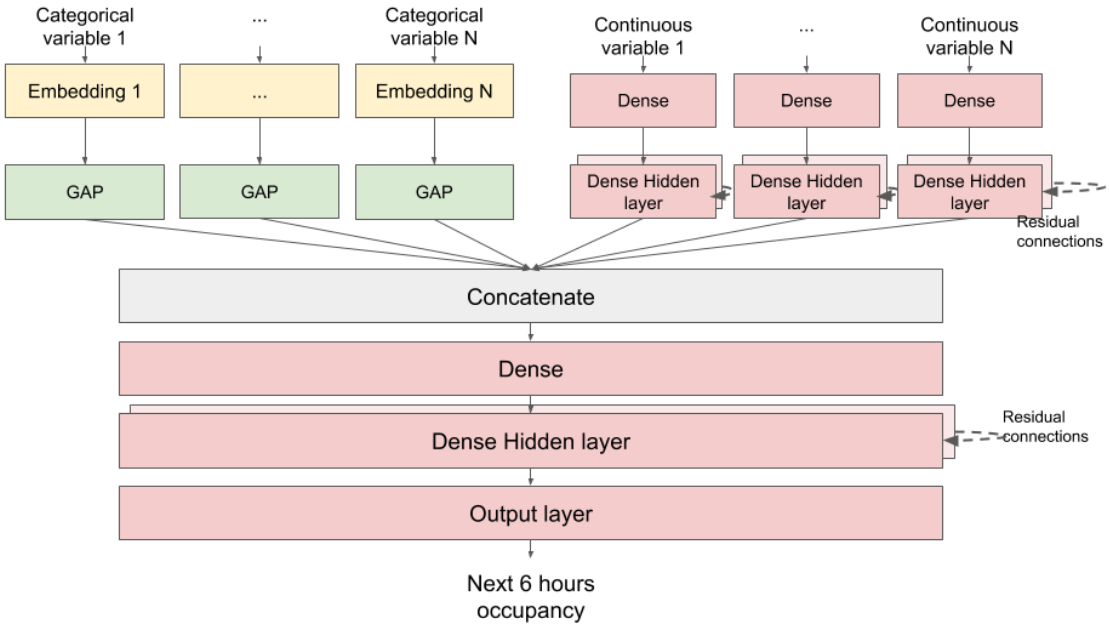


Figure 8.125: Proposed exogenous MLP model architecture.

The outputs from all the branches are merged by concatenating the last dimension (features/columns), thus forming a new two dimensional tensor. This two dimensional tensor is then used as input for a sub-network block that has the same structure as the sub-network for the numerical variables. The output of this block is used as input for the output layer.

In addition, because a large number of layers are used, residual connections are made between the hidden layers of the numerical sub-network. Residual connections take the output from a previous layer and add it to a subsequent layer (although not necessarily the very next one). In doing this, the model can learn to skip some layers if no useful representation of the information is learned. Moreover, residual connections (and skip connections) enhance the training of the model, as they help the gradient flow better to the first layers of the model. Further information about residual connections can be found in He et al. [2016].

One major drawback of the proposed MLP model is that numerical variables cannot be handled all together, and it is possible that patterns shared among variables become impossible to learn. This does not happen with RNN methods, because they can naturally handle multiple sequences (each sequence is a feature) and learn patterns among them in order to provide the forecasts.

The final architecture of the MLP model is presented in Figure 8.125.

8.3.2 Proposed RNN architecture

For the RNN methods (LSTM and GRU), a similar architecture is proposed, although all the numerical variables are used as inputs of the same RNN layer. Therefore, the models can learn patterns that are shared among the numerical inputs. A random search has been performed on only some of the numerical variables. Different architectures have been tested by trial and error, by which we make the following observations after closely monitoring the errors of the models while training:

- Normalization layers after each RNN layer: Not used because this increased the training time.
- Normalization layer after merging;: Similar to the previous one.
- Global Average Pooling of the embeddings in the time dimension (as done in the MLP architecture): Some models obtained slightly similar scores to those obtained by the respective endogenous models.
- RNN layer after each embedding followed by concatenation: Similar to the previous one, but it makes the training slower.
- Concatenating all the outputs of the embeddings with the numerical inputs, then training a RNN sub-network with all this information: Slow, and bad scores.
- Concatenating all the outputs of the embeddings with the output of the sub-network of the numerical inputs, then using the concatenated features as input for a dense layer of 256 neurons before the output layer: The results better than for the endogenous models.
- Concatenating all the outputs of the embeddings with the output of the sub-network of the numerical inputs, then using the concatenated features as input for a new sub-network with a structure similar to the sub-network with numerical inputs: This achieved the best results.
- In addition, testing residual and skip connections found that skip connections improved training time and RMSE by using them between RNN layers.

The final architecture proposed for the RNN methods is presented in 8.126.

8.3.3 Design of experiments

An experiment constitutes a modeling process for each sector in each city using a specific NN method layer type for the hidden layers in each of the proposed architectures. The NN methods considered are:

- Multilayer perceptron (MLP).
- The recurrent neural networks LSTM and GRU.

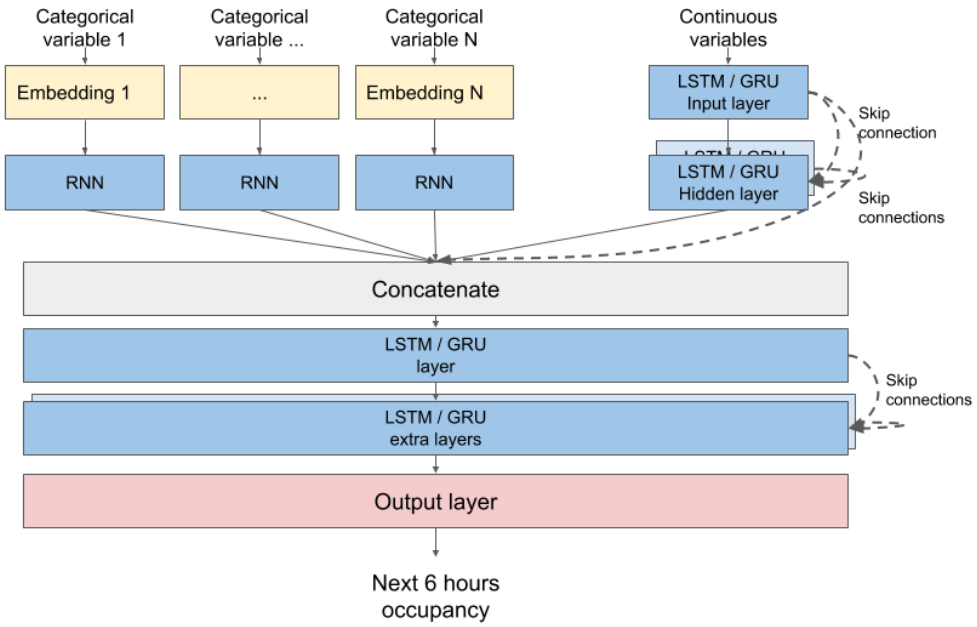


Figure 8.126: Proposed exogenous RNN model architecture.

City	Sector	Experiment name	Comments
Antwerp	Pelikaanstraat	MLP_exogenous_antwerp_pelikaanstraat	Noisy sector that combines regular global patterns with non-regular local ones.
		LSTM_exogenous_antwerp_pelikaanstraat	
		GRU_exogenous_antwerp_pelikaanstraat	
Wattens	Outside	MLP_exogenous_wattens_outside	Small–medium sized sector. Both Wattens sectors are similar, but endogenous models of the Outside sector achieved far worse scores for no reason.
		LSTM_exogenous_wattens_outside	
		GRU_exogenous_wattens_outside	
Los Angeles	Standard	MLP_exogenous_LA_standard	2018 and 2019 have different occupancy values because of street works.
		LSTM_exogenous_LA_standard	
		GRU_exogenous_LA_standard	
Riyadh	Sector-10	MLP_exogenous_riyadh_sector10	Large sized sector. Endogenous models scored worse in Sector-10 in comparison to Sector-1, although they have similar occupancy patterns.
		LSTM_exogenous_riyadh_sector10	
		GRU_exogenous_riyadh_sector10	

Table 8.19: Table of NN experiments.

For each experiment, the outcome is the best model from each method within the same validation framework presented in 7.2. Comments are given about facts that emerged during the execution, and we also provide the RMSE values of the best model performances throughout the different forecasting steps (from t to $t+6$). Remember that, in order to reduce the stochasticity of the NN models, we use the best architecture from each experiment to re-create the model 20 times. The presented results correspond to the mean RMSE values of each forecasting step.

Table 8.19 shows the experiments, with their names and comments about the scenarios (city and sectors).

Questions that arise from the experiments are:

- What are the effects of the exogenous variables on the RMSE?
- Which exogenous variables contribute more to providing better forecasts?

- Which exogenous variables do not contribute or make the forecasts worse?
- Is it worth using an exogenous model instead of an endogenous one?
- Can the models be improved?

A random search has been made for some parts of the model (those that concern the numerical inputs of the model). The range of values for the hyperparameters are the same as those in Chapter 7, with the addition of the hyperbolic tangent activation function.

8.4 Antwerp experiments

Table 8.20 presents the date range for the training, validation and test sets for the experiment in the Antwerp scenario. The models provide forecasts for the Pelikaanstraat sector. We consider this sector because the exogenous variables are expected to help the models reduce the noise in the endogenous data while providing better and smoother forecasts.

8.4.1 MLP

Table 8.21 presents the hyperparameters obtained by random search when creating the MLP model trained with exogenous variables for the Pelikaanstraat sector in the Antwerp scenario. These hyperparameters are used to define both the sub-network with the numerical inputs and the sub-network that occurs after the concatenate operation.

Table 8.22 presents the RMSE scores for each hour and the mean RMSE for the MLP model trained with exogenous variables for the Antwerp scenario.

A visualization of each forecasting step from t to $t+6$ on the date interval of the test set is presented in Figure 8.127. It can be observed that the model forecasts for forecast step one (next hour from now) are accurate. At later forecast steps, the model forecasts are smoother, but the model still provides different occupancy levels based on the week of the day. For example, the model predicts higher occupancy values for Sunday afternoons (March 31 and April 7 and 14) than for the other days.

In order to assess the findings from Section 8.1, the same model has been trained with a subset of the variables. The variables of this subset are selected according to the scenario characteristics. For the Antwerp scenario, the chosen exogenous variables are:

Sector	Train	Validation	Test
Pelikaanstraat	2018-02-01, 2019-01-19	2019-01-20, 2019-03-04	2019-03-05, 2019-04-17

Table 8.20: Antwerp sector intervals of dates used for the training, validation and test sets.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_exogenous_antwerp_pelikaanstraat	16	3	0	He normal	Tanh	32	RMSprop	0.0013431

Table 8.21: Hyperparameter values obtained by the random search for the Antwerp MLP model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_antwerp_pelikaanstraat	0.116387	0.135267	0.147768	0.157503	0.164878	0.173999	0.150557

Table 8.22: Antwerp MLP exogenous model's RMSE values obtained in the test set.

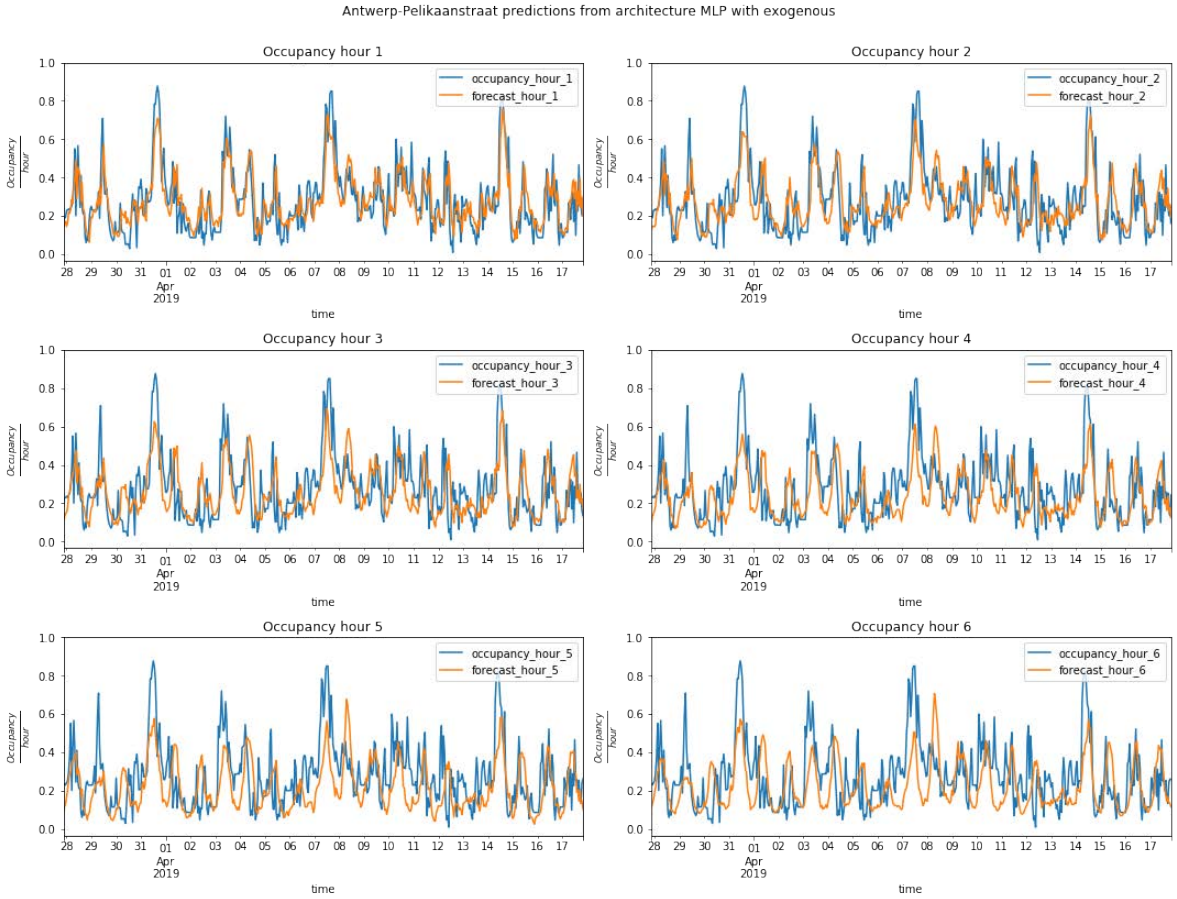


Figure 8.127: Forecast visualizations for the Antwerp Pelikaanstraat MLP model trained with exogenous information. From top left to bottom right are steps one to six

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_antwerp_pelikaanstraat	0.115349	0.138901	0.153771	0.165558	0.173725	0.180106	0.156209

Table 8.23: Antwerp MLP model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of the exogenous variables.

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.116387	0.135267	0.147768	0.157503	0.164878	0.173999
Subset of exogenous variables	0.115349	0.138901	0.153771	0.165558	0.173725	0.180106

Table 8.24: Antwerp RMSE comparison of MLP exogenous models trained with all the exogenous variables and with only a subset of them.

- Summary.
- Precipitation Type.
- Day of the Week.
- Month.
- Holiday.
- Precipitation Intensity.
- Precipitation Accumulated.

If better RMSE scores are obtained by this model, then some of the exogenous variables add noise to the model when it is trained with all of them. If not, this could be indicative that NN methods for time series are robust to noise and/or non-informative features. On the other hand, if the RMSE scores are similar, then the subset of variables is the one from which the model learns.

The RMSE scores obtained from the model trained with the subset of exogenous variables are presented in Table 8.23.

Table 8.24 presents a side-by-side comparison of the RMSE scores obtained by the model trained with all the exogenous variables and those from the model trained with only the subset. It can be seen that the model trained with only a subset of the exogenous variables presents similar or higher RMSE values, especially from hour 3 onward.

8.4.2 LSTM

Table 8.25 presents the hyperparameter values obtained by the random search for the LSTM sub-network for numerical inputs and the sub-network after the concatenate operation.

Table 8.26 presents the the RMSE obtained for the successive hours one to six and the mean RMSE for the LSTM model.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_exogenous_antwerp_pelikaanstraat	512	1	0.125	0.125	Glorot normal	ReLu	32	SGD	0.045333421	0.7

Table 8.25: Hyperparameter values obtained by random search for the Antwerp LSTM model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_exogenous_antwerp_pelikaanstraat	0.112092	0.128344	0.135903	0.141196	0.145912	0.150256	0.136307

Table 8.26: Antwerp LSTM exogenous model's RMSE values obtained in the test set.

Figure 8.128 presents the visualizations of the LSTM model at each forecasting step, from step $t = 1$ to step $t + 6$ when making forecasts from step $t = 0$. The results show very good forecasts at forecasting steps one, two and three. After those, the forecasts are smooth and more conservative (lower values at the peak and higher values in the valleys). One remarkable point is that the model forecasts for Sundays are higher than the other days, independently of the forecasting step, showing that the model has learnt perfectly the daily dynamics of parking occupancy.

The RMSE scores obtained from the model trained with the selected subset of exogenous variables are presented in Table 8.27.

Table 8.28 presents a side-by-side comparison of the RMSE scores obtained by the model trained with all the exogenous variables and the model trained with only the subset. The results show that the model trained with the subset of exogenous variables has obtained the lowest error in forecasting steps one and two, with a large margin in forecasting step one. From forecasting step three onward, the LSTM model trained with all the exogenous variables shows lower error than the others. This could be because some of the exogenous variables are adding noise in the early steps of the forecasts while helping in the later steps.

8.4.3 GRU

Table 8.29 presents the hyperparameter values obtained by the random search for the GRU sub-networks .

The obtained RMSE scores for each forecasting step of the GRU model trained with exogenous variables are presented in Table 8.30. The results show that the model error increases greatly between forecasting step one and two when compared with the other forecasting steps.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_exogenous_antwerp_pelikaanstraat	0.099110	0.122019	0.133387	0.141840	0.147803	0.152289	0.133957

Table 8.27: Antwerp LSTM model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of the exogenous variables.

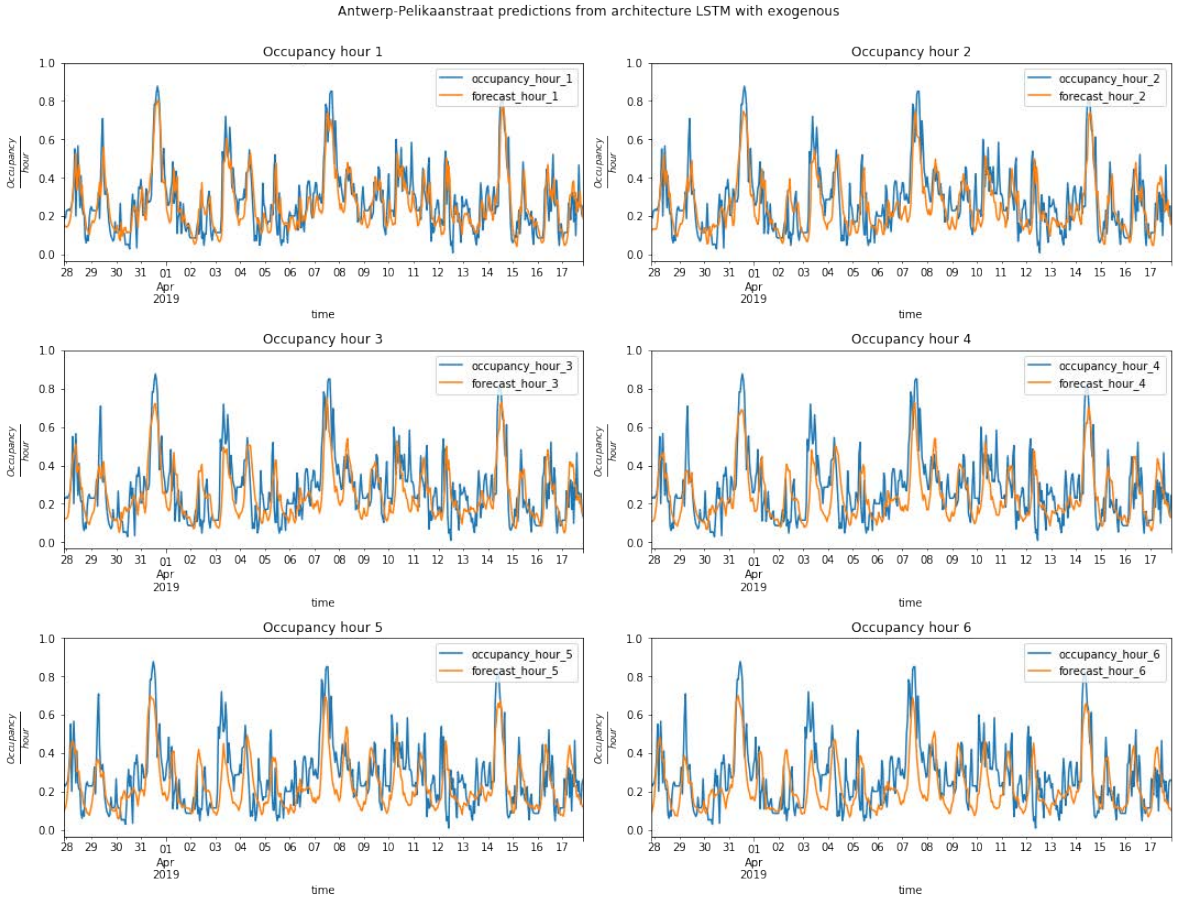


Figure 8.128: Antwerp Pelikaanstraat LSTM model trained with hourly forecast visualizations of the exogenous variables. From top left to bottom right are steps one to six

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.112092	0.128344	0.135903	0.141196	0.145912	0.150256
Subset of exogenous variables	0.099110	0.122019	0.133387	0.141840	0.147803	0.152289

Table 8.28: Antwerp RMSE comparison of LSTM exogenous models trained with all the exogenous variables and with only a subset of them.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_exogenous_antwerp_pelikaanstraat	256	1	0	0	Glorot normal	Tanh	256	RMSProp	0.0002656	0.4

Table 8.29: Hyperparameter values obtained by the random search for the Antwerp GRU model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_exogenous_antwerp_pelikaanstraat	0.114243	0.140703	0.143994	0.150458	0.158148	0.164877	0.14648

Table 8.30: Antwerp GRU exogenous model's RMSE values obtained in the test set.

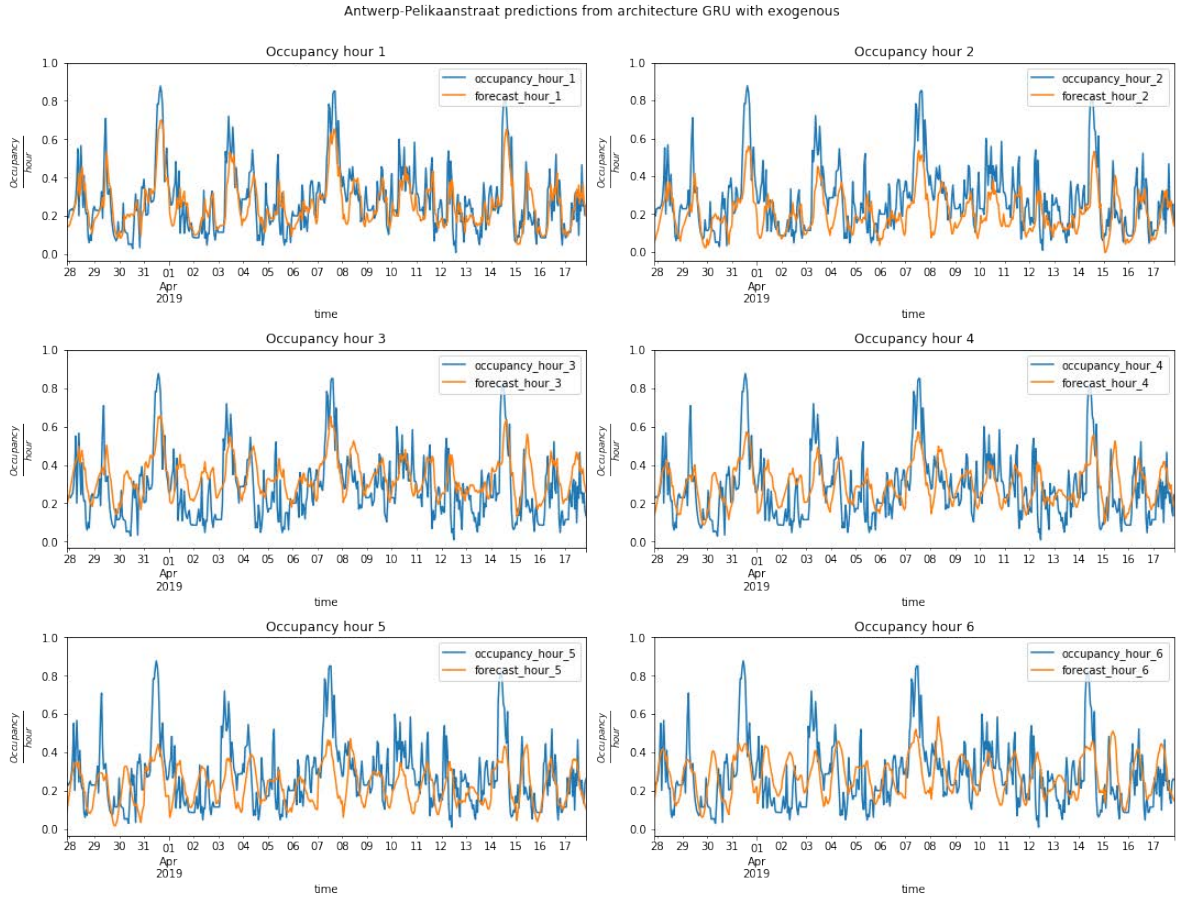


Figure 8.129: Forecast visualizations for the Antwerp Pelikaanstraat GRU model with exogenous information. From top left to bottom right are steps one to six.

The visualizations of the forecasts at horizon steps one to six are presented in Figure 8.129 for the Pelikaanstraat GRU model. The forecast for the next hour (occupancy hour 1 visualization) presents a low error rate and clearly discriminates the daily patterns. An example of the daily discrimination can be found on days 7 and 14 of April, which correspond to Sundays and have different occupancy patterns. Forecasts become smoother as the forecasting hour extends further in time. However, the daily patterns are maintained instead of becoming the mean of the series (something that was observed in the endogenous models).

As we have done with the previous methods, Table 8.31 shows the GRU model's RMSE scores when using a subset of exogenous variables.

Table 8.32 shows a side-by-side comparison of the obtained RMSE scores. The scores show that the model trained with the subset of exogenous variables has a higher RMSE than the one that uses them all.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.124101	0.142577	0.160875	0.160392	0.166151	0.169904	0.155257

Table 8.31: Antwerp GRU model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of exogenous variables.

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.114243	0.140703	0.143994	0.150458	0.158148	0.164877
Subset of exogenous variables	0.124101	0.142577	0.160875	0.160392	0.166151	0.169904

Table 8.32: Antwerp RMSE comparison of GRU exogenous models trained with all the exogenous variables and with only a subset of them.

8.5 Wattens experiments

Table 8.33 presents the date range for the training, validation and test sets of the Wattens experiment with exogenous variables. The Outside sector has been chosen because the endogenous models provided high forecasting error rates when compared to the Inside sector's error rate, although some similarities occurred. It is expected that exogenous information helps the model better discriminate the occupancy patterns based on the day of the week.

8.5.1 MLP

The hyperparameter values of the MLP candidate models obtained by random search for the Wattens scenario are presented in Table 8.34. The model is trained using all the exogenous variables presented in Section 8.1.

Table 8.35 presents the scores for the MLP experiments in the Wattens scenario. They show low error rates for forecasting step one, followed by increases at each successive time step. However, the increase in error is more noticeable from steps one to two.

Figure 8.130 offers visualizations of the MLP candidate model for the Outside sector in Wattens at different forecasting horizons (one to six). The visualizations for each forecasting hour show that the model perfectly discriminates the occupancy based on the day of the week. For example, it shows less occupancy on weekdays and weekends as well as different levels for Mondays and Fridays. Another interesting observation is

Sector	Training	Validation	Test
Outside	2018-03-01, 2019-02-23	2019-02-24, 2019-04-09	2019-04-10, 2019-05-23

Table 8.33: Wattens sector intervals of dates used for the training, validation and test sets.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_exogenous_wattens_outside	16	2	0	Glorot normal	Tanh	128	RMSprop	0.00682913

Table 8.34: Hyperparameter values obtained by random search for the Wattens MLP model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_wattens_outside	0.086194	0.103884	0.115519	0.123145	0.126526	0.130674	0.11554

Table 8.35: Wattens MLP exogenous model's RMSE values obtained in the test set.

that the model differentiates occupancies on Holidays, as seen on 1 May (Labour day), when the forecasted occupancy is lower than for a normal Wednesday.

In order to assess the findings from Section 8.1, the same model has been trained with a subset of the variables, which were selected according to the scenario characteristics. For the Wattens scenario, the chosen exogenous variables are:

- Summary.
- Precipitation Type.
- Day of the Week.
- Month.
- Holiday.
- Precipitation Intensity.
- Precipitation Accumulated.
- Wind Gust.
- Wind Speed.

The RMSE scores obtained from the model trained with the subset of exogenous variables are presented in Table 8.36. The obtained RMSE scores indicate a lower increase in the error rate for the latest hours in comparison to the earliest ones.

In order to compare the RMSE values obtained by both models for each hour, a side-by-side table of results is presented in Table 8.37. The results show that the forecasts of the model trained with a subset of the exogenous variables presents lower RMSE values for the first time steps, but from Hour 3 onward the model with all the exogenous variables has the lowest error rate. The means for the 6 time steps are similar in both models. Therefore, it is possible that both models are nearly equivalent.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.070058	0.097697	0.115905	0.126067	0.130791	0.132521	0.114405

Table 8.36: Wattens MLP model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of the exogenous variables.

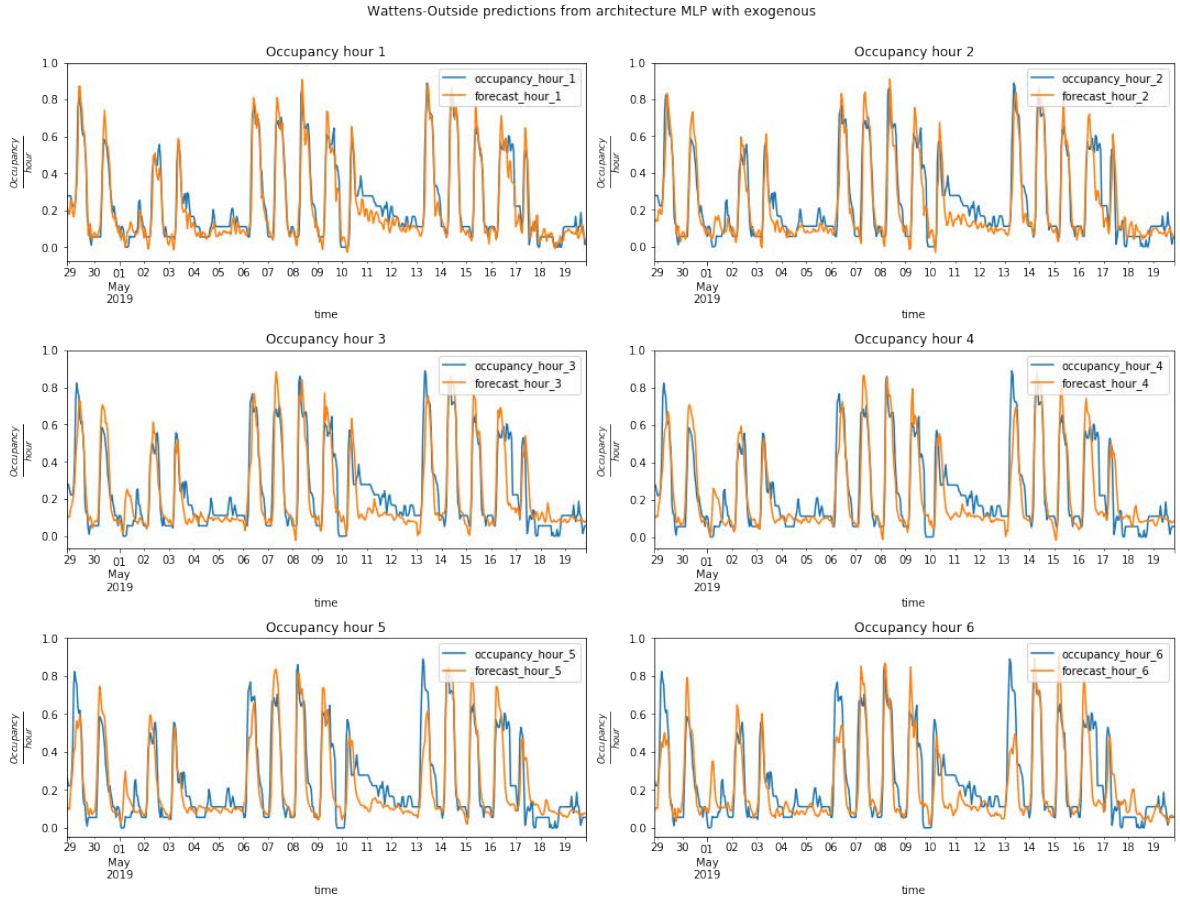


Figure 8.130: Forecast visualizations for the Wattens Outside sector MLP model with exogenous variables. From top left to bottom right are steps one to six

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.086194	0.103884	0.115519	0.123145	0.126526	0.130674
Subset of exogenous variables	0.070058	0.097697	0.115905	0.126067	0.130791	0.132521

Table 8.37: Wattens RMSE comparison of MLP exogenous models trained with all the exogenous variables and with only a subset of them.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_exogenous_wattens_outside	64	2	0.25	0.125	Glorot normal	Linear	256	SGD	0.00344656	1

Table 8.38: Hyperparameter values obtained by random search for the Wattens LSTM model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_exogenous_wattens_outside	0.099849	0.115229	0.125736	0.132120	0.135185	0.139255	0.125309

Table 8.39: Wattens LSTM exogenous model's RMSE values obtained in the test set.

8.5.2 LSTM

Table 8.38presents the hyperparameter values obtained by random search for the Wattens Outside sector model that was trained using exogenous variables.

Table 8.39 presents the scores for the LSTM experiments in the Wattens scenario. Observe that the error rate increases faster for the first forecasting hours than for the later ones.

The visualizations of the parking occupancy forecasts at step horizons one to six are presented in Figure 8.131 for the Wattens Outside sector. The model has been trained using all the exogenous variables. The visualizations show that the model forces the forecasts to be more in line with the global patterns and less so with the local ones. This is observed on days 11 and 12 of May, where the true parking occupancy presents unusual parking occupancy levels for weekend days, but the model forecasts the usual daily cycle with lower values of occupancy. The reason for this reduced flexibility could be related to the dropout values used.

The RMSE scores obtained from the model trained with the subset of exogenous variables are presented in Table 8.40. The RMSE scores show that the error increase between successive forecasting steps goes from greater to lower, resulting in a very small difference between the RMSE scores of hours five and six.

Table 8.41 presents the RMSE from the model trained with all the exogenous variables and the model trained with the subset. The results show that both have similar RMSE scores at the different forecasting steps, although the model with all the exogenous information is a bit higher in the first hours (until hour 3), then becomes a bit lower for the following forecasting steps.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.092725	0.112870	0.127217	0.136021	0.140750	0.145832	0.12723

Table 8.40: Wattens LSTM model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of the exogenous variables.

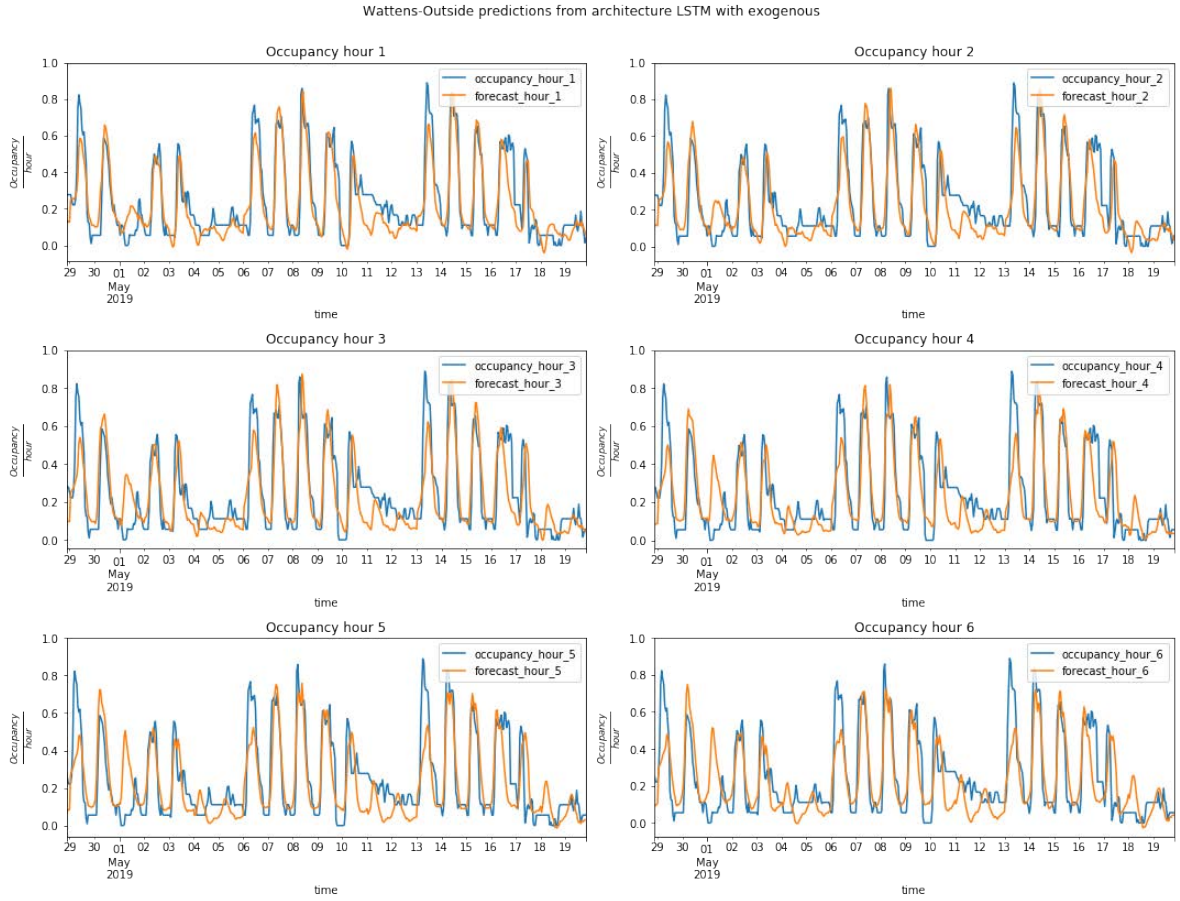


Figure 8.131: Forecast visualizations for the Wattens Outside sector LSTM model with exogenous variables. From top left to bottom right are steps one to six

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.099849	0.115229	0.125736	0.132120	0.135185	0.139255
Subset of exogenous variables	0.092725	0.112870	0.127217	0.136021	0.140750	0.145832

Table 8.41: Wattens RMSE comparison of LSTM exogenous models trained with all the exogenous variables and with only a subset of them.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_full_exogenous_wattens_outside	128	1	0.25	0	Glorot normal	Linear	256	RMSProp	0.00001349	0.4

Table 8.42: Hyperparameter values obtained by random search for the Wattens GRU model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_exogenous_wattens_outside	0.099434	0.110648	0.117117	0.121229	0.123851	0.125638	0.116672

Table 8.43: Wattens GRU exogenous model's RMSE values obtained in the test set.

8.5.3 GRU

The hyperparameter values obtained by the random search for the GRU model trained with all the exogenous variables are presented in Table 8.42.

Table 8.43 presents the RMSE values for each hour of the forecast horizon (six hours). The error rate increases considerably between forecasting steps one and two, and less so for the following steps.

Figure 8.132 presents visualizations of the real and forecasted (hour by hour) parking occupancy of the Wattens Outside sector. The visualizations show that the model accurately forecasts the daily patterns of the parking sector. Similarly to what has been observed in the LSTM case, the model fails to accurately predict days 11 and 12 of May, as there was more occupancy than expected for a weekend. On the other hand, the model takes into account holiday information and predicts lower occupancy levels for Wednesday, the first of May, which was Labour Day.

The RMSE scores obtained from the model trained with the subset of exogenous variables are presented in Table 8.44. The RMSE scores show similar behaviour throughout the hours, just as when the model is trained with all the exogenous variables.

In order to compare the RMSE values obtained by both models, a side-by-side table of results is presented in Table 8.45, which shows the scores for each forecasting step. It can be seen that the error rates of both models are similar in the first forecasting steps, but for the later ones, the model with all the exogenous variables shows lower RMSE.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.097499	0.117115	0.128949	0.134831	0.138059	0.139208	0.126804

Table 8.44: Wattens GRU model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of the exogenous variables.

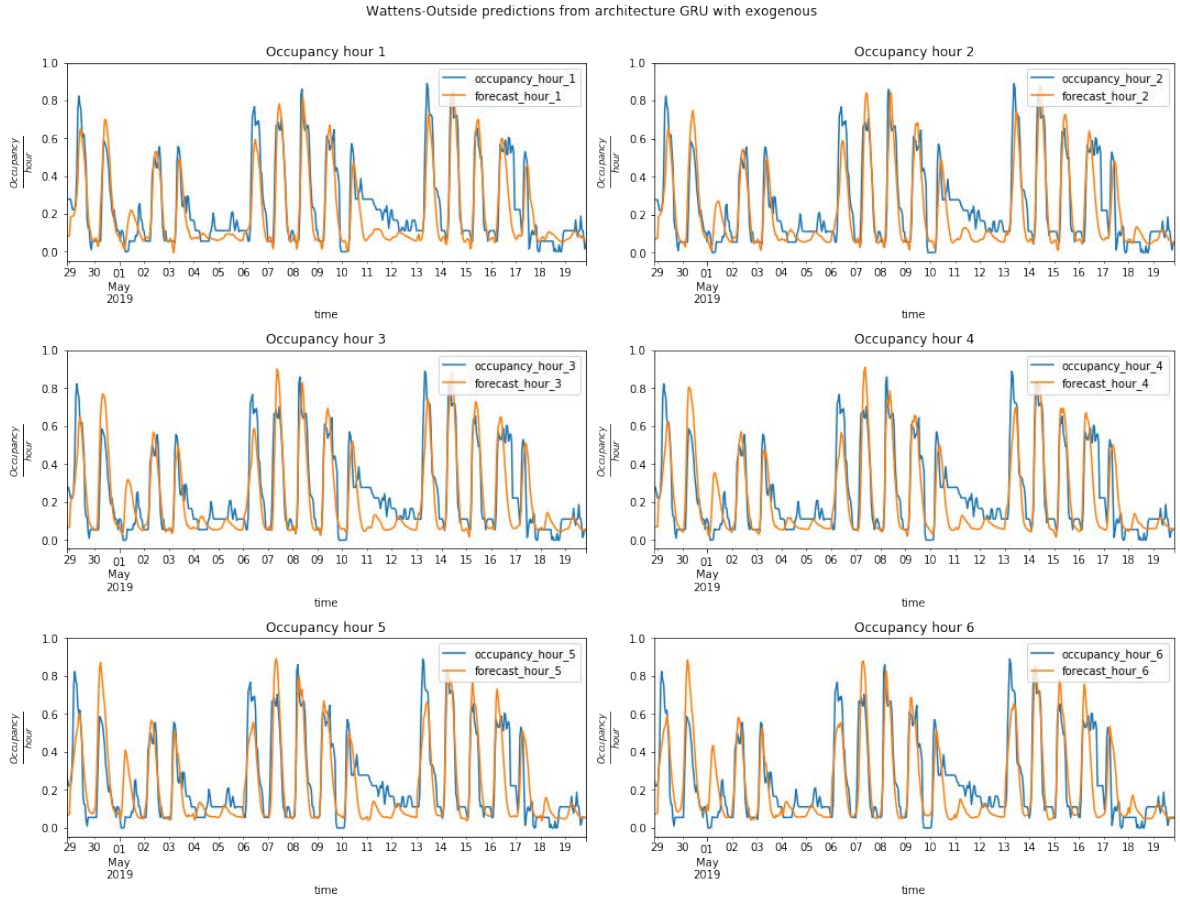


Figure 8.132: Forecast visualizations for the Wattens Outside sector GRU model with exogenous variables. From top left to bottom right are steps one to six

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.099434	0.110648	0.117117	0.121229	0.123851	0.125638
Subset of exogenous variables	0.097499	0.117115	0.128949	0.134831	0.138059	0.139208

Table 8.45: Wattens RMSE comparison of GRU exogenous models trained with all the exogenous variables and with only a subset of them.

Sector	Training	Validation	Test
Standard	2018-04-01, 2019-07-01	2019-07-02, 2019-08-27	2019-08-28, 2019-10-23

Table 8.46: Los Angeles sector intervals of dates used for the training, validation and test sets.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_exogenous_LA_Standard	64	2	0.125	He normal	ReLu	256	RMSprop	0.007757

Table 8.47: Hyperparameter values obtained by random search for the Los Angeles MLP model trained with exogenous variables.

8.6 Los Angeles experiments

Table 8.46 presents the date range for the training, validation and test sets for each of the Los Angeles Standard sector's experiments with exogenous variables. The models were built to study the effects of the exogenous variables in time series with temporal distribution changes.

8.6.1 MLP

Table 8.47 shows the hyperparameter values obtained by the random search from the MLP experiments in the city of Los Angeles.

Table 8.48 presents the RMSE scores obtained from the model MLP trained with exogenous variables for the Los Angeles scenario. All the forecasting hours present similar RMSE values.

Figure 8.133 shows the forecasts for the six different forecasting steps of the Standard sector's candidate model. The visualizations show that the model has adapted somewhat to the street works. However, at later forecasting hours, the provided forecasts are of bad quality.

The subset of exogenous variables for training a model with fewer exogenous variables based on the scenario characteristics are:

- Summary.
- Precipitation Type.
- Month.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_endogenous_LA_Standard	0.133507	0.132600	0.135605	0.134806	0.138004	0.137220	0.135734

Table 8.48: Los Angeles MLP exogenous model's RMSE values obtained in the test set.

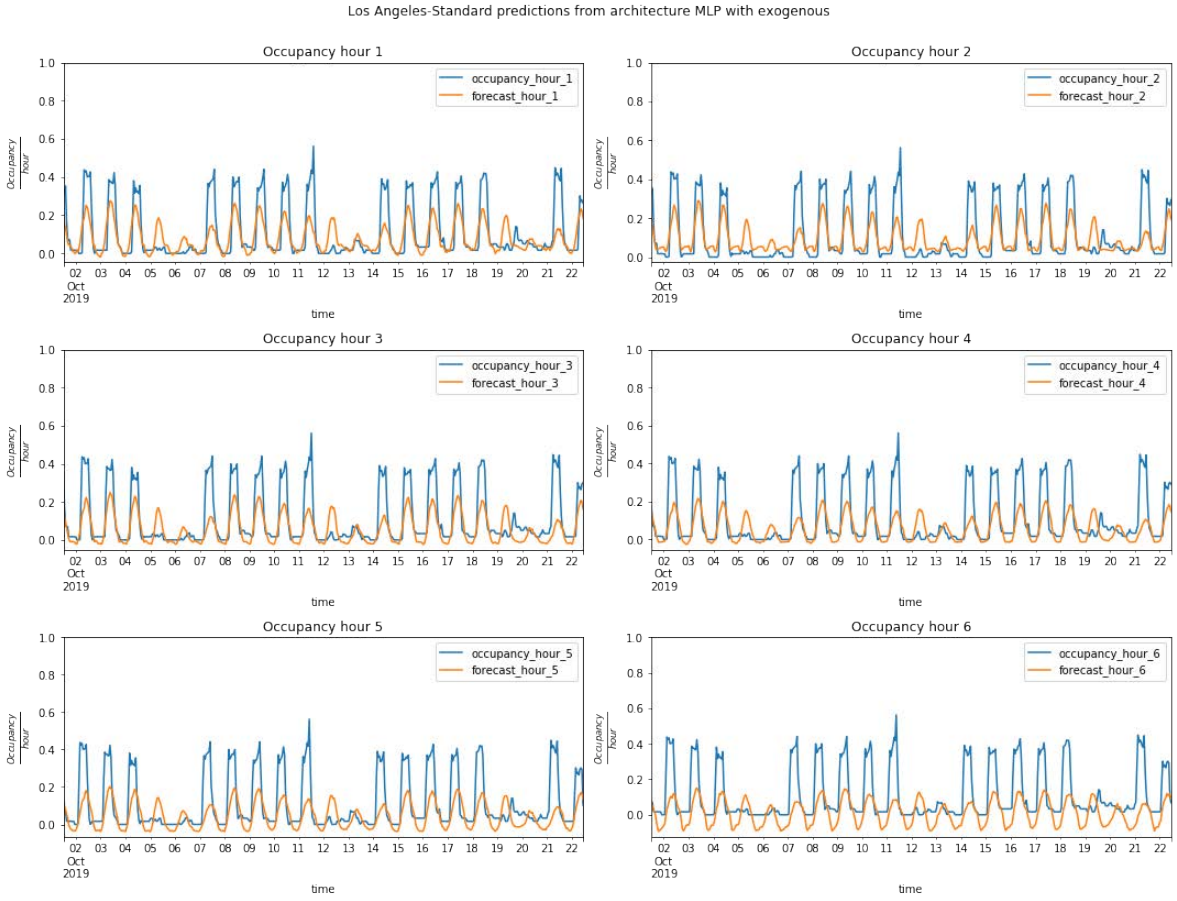


Figure 8.133: Forecast visualizations for the Los Angeles Standard sector MLP model with exogenous variables. From top left to bottom right are steps one to six.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.117905	0.122205	0.131001	0.130840	0.132634	0.134908	0.128791

Table 8.49: Los Angeles MLP model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of the exogenous variables.

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.133507	0.132600	0.135605	0.134806	0.138004	0.137220
Subset of exogenous variables	0.117905	0.122205	0.131001	0.130840	0.132634	0.134908

Table 8.50: Los Angeles RMSE comparison of MLP exogenous models trained with all the exogenous variables and with only a subset of them.

- Holiday.
- Day of the Week.
- Precipitation Intensity.

The RMSE scores obtained from the model trained with the subset of exogenous variables are presented in Table 8.49. The obtained results show that the model with a subset of the exogenous variables has the lowest error rate in forecasting hours one and two. For the following hours, the error is very similar and the increase is small.

In order to compare the RMSE values obtained by both models, a side-by-side table of results is presented in Table 8.50, which shows the scores for each forecasting step. It can be seen that the error rates of both models are similar in the first forecasting steps, but for the later ones, the model with all the exogenous variables shows lower RMSE.

8.6.2 LSTM

LSTM hyperparameter values of the model trained with exogenous information, which were obtained by random search are presented in Table 8.51.

Table 8.52 shows the RMSE values obtained for the LSTM model trained with all the exogenous variables. It can be observed that the error barely increases in the later forecasting steps, specifically from forecasting step four onward.

The visualizations of the forecasts at different forecasting horizons (from one to six hours) are presented in Figure 8.134. The visualizations show that the model forecasts

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_exogenous_LA_Standard	1024	1	0.125	0.25	He normal	ReLu	64	SGD	0.005	0.4

Table 8.51: Hyperparameter values obtained by random search for the Los Angeles LSTM model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_exogenous_LA_Standard	0.099506	0.118512	0.121639	0.131558	0.132344	0.132003	0.123919

Table 8.52: Los Angeles LSTM exogenous model's RMSE values obtained in the test set.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.085215	0.105905	0.116418	0.129116	0.138504	0.139243	0.121193

Table 8.53: Los Angeles LSTM model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of exogenous variables.

are close to the real parking occupancy values for the first forecasting steps, but at the later ones, the forecasts show that there is no distinction in the daily patterns. Moreover, some of the forecasts have negative values, which were also observed in the endogenous version of the model for this same scenario. This is an effect caused by the change in the realizations of the data due to street works..

The RMSE scores obtained from the model trained with the subset of exogenous variables are presented in Table 8.53.

In order to compare the RMSE values obtained by both models, a side-by-side table of results is presented in Table 8.54, which shows the scores for each forecasting step. The results show that the LSTM with all the exogenous variables has a higher RMSE in forecasting steps one to four. However, the following forecasting steps present a lower error rate than the model trained with the subset of exogenous variables. This indicates that some of the exogenous information adds noise in the first forecasting steps, but it contributes to reducing error in the later ones.

8.6.3 GRU

The hyperparameter values obtained for the GRU candidate models for the Los Angeles sector are presented in Table 8.55.

Table 8.56 presents the scores for the GRU experiments in the city of Los Angeles. The obtained RMSE scores show that the increase in error is smaller for the later forecasting hours (from hour four onward).

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.099506	0.118512	0.121639	0.131558	0.132344	0.132003
Subset of exogenous variables	0.085215	0.105905	0.116418	0.129116	0.138504	0.139243

Table 8.54: Los Angeles RMSE comparison of LSTM exogenous models trained with all the exogenous variables and with only a subset of them.

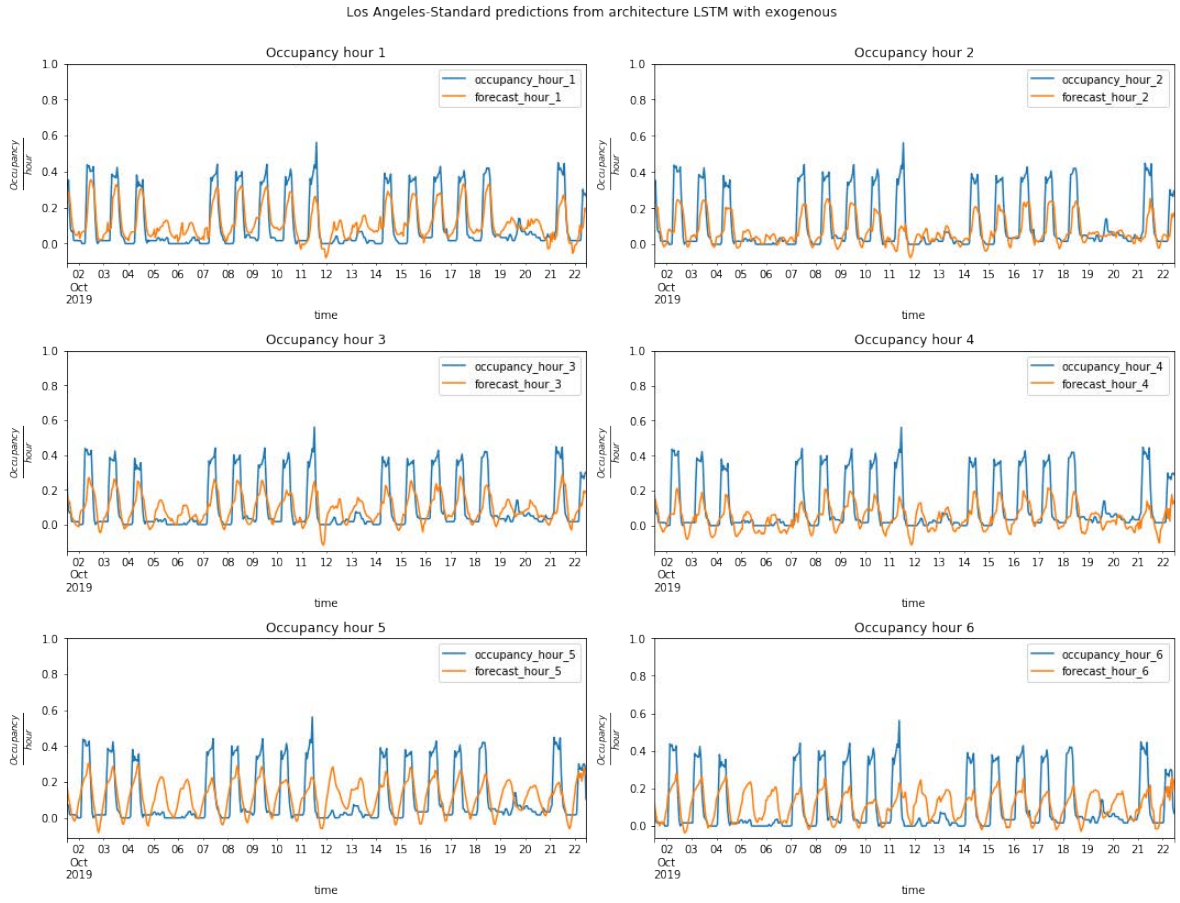


Figure 8.134: Forecast visualizations for the Los Angeles Standard sector LSTM model with exogenous variables. From top left to bottom right are steps one to six.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_exogenous_LA_Standard	8	3	0.125	0.25	Glorot normal	Linear	32	RMSProp	0.000271	1

Table 8.55: Hyperparameter values obtained by random search for the Los Angeles GRU model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_exogenous_LA_Standard	0.103600	0.116677	0.125230	0.129723	0.131936	0.132077	0.124002

Table 8.56: Los Angeles GRU exogenous model's RMSE values obtained in the test set.

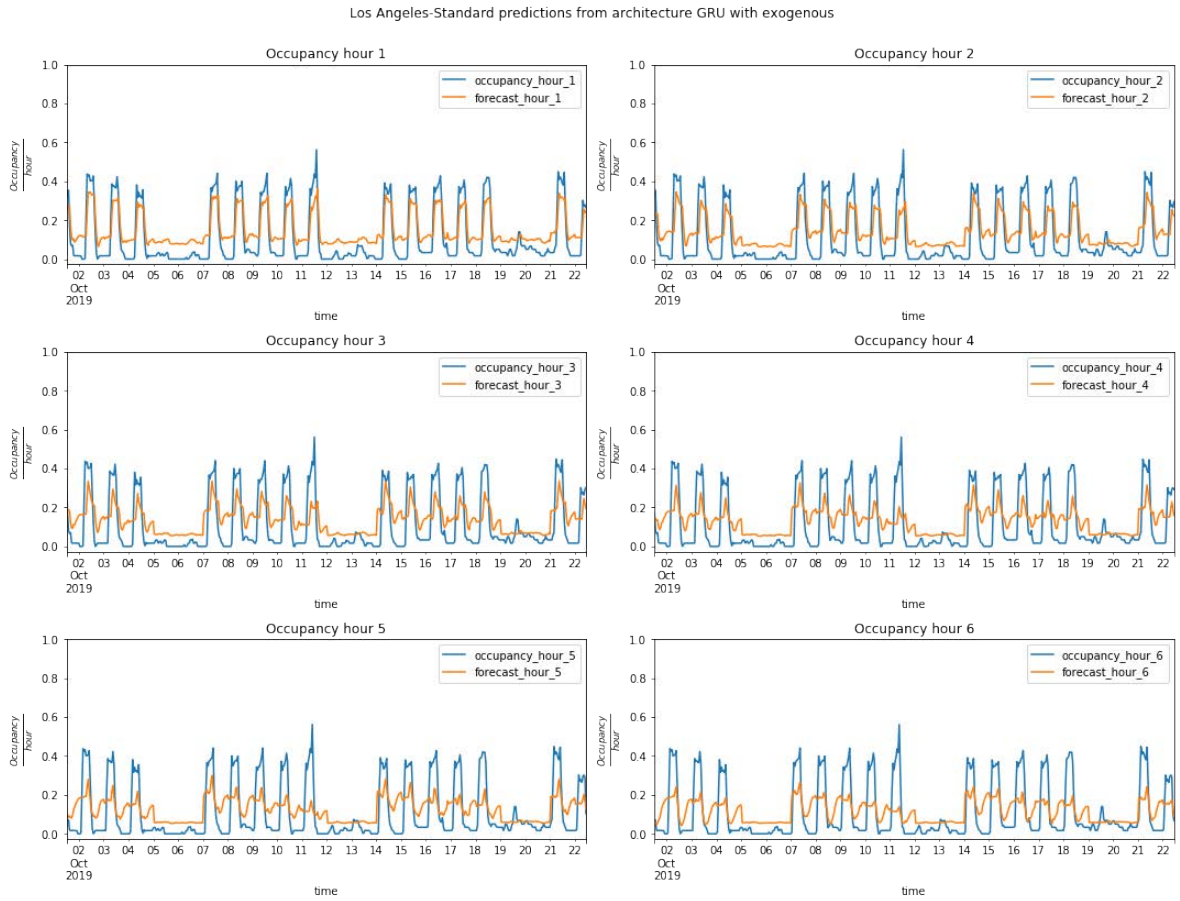


Figure 8.135: Forecast visualizations for the Los Angeles Standard sector GRU model with exogenous variables. From top left to bottom right are steps one to six.

Figure 8.135 shows that the GRU model differentiates daily and weekly patterns better than previous models, although it fails when providing the lower occupancy values of the time series, as they are higher than the true occupancy. In the later forecasting hours, the forecasts still follow the time series pattern, but those values are more conservative with a tendency towards the mean.

The RMSE scores obtained from the GRU model trained with the subset of exogenous variables are presented in Table 8.57 for each of the forecasting hours (from one to six) and their mean.

Table 8.58 shows the RMSE scores obtained for the model trained with all the exogenous

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.114316	0.124537	0.131245	0.135404	0.137455	0.137850	0.130909

Table 8.57: Los Angeles GRU model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of exogenous variables.

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.103600	0.116677	0.125230	0.129723	0.131936	0.132077
Subset of exogenous variables	0.114316	0.124537	0.131245	0.135404	0.137455	0.137850

Table 8.58: Los Angeles RMSE comparison of GRU exogenous models trained with all the exogenous variables and with only a subset of them.

variables and the one trained with a subset of them. The subset of exogenous variables has been selected based on the observations in Section 8.1. In all the forecasting steps, the model that uses all the exogenous information has the lowest error rate.

8.7 Riyadh experiments

Table 8.59 presents the date range for the training, validation and test sets for each of the Riyadh experiments. The sector for training the model with exogenous information is Sector-10, because it presenteda higher error rate than Sector-1, although they have similarities.

8.7.1 MLP

The hyperparameter values obtained by the random search from the MLP experiments are presented in Table 8.60.

Table 8.61 presents the obtained RMSE scores for the Riyadh Sector-10 MLP model trained with exogenous information. The model presents a very low error rate at forecasting step one (hour 1), which increases a little in the successive steps

Figure 8.136 shows the forecasts of the candidate MLP model for Sector-10 in Riyadh. The model shows nearly perfect accuracy throughout the month in terms of the data used in the test set for one to six forecasting steps, although it fails to achieve lower values for some days of the weeks. Specifically, this can be observed for days 4, 11 and 18, which correspond to Fridays and are part of the weekend.

Similarly to the previous scenarios, in order to assess the findings from Section 8.1, the same model has been trained with a subset of the variables. The variables of this subset are selected according to the scenario characteristics and the findings mentioned. For the Riyadh scenario, the chosen exogenous variables are:

Sector	Training	Validation	Test
All sectors	2018-09-01, 2019-08-01	2019-08-02, 2019-09-12	2019-09-13, 2019-10-24

Table 8.59: Riyadh sector intervals of dates used for the training, validation and test sets.

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate
MLP_exogenous_riyadh_sector10	64	2	0	He normal	ReLu	32	RMSprop	0.0013431

Table 8.60: Hyperparameter values obtained by random search for the Riyadh MLP model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_riyadh_sector10	0.054872	0.068552	0.076159	0.080320	0.081744	0.081357	0.07451

Table 8.61: Riyadh MLP exogenous model's RMSE values obtained in the test set.

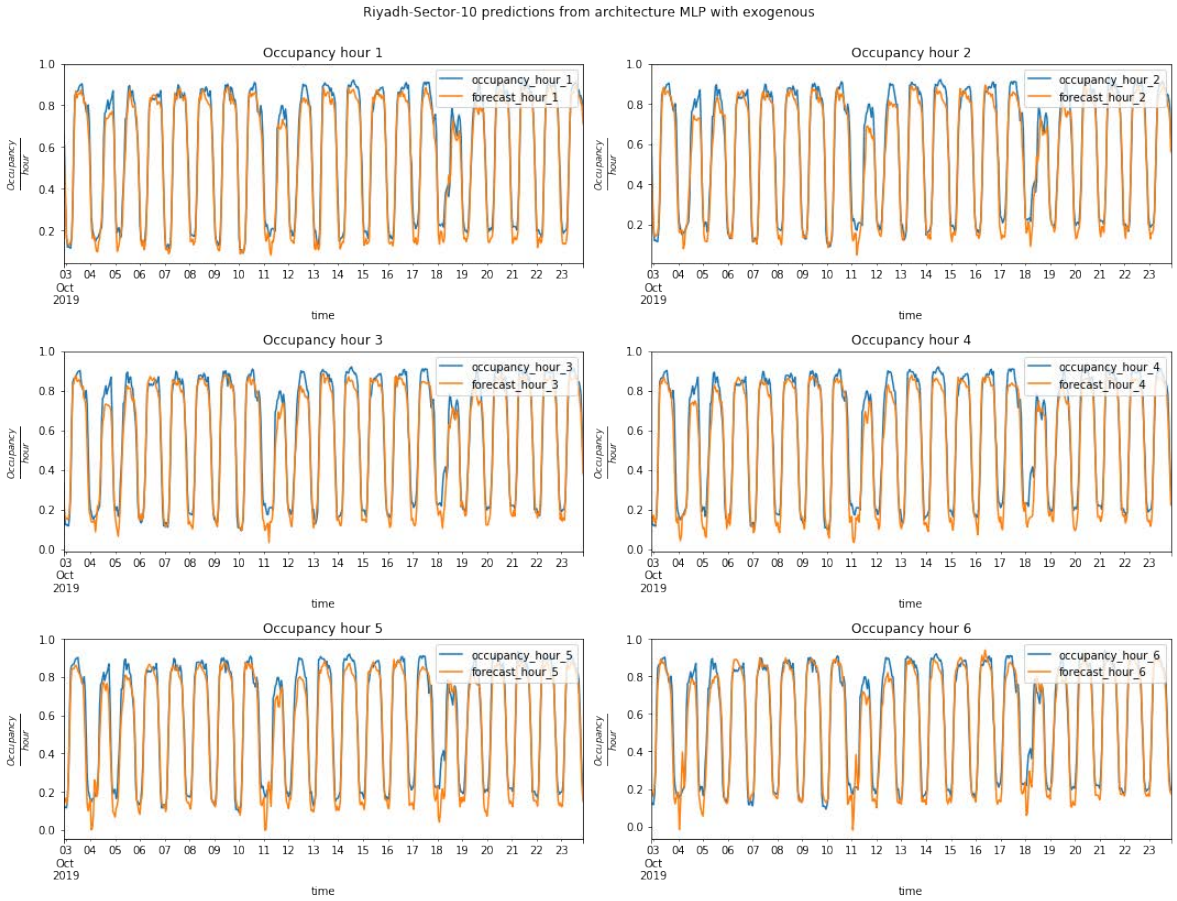


Figure 8.136: Forecast visualizations for the Riyadh Sector-10 MLP model with exogenous variables. From top left to bottom right are steps one to six.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.051951	0.060792	0.066598	0.070552	0.071579	0.068780	0.065469

Table 8.62: Riyadh MLP model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of exogenous variables.

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.054872	0.068552	0.076159	0.080320	0.081744	0.081357
Subset of exogenous variables	0.051951	0.060792	0.066598	0.070552	0.071579	0.068780

Table 8.63: Riyadh RMSE comparison of MLP exogenous models trained with all the exogenous variables and with only a subset of them.

- Summary.
- Precipitation Type.
- Day of the Week.
- Month.
- Holiday.
- Precipitation Intensity.
- Wind Gust.
- Wind Speed

The RMSE scores obtained from the model trained with the subset of exogenous variables are presented in Table 8.62.

Table 8.63 presents a side-by-side comparison of the RMSE scores obtained by the model trained with all the exogenous variables and the model trained with only the subset. The results show that the model trained with the subset of exogenous variables presents lower RMSE values in all the hours, indicating that not all the exogenous variables provide information and that some of them add noise to the model.

8.7.2 LSTM

The hyperparameter values obtained by random search when using exogenous variables for the LSTM model are shown in Table 8.64.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
LSTM_exogenous_riyadh_sector10	512	1	0.125	0.125	Glorot normal	ReLu	32	SGD	0.0453334	0.7

Table 8.64: Hyperparameter values obtained by random search for the Riyadh LSTM model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
LSTM_exogenous_riyadh_sector10	0.049109	0.049818	0.051592	0.052692	0.052850	0.052960	0.051538

Table 8.65: Riyadh LSTM exogenous model's RMSE values obtained in the test set.

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.040634	0.045190	0.047528	0.048567	0.049684	0.051854	0.047379

Table 8.66: Riyadh LSTM model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of exogenous variables.

Table 8.65 presents the obtained RMSE scores for the different forecasting steps of the LSTM model that was trained by incorporating the exogenous variables for Sector-10 in the Riyadh scenario. The model presents a low error rate that increases slightly at each successive forecasting step.

The visualizations of the Riyadh Sector-10 forecasts at horizon steps one to six are presented in Figure 8.137. The visualizations show that the model outputs accurate forecasts for all the forecasting steps, something that was not accomplished in the endogenous version of the model.

Table 8.66 shows the RMSE values at each time step, which were obtained by an LSTM model trained with a subset of exogenous variables. The error is low and does not increase a lot between successive time steps.

A side-by-side comparison of the obtained RMSE scores is presented in Table 8.67, which contrasts the model trained with all the exogenous variables and the model trained with a subset of them. The table shows that the model trained with a subset of the variables achieves a lower error rate in all the time steps than does the model trained with all the exogenous variables, which is similar to what has been observed in the MLP model. This indicates that the subset of variables considered are the ones that improve the model and that the others add noise to the training process.

8.7.3 GRU

The hyperparameter values obtained by the random search for the GRU model of sector Sector-10 in Riyadh when using exogenous variables are presented in Table 8.68.

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.049109	0.049818	0.051592	0.052692	0.052850	0.052960
Subset of exogenous variables	0.040634	0.045190	0.047528	0.048567	0.049684	0.051854

Table 8.67: Riyadh RMSE comparison of LSTM exogenous models trained with all the exogenous variables and with only a subset of them.

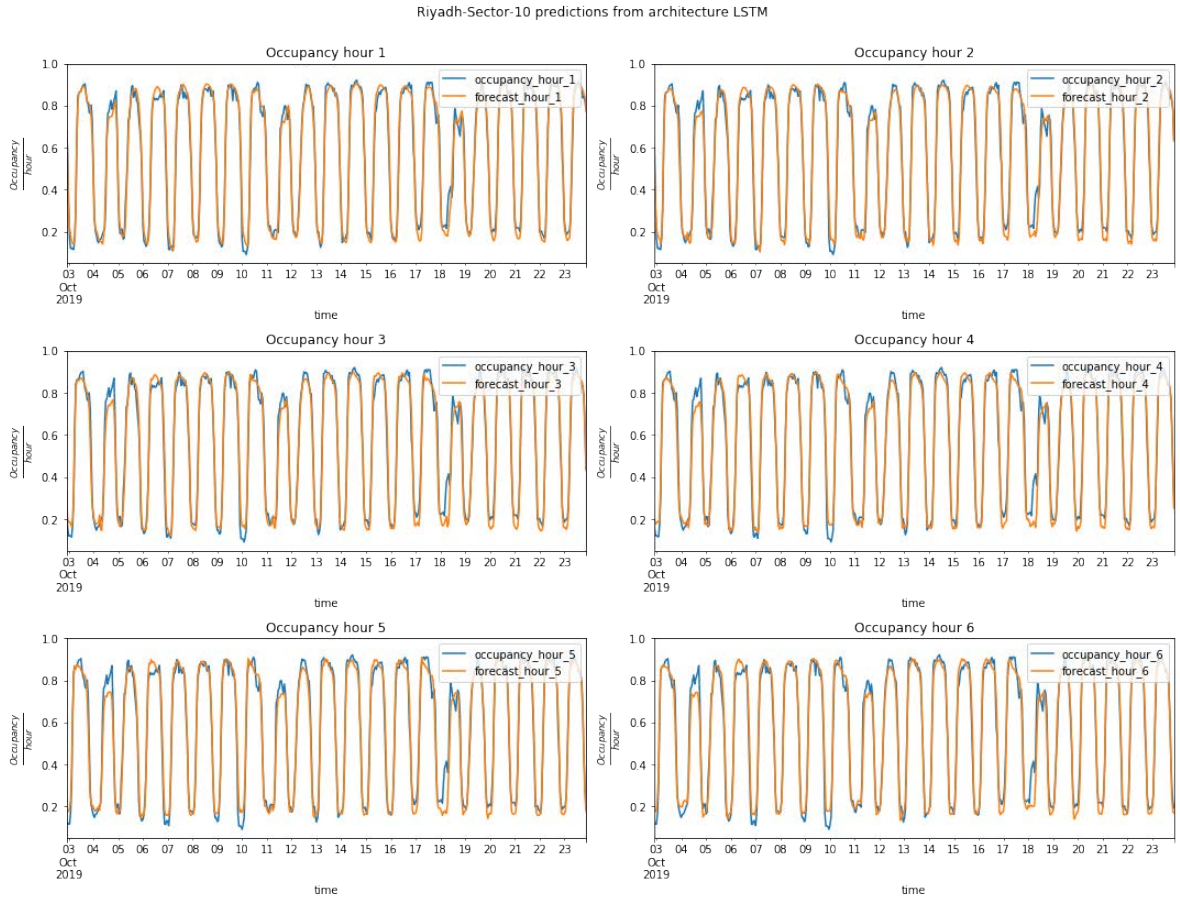


Figure 8.137: Forecast visualizations for the Riyadh Sector-10 LSTM model with exogenous variables. From top left to bottom right are steps one to six.

Experiment name	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate	Clipping norm value
GRU_exogenous_riyadh_sector10	64	3	0.125	0	He normal	Tanh	256	RMSProp	0.00128415	0.4

Table 8.68: Hyperparameter values obtained by random search for the Riyadh GRU model trained with exogenous variables.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
GRU_exogenous_riyadh_sector10	0.053634	0.057642	0.056152	0.059179	0.060058	0.061906	0.058238

Table 8.69: Riyadh GRU exogenous model's RMSE values obtained in the test set.

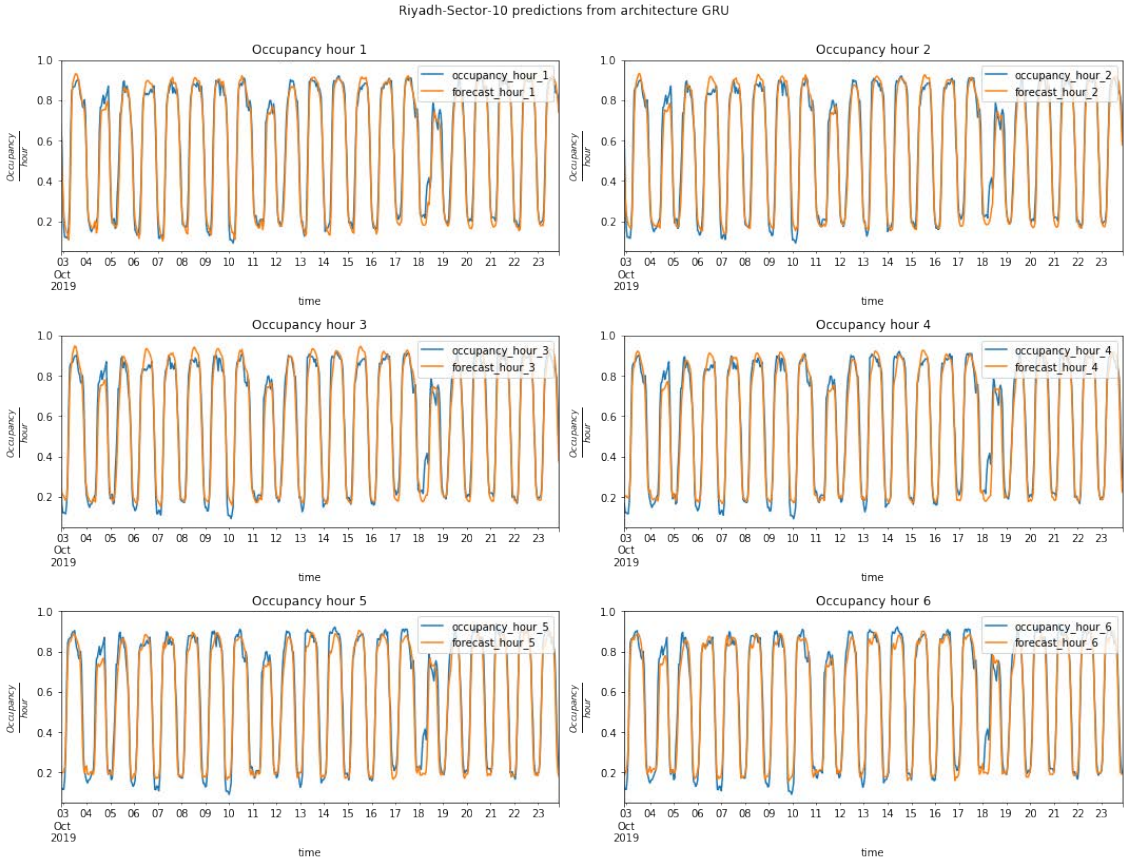


Figure 8.138: Forecast visualizations for the Riyadh Sector-10 GRU model with exogenous variables. From top left to bottom right are steps one to six.

Table 8.69 presents the obtained error scores for the GRU model trained with exogenous variables for Sector-10 in Riyadh. They present similar RMSE values across all the forecasting hours, thus improving the forecasts of later hours when compared to the endogenous model of the sector.

The visualizations of the forecasts at step horizons one to six are presented in Figure 8.138 for Sector-10 in Riyadh. The visualizations show that the model forecasts are very similar in all the forecasting steps. The model shows consistent predictions for the weekends, something that the endogenous model failed to do.

Table 8.70 shows the scores obtained by the GRU model trained with a subset of exogenous variables.

Similarly to previous methods, a side-by-side comparison is provided in Table 8.71, contrasting the model trained with all the exogenous variables and the model trained

RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
0.049293	0.053471	0.056453	0.056714	0.057627	0.061467	0.056044

Table 8.70: Riyadh GRU model's RMSE values when testing the model forecasts in the test set. The model has been trained with a subset of exogenous variables.

Experiment	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6
All exogenous variables	0.053634	0.057642	0.056152	0.059179	0.060058	0.061906
Subset of exogenous variables	0.049293	0.053471	0.056453	0.056714	0.057627	0.061467

Table 8.71: Riyadh RMSE comparison of GRU exogenous models trained with all the exogenous variables and with only a subset of them.

with a subset of them. Similarly to the MLP and LSTM cases, the model trained with a subset of the exogenous variables presents lower RMSE values, although the difference is very small in the later time steps.

8.8 Results and conclusions

This section presents a comparative study of the results obtained for each scenario from the experiments with the exogenous variables. Next, the findings of the chapter are discussed and commented.

8.8.1 Result comparison

Table 8.72 shows the RMSE scores of the model trained with all the exogenous variables from the experiments in the Antwerp scenario. All models have similar RMSE values at forecasting step one. However, the LSTM model is the one that presents a smaller increase in error rate for the successive time steps.

Figure 8.139 more clearly presents the evolution of the RMSE for the trained models at each time step. As stated before, the LSTM model obtains the lowest RMSE for all the time steps, followed by GRU. MLP presents a similar score as GRU, although it has a higher increase at each time step.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_antwerp_pelikaanstraat	0.116387	0.135267	0.147768	0.157503	0.164878	0.173999	0.150557
LSTM_exogenous_antwerp_pelikaanstraat	0.112092	0.128344	0.135903	0.141196	0.145912	0.150256	0.136307
GRU_exogenous_antwerp_pelikaanstraat	0.114243	0.140703	0.143994	0.150458	0.158148	0.164877	0.14648

Table 8.72: RMSE scores obtained from the models trained with exogenous variables for the experiments in the Antwerp scenario.

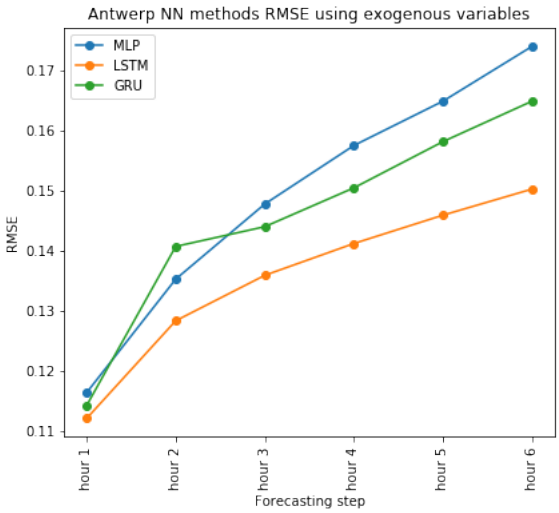


Figure 8.139: Visualizations of the RMSE scores obtained from the models trained with exogenous variables for the experiments in the Antwerp scenario.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_wattens_outside	0.086194	0.103884	0.115519	0.123145	0.126526	0.130674	0.11554
LSTM_exogenous_wattens_outside	0.099849	0.115229	0.125736	0.132120	0.135185	0.139255	0.125309
GRU_exogenous_wattens_outside	0.099434	0.110648	0.117117	0.121229	0.123851	0.125638	0.116672

Table 8.73: RMSE scores obtained from the models trained with exogenous variables for the experiments in the Wattens scenario.

Table 8.73 shows the RMSE scores of the model trained with all the exogenous variables from the experiments in the Wattens scenario. The error from MLP is the lowest one at forecasting step one. However, its error rate increases faster than the error from GRU. This fact ensures that both models achieve similar mean RMSE.

In Figure 8.140, the dynamics can be seen more clearly through the different forecasting steps. As mentioned previously, MLP has the lowest error rate in the first forecasting steps, but GRU scores better in forecasting steps four, five and six. LSTM is the one witha higher error rate, although its error rate is similar to that of GRU at forecasting step one.

Table 8.74 shows the RMSE scores of the model trained with all the exogenous variables from the Los Angeles scenario. At first sight, LSTM and GRU have similar values at all the forecasting steps, while MLP shows worse errors than those.

In Figure 8.141, the dynamics can be seen more clearly through the different forecasting steps. GRU and LSTM show very similar behaviour across all the time steps, while MLP presents higher error rates but with lower increases between the successive steps.

Table 8.75 shows the RMSE scores of the model trained with all the exogenous variables from the experiments in the Riyadh scenario. At first sight, RNN models present the lowest of the errors.

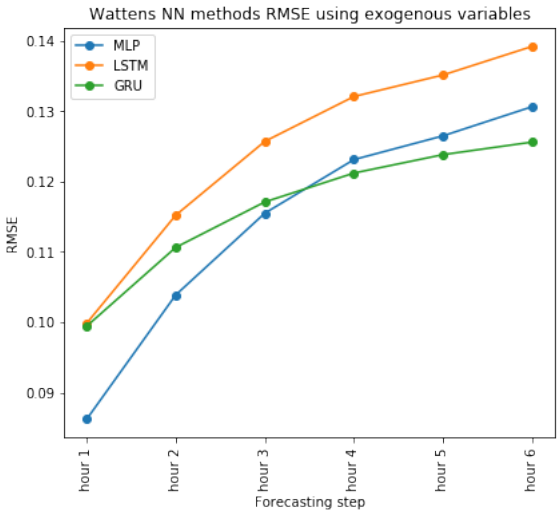


Figure 8.140: Visualizations of the RMSE scores obtained from the models trained with exogenous variables for the experiments in the Wattens scenario.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_LA_standard	0.133507	0.132600	0.135605	0.134806	0.138004	0.137220	0.135734
LSTM_exogenous_LA_standard	0.099506	0.118512	0.121639	0.131558	0.132344	0.132003	0.123919
GRU_exogenous_LA_standard	0.103600	0.116677	0.125230	0.129723	0.131936	0.132077	0.124002

Table 8.74: RMSE scores obtained from the models trained with exogenous variables for the experiments in the Los Angeles scenario.

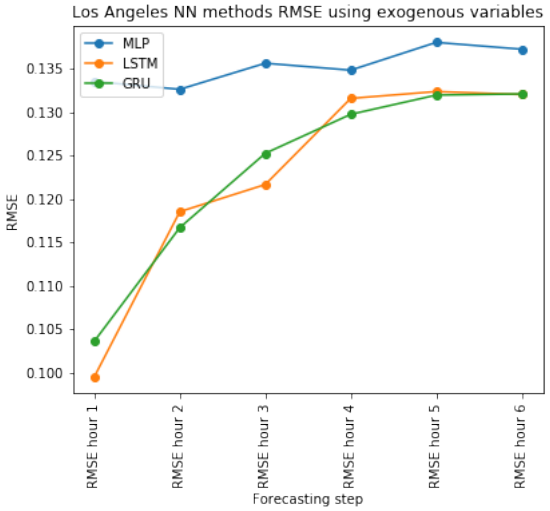


Figure 8.141: Visualizations of the RMSE scores obtained from the models trained with exogenous variables for the experiments in the Los Angeles scenario.

Experiment name	RMSE hour 1	RMSE hour 2	RMSE hour 3	RMSE hour 4	RMSE hour 5	RMSE hour 6	Mean RMSE
MLP_exogenous_riyadh_sector10	0.054872	0.068552	0.076159	0.080320	0.081744	0.081357	0.07451
LSTM_exogenous_riyadh_sector10	0.049109	0.049818	0.051592	0.052692	0.052850	0.052960	0.051538
GRU_exogenous_riyadh_sector10	0.053634	0.057642	0.056152	0.059179	0.060058	0.061906	0.058238

Table 8.75: RMSE scores obtained from the models trained with exogenous variables for the experiments in the Riyadh scenario.

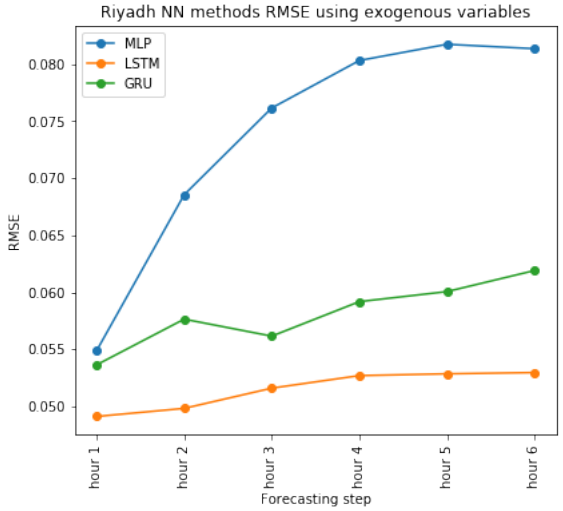


Figure 8.142: Visualizations of the RMSE scores obtained from the models trained with exogenous variables for the experiments in the Riyadh scenario.

The visualizations in Figure 8.142 show that LSTM maintains a very low error rate during the successive forecasting steps, something that can be seen also in GRU, although its error rate increases faster than the former. MLP presents a very high error increase in forecasting steps one to three, followed by slower error rate increases.

In three of the four scenarios, LSTM has shown to be the best method when exogenous variables are included in the process of modeling parking occupancy time series by means of NN models. Specifically, it presents a smaller increase in error rate at the different steps of the forecasting horizon and the lowest error rate in all the time steps.

8.8.2 Conclusions

This chapter has presented a set of exogenous variables from different sources, namely calendar and weather. These are intuitively perceived as possible sources of information that can improve the modeling of parking occupancy time series.

We have studied the effects of the values of those variables, specifically on parking systems, by analysing the embeddings of the categorical variables. This analysis of the embeddings involved using dimensional reduction techniques like PCA and/or tSNE.

For the numerical variables, we have studied the relationship between their values and the parking occupancy. This was done by training NN models with each of the numerical exogenous variables in order to create a model and predict the parking occupancy. This has shown how the model forecasts changed when conditioned by the selected exogenous variable values.

Different architectures have been proposed for incorporating the exogenous information into the MLP and RNN methods, and we have shown the RMSE scores obtained by using all the variables and a subset of the exogenous variables for the scenarios. The subsets of the exogenous variables were selected from the findings obtained when performing the study of the exogenous variables. In some cases, the model trained with the subset of the exogenous variables has shown equal or lower error rates when compared to the model trained with all the exogenous information. This indicates that not all the exogenous variables contribute to the forecasts and that some of them can also add noise to the training process. It would be interesting for further research to develop a NN methods to detect which variables contribute and which ones add noise.

In summary, the proposals and findings of this chapter are:

- An analysis of the categorical and numerical exogenous variables has been proposed.
- Different architectures for multivariate time series with exogenous variables (categorical) have been proposed.
- Categorical information that influences parking system behaviour has been identified.
- Interpretability has been established for categorical information that conditions parking systems.
- Scaled normalization has been shown to work better than standardized normalization.
- Possible architectures have been proposed for NN methods to handle multivariate time series with exogenous variables.
- LSTM models have proven to be superior at forecasting parking occupancy when adding exogenous information to the problem.
- A subset of exogenous data has all the necessary information for improving the models, while using all the exogenous variables can add noise to the training process. This subset depends on the scenario. Therefore, it could be useful to research NN methods that can discern the contribution of the variables.

9 Conclusions

In this final chapter, we turn back to the main achievements of this dissertation and present a summary of the primary results obtained. Following this, we discuss the main research findings and, finally, formulate some recommendation for further research.

9.1 Results Discussion

In this work, different methods have been studied in order to characterize different heterogeneous parking systems (namely parking sectors). Specifically, we use sensor data to capture the occupancy and vacancy of a parking space. Moreover, this characterization has been used to produce models that predict hourly parking occupancy from one to six hours following the current timestamp. The data and its realizations are described in Chapter 2.

Statistical methods and visualizations have been used to perform the characterization of the systems and provide a guideline for detecting the relevant aspects that can be learnt from the data. Insights about this characterization can be found in Chapter 3.

For the modeling aspect, two families of methods have been proposed: ARIMA and NN. For ARIMA, we used the SARIMA method while the NN methods focus on MLP, LSTM and GRU. The results of and discussions on each method are provided in Chapters 6 and 7.

In order to improve the models, a study of exogenous data has been conducted by inspecting its behaviour in relation to parking occupancy when using NN methods. In this way, we provide a methodology for studying and incorporating such exogenous information into a time series context for NN. Using this information, we formulated new models for comparison against models without this kind of data. Detailed descriptions and results are presented in Chapter 8, which characterizes parking systems according to the effect of exogenous influences.

Next, a comparison of the obtained results is provided for the models that used the endogenous variable (ARIMA, MLP, LSTM and GRU). For a subset of models from the NN family, we further compare the endogenous models with those that also use exogenous information.

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
ARIMA_antwerp_pelikaanstraat	0.1033	0.1360	0.1562	0.1737	0.1881	0.1994	0.1594
MLP_endogenous_antwerp_pelikaanstraat	0.110130	0.133542	0.150384	0.162346	0.172676	0.179238	0.153562
LSTM_endogenous_antwerp_pelikaanstraat	0.110668	0.137267	0.153259	0.165521	0.174594	0.181250	0.155625
GRU_endogenous_antwerp_pelikaanstraat	0.113561	0.135125	0.151318	0.165038	0.173547	0.182441	0.157462
ARIMA_antwerp_vanwesenbekestraat	0.0778	0.0997	0.1073	0.1110	0.1130	0.1146	0.1039
MLP_endogenous_antwerp_vanwesenbekestraat	0.095566	0.116724	0.125885	0.131969	0.133297	0.133658	0.124019
LSTM_endogenous_antwerp_vanwesenbekestraat	0.103780	0.126287	0.137642	0.141567	0.142330	0.144293	0.133610
GRU_endogenous_antwerp_vanwesenbekestraat	0.103207	0.124504	0.131194	0.137746	0.142796	0.132778	0.129447
ARIMA_antwerp_vanwesenbekestraat2	0.0824	0.1039	0.1130	0.1190	0.1221	0.1233	0.1106
MLP_endogenous_antwerp_vanwesenbekestraat2	0.101069	0.124634	0.135584	0.143275	0.146959	0.148462	0.134506
LSTM_endogenous_antwerp_vanwesenbekestraat2	0.093660	0.121632	0.137392	0.148286	0.154991	0.159997	0.137875
GRU_endogenous_antwerp_vanwesenbekestraat2	0.099607	0.120591	0.128654	0.138555	0.139059	0.141712	0.129501
ARIMA_antwerp_vanschoonhovestraat1	0.1042	0.1259	0.1367	0.1423	0.1459	0.1489	0.134
MLP_endogenous_antwerp_vanSchoonhovestraat1	0.123471	0.139036	0.148395	0.154933	0.158351	0.161390	0.149402
LSTM_endogenous_antwerp_vanSchoonhovestraat1	0.113743	0.132188	0.144037	0.145389	0.153726	0.155251	0.141533
GRU_endogenous_antwerp_vanschoonhovestraat1	0.119425	0.134792	0.141594	0.144979	0.146056	0.148689	0.139435
ARIMA_antwerp_vanschoonhovestraat2	0.1310	0.1727	0.1975	0.2109	0.2207	0.2271	0.1933
MLP_endogenous_antwerp_vanSchoonhovestraat2	0.157954	0.204540	0.235772	0.255512	0.268585	0.280565	0.237706
LSTM_endogenous_antwerp_vanSchoonhovestraat2	0.177184	0.214003	0.231087	0.237052	0.254416	0.258750	0.233800
GRU_endogenous_antwerp_vanschoonhovestraat2	0.163100	0.201986	0.229196	0.245610	0.258182	0.265528	0.229798
ARIMA_antwerp_vanschoonhovestraat3	0.1307	0.1540	0.1642	0.1710	0.1731	0.1738	0.1611
MLP_endogenous_antwerp_vanSchoonhovestraat3	0.13	0.156846	0.166445	0.173264	0.176913	0.177004	0.164277
LSTM_endogenous_antwerp_vanSchoonhovestraat3	0.138450	0.157465	0.166710	0.171605	0.175181	0.175716	0.164694
GRU_endogenous_antwerp_vanschoonhovestraat3	0.128000	0.15256	0.162955	0.167865	0.169857	0.171224	0.15943

Table 9.1: Candidate model RMSE scores in test set for the Antwerp scenario.

9.1.1 Endogenous results

Table 9.1 presents the obtained RMSE scores for each of the experiments performed in the Antwerp scenario. In checking the mean RMSE values obtained in each experiment, it is observable that, overall, there are no huge discrepancies between the models; although in some cases ARIMA outperformed NN methods by some margins.

By inspecting the visualizations in Figure 9.1, we can more easily identify the various models' scores in the Antwerp sectors. For each of the sectors, the following can be concluded:

- For Pelikaanstraat, NN methods performed very similarly while ARIMA provides slightly better first-step forecasts, but the error increases more quickly at each successive step than with the other methods.
- ARIMA models from the Van Wesenbekestraat sectors perform better than any of the NN methods considered. This indicates that few parameters are enough for modeling those sectors, because both ARIMA models have one AR parameter in the regular part and one MA parameter in the seasonal part.
- In the Van Schoonhovestraat sectors, ARIMA performed slightly better for Zone 1 in the first steps, but GRU has slightly less error in the last forecasting step. Although the characteristics of Zone 2 make it a random walk, ARIMA performs slightly better than the others. On the other hand, GRU performs best for Zone 3 at all the time steps considered, while ARIMA is the second best.

RMSE scores for Barcelona experiments are presented in Table 9.2. At first glance, it seems that ARIMA is the method with the lowest error.

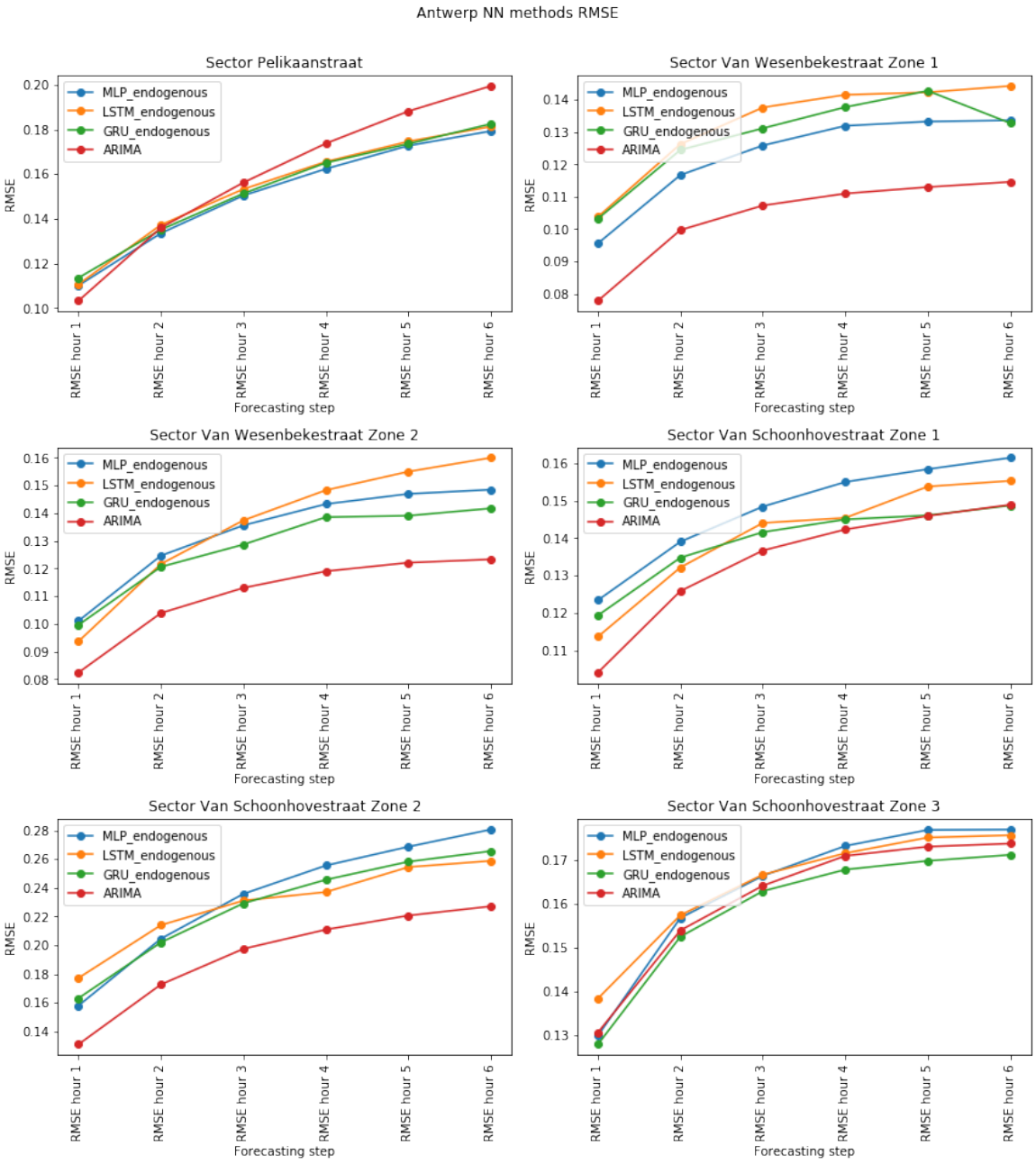


Figure 9.1: Antwerp sector endogenous modelRMSE scores for each hour. MLP in blue, LSTM in orange, GRU in green and ARIMA in red.

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
ARIMA_barcelona_regular	0.0656	0.0929	0.1079	0.1157	0.1218	0.1277	0.1053
MLP_endogenous_barcelona_regular	0.092727	0.109607	0.119814	0.125462	0.130421	0.135289	0.119841
LSTM_endogenous_barcelona_regular	0.100250	0.113873	0.128788	0.146382	0.164747	0.175187	0.141719
GRU_endogenous_barcelona_regular	0.096280	0.114129	0.130481	0.142458	0.148959	0.158079	0.133832

Table 9.2: Candidate model RMSE scores in test set for the Barcelona scenario.

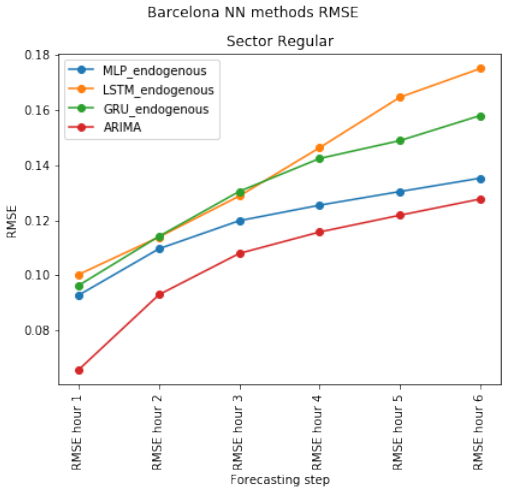


Figure 9.2: Barcelona sector endogenous modelRMSE scores for each hour. MLP in blue, LSTM in orange, GRU in green and ARIMA in red.

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
ARIMA_wattens_inside	0.0501	0.0825	0.1060	0.1244	0.1368	0.1465	0.1077
MLP_endogenous_wattens_inside	0.056937	0.081595	0.101219	0.116856	0.127939	0.138484	0.107350
LSTM_endogenous_wattens_inside	0.063618	0.098741	0.126547	0.149175	0.162757	0.176160	0.134990
GRU_endogenous_wattens_inside	0.056622	0.079935	0.100578	0.116191	0.127633	0.139902	0.107230
ARIMA_wattens_outside	0.0587	0.0989	0.1248	0.1436	0.1561	0.1665	0.1248
MLP_endogenous_wattens_outside	0.167275	0.168808	0.160449	0.161442	0.173651	0.192762	0.173874
LSTM_endogenous_wattens_outside	0.095564	0.132638	0.162885	0.186857	0.191105	0.220024	0.169323
GRU_endogenous_wattens_outside	0.083591	0.116279	0.139536	0.156237	0.165543	0.174135	0.142176

Table 9.3: Candidate model RMSE scores in test set for the Wattens scenario.

The visualizations in Figure 9.2 allow observing that the ARIMA model provided the lowest error forecasts at each of the time steps considered, but the MLP model error increases more slowly than that of ARIMA at each forecasting step, resulting in both methods having similar errors at the last time step. Both RNN methods started by performing similarly, but GRU has lower error than LSTM at higher time steps.

The results for the experiments in Wattens are provided in Table 9.3 for each of the models obtained. Although in the Inside sector all models performed similarly, ARIMA is clearly the method with less error in the Outside sector.

Inspecting the visualizations in Figure 9.3, one could conclude that:

- In the Inside sector, all models have similar scores at the first forecasting step (ARIMA the lowest), but MLP and GRU present the lowest RMSE values as the time step increases.
- In the Outside sector, ARIMA performed better than the other methods, but its error increases more quickly at each forecasting step than the error from GRU, resulting in similar scores in the sixth forecasting step.

Table 9.4 presents the RMSE scores of the endogenous models from the Los Angeles

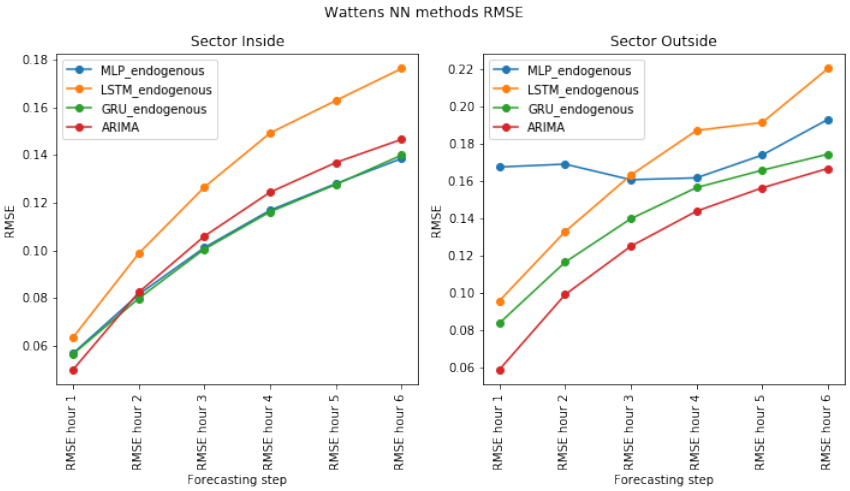


Figure 9.3: Wattens sector endogenous modelRMSE scores for each hour. MLP in blue, LSTM in orange, GRU in green and ARIMA in red.

experiments. After checking the mean RMSE, the models seem to perform similarly (with some exceptions) and it is not clear which one scores better.

The visualizations shown in Figure 9.4 allow close inspection of the RMSE scores obtained, by which we can say:

- For the Standard sector, ARIMA outperforms the other models at the three first forecasting steps; but, from there, MLP starts to have a similar score. GRU performs similarly to MLP, but slightly worse.
- In the Permit sector, ARIMA has a very low error for the first forecasting step, but the error increases more quickly than the other methods for the successive time steps. Anecdotally, LSTM achieved the lowest error at the last forecasting step,

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
ARIMA_LA_standard	0.0396	0.0672	0.0849	0.0971	0.1064	0.1140	0.0849
MLP_endogenous_LA_Standard	0.083213	0.091379	0.096799	0.102012	0.105445	0.112373	0.099285
LSTM_endogenous_LA_Standard	0.157765	0.170322	0.187332	0.202759	0.221877	0.232236	0.198609
GRU_endogenous_LA_Standard	0.07801	0.09512	0.10619	0.11393	0.12064	0.12356	0.107401
ARIMA_LA_permit	0.0179	0.0259	0.0294	0.0322	0.0334	0.0344	0.0289
MLP_endogenous_LA_Permit	0.026132	0.027593	0.028446	0.029413	0.030845	0.032627	0.029377
LSTM_endogenous_LA_Permit	0.023564	0.026408	0.028256	0.030966	0.0344788	0.029721	0.030148
GRU_endogenous_LA_Permit	0.032748	0.029283	0.031485	0.033495	0.034186	0.037622	0.032948
ARIMA_LA_disabled	0.0387	0.0594	0.0734	0.0827	0.0893	0.0929	0.0727
MLP_endogenous_LA_Disabled	0.068539	0.070388	0.073355	0.076586	0.078879	0.079243	0.074189
LSTM_endogenous_LA_Disabled	0.055212	0.079277	0.087534	0.119898	0.143644	0.192689	0.126183
GRU_endogenous_LA_Disabled	0.0476	0.05673	0.06614	0.06944	0.07601	0.08876	0.06872
ARIMA_LA_zipcar	0.0643	0.0913	0.1056	0.1139	0.1212	0.1261	0.1037
MLP_endogenous_LA_ZipCar	0.065272	0.088367	0.099961	0.106587	0.111648	0.115079	0.099335
LSTM_endogenous_LA_ZipCar	0.065059	0.088938	0.101202	0.108231	0.1166095	0.119523	0.101638
GRU_endogenous_LA_ZipCar	0.07025	0.09021	0.10057	0.10946	0.11207	0.11485	0.10076

Table 9.4: Candidate model RMSE scores in test set for the Los Angeles scenario.

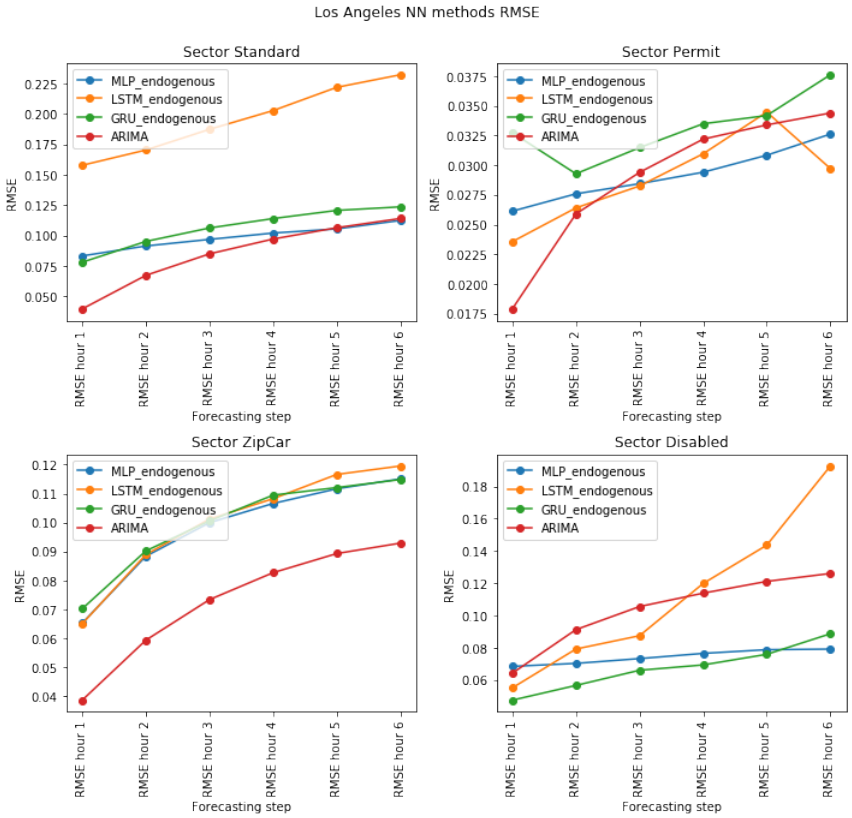


Figure 9.4: Los Angeles sector endogenous modelRMSE scores for each hour. MLP in blue, LSTM in orange, GRU in green and ARIMA in red.

due to reducing its error from step five by a great margin.

- In the ZipCar sector, which is considered a random walk, all models present similar error curves in the forecasting steps, with ARIMA achieving the lowest error in all the steps.
- In the Disabled sector, GRU outscores the other methods in the first five forecasting steps, but MLP scores slightly lower at the sixth step.

Table 9.5 presents the RMSE scores for the endogenous experiments in Riyadh. Contrary to what has been observed in previous scenarios, NN methods outperform ARIMA in both of the sectors considered.

The visualizations shown in Figure 9.5 present a more detailed overview of the RMSE scores obtained by the models from the Riyadh experiments. Observe that:

- In both scenarios, the ARIMA models have a similar score as the other methods, but their errors increase more quickly than the other methods at successive time steps. Another interesting point to note is that NN methods seem to have a more linear increase in error when compared to ARIMA.
- For the Sector-1, NN methods perform similarly at all time steps.

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
ARIMA_riyadh_sector1	0.05216	0.0955	0.1289	0.1564	0.1793	0.1999	0.1354
MLP_endogenous_riyadh_sector1	0.047095	0.060557	0.073404	0.083346	0.093408	0.103209	0.079634
LSTM_endogenous_riyadh_sector1	0.051068	0.069476	0.080824	0.085531	0.093426	0.111499	0.084376
GRU_endogenous_riyadh_sector1	0.059842	0.076781	0.086742	0.092680	0.096025	0.106015	0.087838
ARIMA_riyadh_sector10	0.05	0.0851	0.1117	0.1318	0.1458	0.1553	0.1133
MLP_endogenous_riyadh_sector10	0.071116	0.081256	0.080439	0.086139	0.097126	0.096646	0.087643
LSTM_endogenous_riyadh_sector10	0.078077	0.090971	0.098520	0.108535	0.118905	0.128008	0.104397
GRU_endogenous_riyadh_sector10	0.063591	0.072662	0.081042	0.087759	0.091248	0.097092	0.083619

Table 9.5: Candidate model RMSE scores in test set for the Riyadh scenario.

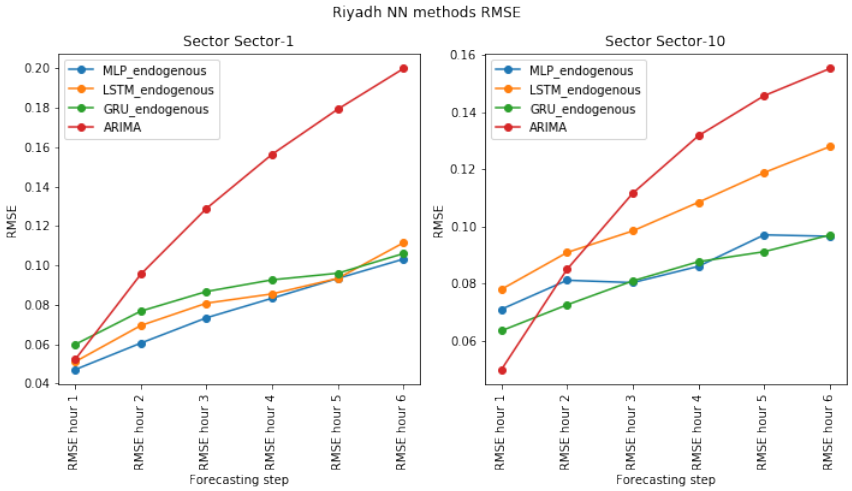


Figure 9.5: Riyadh sector endogenous modelRMSE scores for each hour. MLP in blue, LSTM in orange, GRU in green and ARIMA in red.

- In Sector-10, GRU and MLP perform similarly, while LSTM scores worse than them but better than ARIMA after the second forecasting step.

In the comparison of ARIMA and NN methods using endogenous information, the following has been observed:

- In nearly all cases, ARIMA performed best at the first forecasting step, but it is not always the method with the lowest error at the last forecasting step. This indicates that the ARIMA error for successive predictions increases more quickly than the NN methods, thus making the mean RMSE similar across all methods, although the RMSE scores at each hour are very different. This is a consequence of training the NN models to provide all the forecasting steps at once instead of using a multi-step strategy.
- There is a relationship between model performance and sector characteristics. ARIMA performed better than NN methods in small sectors (Van Schoonhovestraat Zone 2 and ZipCar), but NN methods performed better in sectors with a huge amount of sensors (Riyadh sectors). Subtle dynamics are probably best captured by models with a high number of parameters.

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
MLP_endogenous_antwerp_pelikaanstraat	0.110130	0.133542	0.150384	0.162346	0.172676	0.179238	0.153562
MLP_exogenous_antwerp_pelikaanstraat	0.116387	0.135267	0.147768	0.157503	0.164878	0.173999	0.150557
LSTM_endogenous_antwerp_pelikaanstraat	0.110668	0.137267	0.153259	0.165521	0.174594	0.181250	0.155625
LSTM_exogenous_antwerp_pelikaanstraat	0.112092	0.128344	0.135903	0.141196	0.145912	0.150256	0.136307
GRU_endogenous_antwerp_pelikaanstraat	0.113561	0.135125	0.151318	0.165038	0.173547	0.182441	0.157462
GRU_exogenous_antwerp_pelikaanstraat	0.114243	0.140703	0.143994	0.150458	0.158148	0.164877	0.14648

Table 9.6: Comparison between exogenous and endogenous candidate model RMSE scores in test set for the Antwerp scenario.

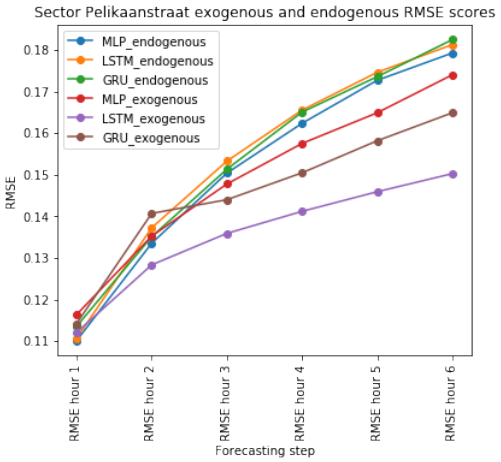


Figure 9.6: RMSE errors from the endogenous and exogenous models for the Antwerp scenario.

9.1.2 Exogenous results

About the exogenous experiments, Table 9.6 compares the NN methods using only the endogenous variables with the models using the exogenous variables in the Antwerp Pelikaanstraat scenario. The obtained results show that the exogenous variables have the effect of improving the forecasts of the latter steps of the forecasting horizon, specially for LSTM and GRU.

Figure 9.6 shows the obtained RMSE scores from the endogenous and exogenous models for the scenario of Antwerp. The visualization shows more clearly that RNN methods are the ones that improve more when using exogenous information, specifically LSTM shows a great improvement when comparing to its endogenous conterpart.

Figure 9.7 shows the forecasts of the LSTM models for the test set of the sector Pelikaanstraat, as it is the method in which the lower RMSE has been obtained. The forecasts of the exogenous model are closer to the real parking occupancy values than the forecasts of the endogenous model. One can observe that on latter forecasting steps (forecasting step three onward), the endogenous model forecasts are more close to the

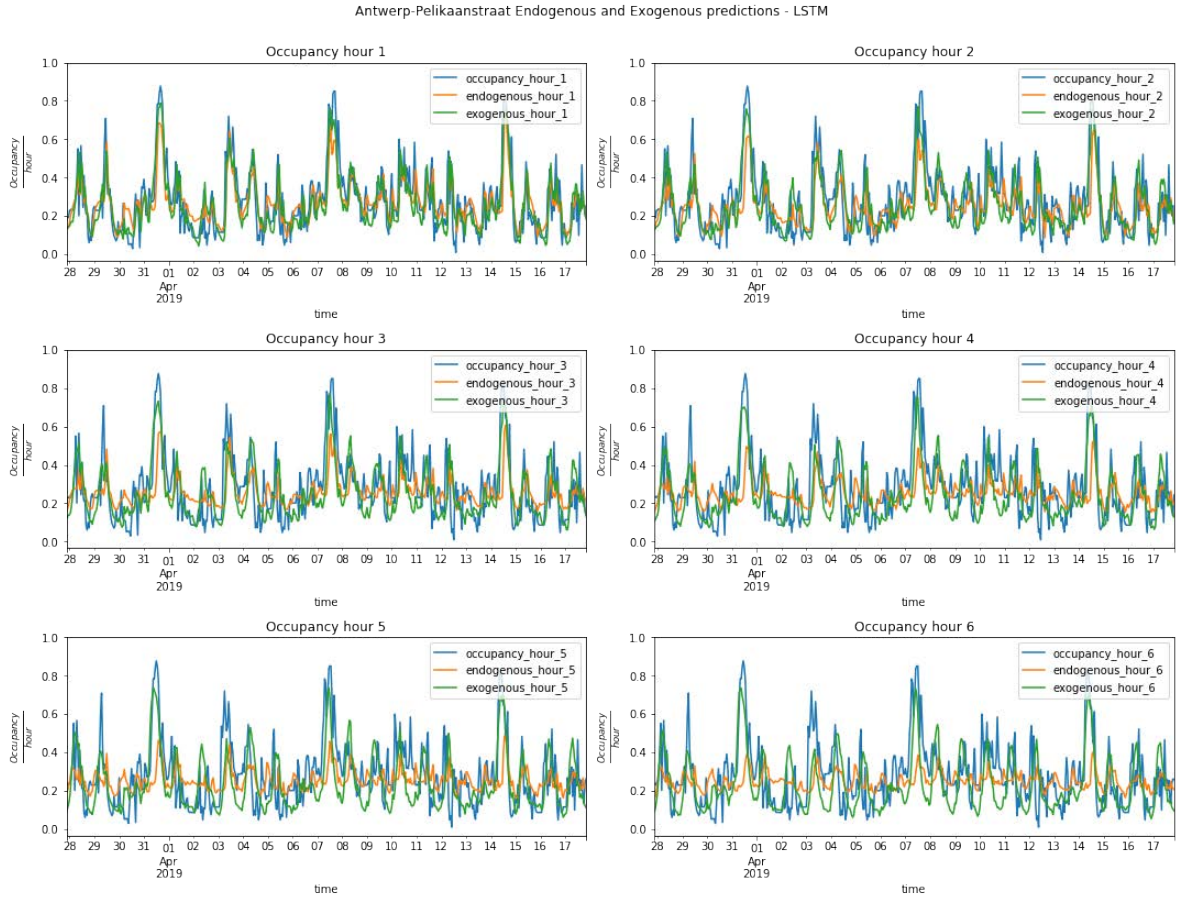


Figure 9.7: Antwerp Pelikaanstraat LSTM model forecasts comparison. Blue belongs to the true parking occupancy, orange to the forecasts from the endogenous model and green to the forecasts from the exogenous model.

mean value of the series, something that does not happens in the forecasts of the exogenous model because the information obtained from the exogenous variables makes the model to differentiate occupancy based on the hours of the day and in the days of the week.

Table 9.7 presents the obtained RMSE values from the forecasts for the Wattens Outside sector when using the endogenous and the exogenous models. The results show that all the NN methods have improved when exogenous information is used in the Wattens scenario. Specifically, MLP shows a great improvement across all the forecasting hours, while LSTM and GRU show a great improvement from the forecasting step two onward.

Figure 9.8 shows the obtained RMSE scores for each of NN models trained with only the endogenous variable and with the exogenous variables. Models trained with exogenous variables present lower error from the forecasting step two onward when comparing with their counterparts.

Figure 9.9 shows the time series forecasts of the models with lower RMSE for the sector

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
MLP_endogenous_wattens_outside	0.167275	0.168808	0.160449	0.161442	0.173651	0.192762	0.173874
MLP_exogenous_wattens_outside	0.086194	0.103884	0.115519	0.123145	0.126526	0.130674	0.11554
LSTM_endogenous_wattens_outside	0.095564	0.132638	0.162885	0.186857	0.191105	0.220024	0.169323
LSTM_exogenous_wattens_outside	0.099849	0.115229	0.125736	0.132120	0.135185	0.139255	0.125309
GRU_endogenous_wattens_outside	0.083591	0.116279	0.139536	0.156237	0.165543	0.174135	0.142176
GRU_exogenous_wattens_outside	0.099434	0.110648	0.117117	0.121229	0.123851	0.125638	0.116672

Table 9.7: Comparison between exogenous and endogenous candidate model RMSE scores in test set for the Wattens scenario.

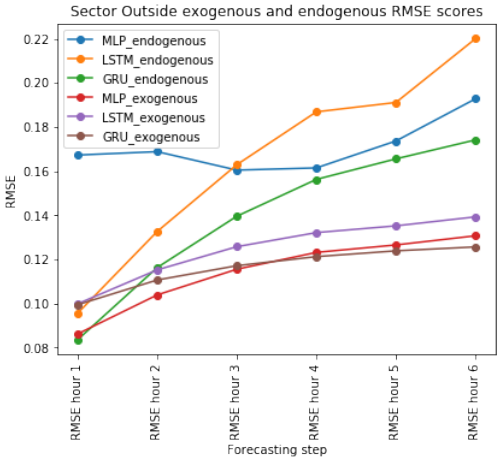


Figure 9.8: RMSE errors from the endogenous and exogenous models for the Wattens scenario.

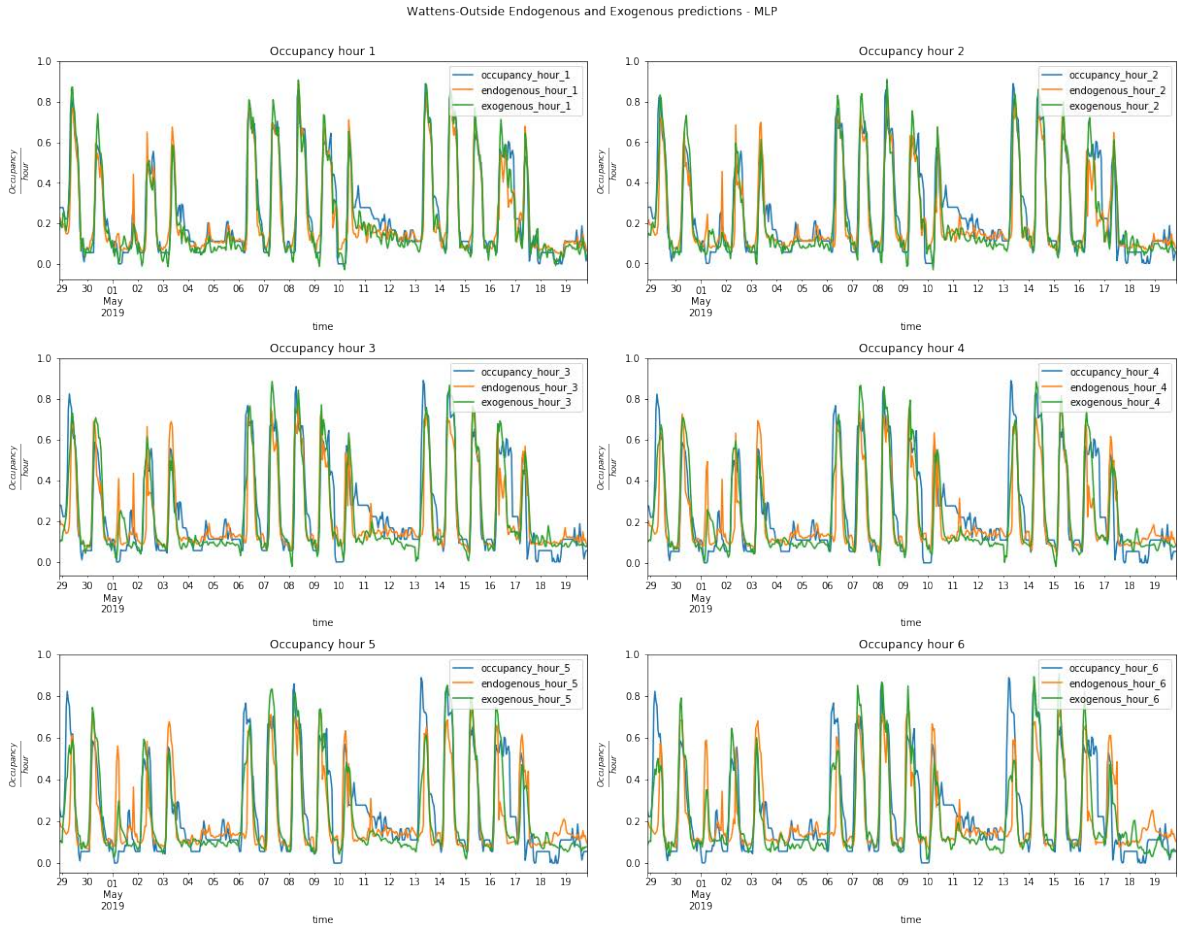


Figure 9.9: Wattens Outside MLP model forecasts comparison. Blue belongs to the true parking occupancy, orange to the forecasts from the endogenous model and green to the forecasts from the exogenous model.

Outside in the Wattens scenario. The visualizations correspond to the models of the MLP method but, must to be considered that GRU has also similar RMSE scores. The endogenous and the exogenous MLP models present similar forecasts values but, the ones from the latter are more aligned with the real parking occupancy in the latter forecasting steps. For May 1st (Labour day), both models forecast low parking occupancy in the first forecasting steps but, at the latter forecasting steps, the endogenous model forecasts a sudden increase, something that the exogenous model does not do until the last forecasting step. Other dates of interest are the days 11 and 12 of May where the occupancy is higher than the usual weekend but the forecasts from both models are lower than the true occupancy.

Table 9.8 shows the obtained RMSE scores by the endogenous and the exogenous models for Standard sector in the scenario of Los Angeles. Only LSTM shows a great improvement in the exogenous model when compared with respect to its endogenous counterpart. MLP and GRU endogenous models present lower errors in all the forecast-

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
MLP_endogenous_LA_Standard	0.083213	0.091379	0.096799	0.102012	0.105445	0.112373	0.099285
MLP_exogenous_LA_Standard	0.133507	0.132600	0.135605	0.134806	0.138004	0.137220	0.135734
LSTM_endogenous_LA_Standard	0.157765	0.170322	0.187332	0.202759	0.221877	0.232236	0.198609
LSTM_exogenous_LA_Standard	0.099506	0.118512	0.121639	0.131558	0.132344	0.132003	0.123919
GRU_endogenous_LA_Standard	0.07801	0.09512	0.10619	0.11393	0.12064	0.12356	0.107401
GRU_exogenous_LA_Standard	0.103600	0.116677	0.125230	0.129723	0.131936	0.132077	0.124002

Table 9.8: Comparison between exogenous and endogenous candidate model RMSE scores in test set for the Los Angeles scenario.

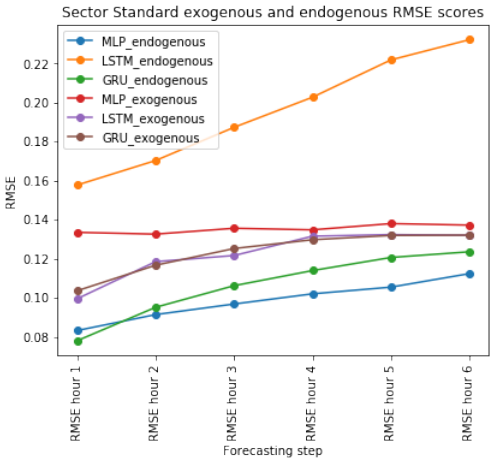


Figure 9.10: RMSE errors from the endogenous and exogenous models for the Los Angeles scenario.

ing steps. Clearly in these cases, exogenous variables are adding noise to the training process of the model.

The visualization in Figure 9.10 shows that the RMSE values obtained for the scenario of Los Angeles have only improved in the case of the LSTM method. For the other methods, the models with endogenous variables present lower scores. The cause of the obtained results can reside in the change of distribution in the data.

Visualizations in Figure 9.11 present the forecasts of the MLP models for the Standard sector in Los Angeles. In this case, the endogenous model has lower RMSE error and this can be perceived in the visualization as the time series corresponding to the endogenous forecasts (orange) has nearly perfect match for the weekends occupancy. Also, the endogenous model forecasts for the weekdays are closer to the true occupancy comparing with the forecasts from the exogenous model. It can be observed than in the latter forecasting step, the exogenous model fails in order to provide good forecasts.

The endogenous and exogenous RMSE scores for the Sector-10 in the Riyadh scenario are presented in Table 9.9. The obtained results show a great improvement in the LSTM and

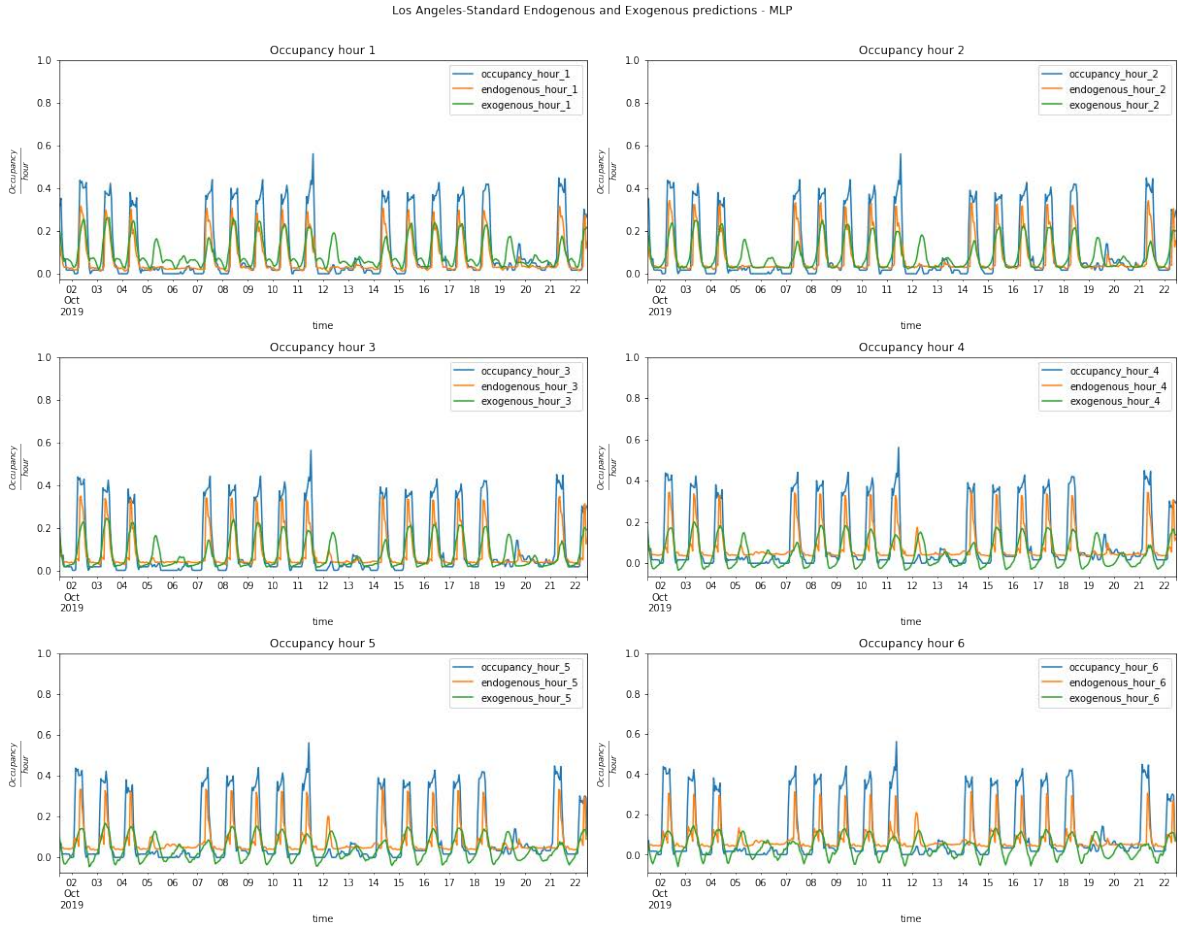


Figure 9.11: Los Angeles Standard MLP model forecasts comparison. Blue belongs to the true parking occupancy, orange to the forecasts from the endogenous model and green to the forecasts from the exogenous model.

Experiment name	RMSE (1 to 6 hours)						Mean RMSE
MLP_endogenous_riyadh_sector10	0.071116	0.081256	0.080439	0.086139	0.097126	0.096646	0.087643
MLP_exogenous_riyadh_sector10	0.054872	0.068552	0.076159	0.080320	0.081744	0.081357	0.07451
LSTM_endogenous_riyadh_sector10	0.078077	0.090971	0.098520	0.108535	0.118905	0.128008	0.104397
LSTM_exogenous_riyadh_sector10	0.049109	0.049818	0.051592	0.052692	0.052850	0.052960	0.051538
GRU_endogenous_riyadh_sector10	0.063591	0.072662	0.081042	0.087759	0.091248	0.097092	0.083619
GRU_exogenous_riyadh_sector10	0.053634	0.057642	0.056152	0.059179	0.060058	0.061906	0.058238

Table 9.9: Comparison between exogenous and endogenous candidate model RMSE scores in test set for the Riyadh scenario.

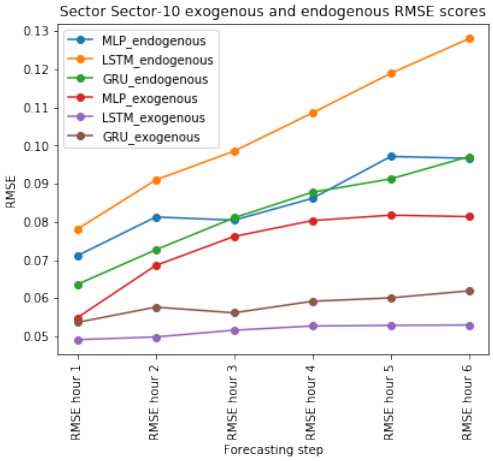


Figure 9.12: RMSE errors from the endogenous and exogenous models for the Riyadh scenario.

GRU models when exogenous information is considered. MLP shows less improvements.

The visualization in Figure 9.12 can help to clearly analyze the evolution of the RMSE on the endogenous and exogenous models. It shows that LSTM method is the one that improves the most from the usage of the exogenous variables followed by the GRU method. MLP also improved with respect to its endogenous counterpart.

Visualizations in Figure 9.13 present the forecasts of the LSTM models for the sector Sector-10 in Riyadh scenario. Both, endogenous and exogenous models, provide very accurate forecasts making hard to distinguish when the exogenous model forecasts are better than those from the endogenous model. In the latter forecasting steps, the exogenous model provides forecasts closer to the true parking occupancy, specially in the peaks and valleys of the time series. One can also observe that both models provide forecasts that differ a little from the true parking occupancy in the days 4, 11 and 18 of October, as those corresponds to Fridays (weekend).

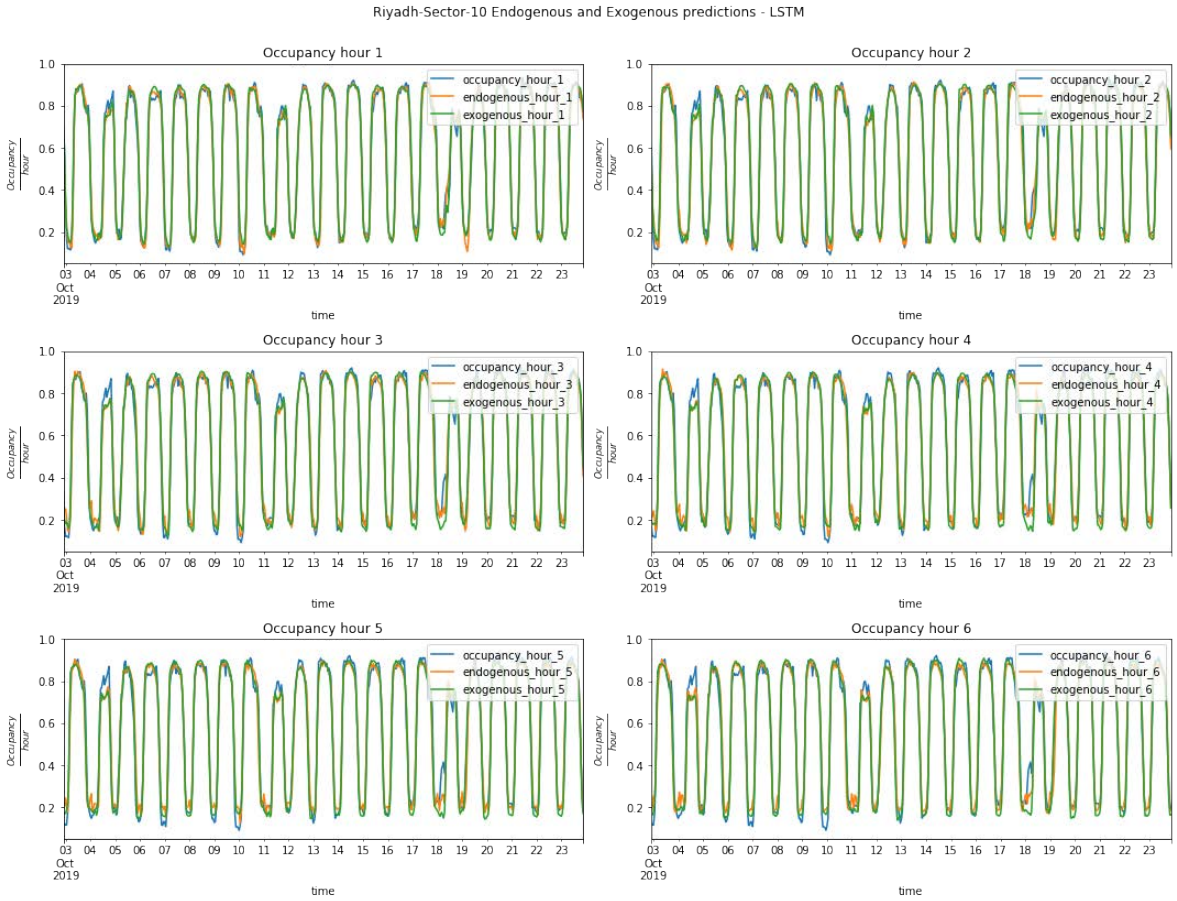


Figure 9.13: Riyadh Sector-10 LSTM model forecasts comparison. Blue belongs to the true parking occupancy, orange to the forecasts from the endogenous model and green to the forecasts from the exogenous model.

9.1.3 Findings by sectors idiosyncrasies

Sectors unique characteristics provide interpretations about the realizations of the processes studied and the obtained goodness-of-fit of the models. Moreover, their identification can help when considering a transfer learning of knowledge between models.

Next, a summary on how the sectors idiosyncrasies have an effect on the models is presented. The findings come from the experiments presented at Chapters 6 and 7.

9.1.3.1 Sector size

It has been observed that the total number of places in a sector has an effect on the sector realizations. Those sectors with a high number of parking spots (Riyadh sectors for example) are less affected by outliers in the realizations and the visualizations on the run charts (the scatterplots with Time at the x-axis) are smooth and continuous. This makes that the time patterns are easy to be recognized and that models trained on these sites are the ones that achieved the lowest error.

On the other hand, those sectors with a low number of places (for example Van Schoonhovestraat Zone 2 and Willow Zipcar) have shown to be governed by random walk processes, with run charts that are not smooth and continuous. Models trained in those sectors presented higher error than the rest.

Mid-size sectors presented an RMSE around 15%.

9.1.3.2 Parking type and parking use

Two important aspects that go hand by hand and have a greatly influence in the behaviour of a parking process are the type of parking places that compose the sector (restrictions to use them) and the reason that the user has in order to leave the car in that specific zone. A very good example of this is the case of Los Angeles where all sectors with the exception of the ZipCar sector present similar patterns in time and that is because the zone is a park-and-ride area where people leaves the car and travels to the work by using the metro. In this case the usage of the parking is the driving force behind the parking occupancy realizations because all the sectors are of different type: Standard are free parking places, Permit corresponds to rented parking places and Disabled are places that can only be used by users with a disabled license. On the other hand, the ZipCar sector has a very different behaviour when compared with the others and the reason this is that these parking places are reserved for the rent-a-car company ZipCar and the usage of the parking places has little in common with the standard usage of the other parking sectors.

When there is a shared context between parking sectors and no restrictions (or few places with restrictions) it is possible to observe that all the parking sectors in the area behave very similar as observed in the Riyadh sectors. This enables the possibility to

Methodology	Number of neurons	Number of layers	Dropout %	Recurrent dropout %	Weights initializer	Activation function	Batch size	Optimizer	Learning rate range	Clipping norms value
MLP	64	4	0.25	-	Does not matter	ReLu	256	RMSProp	$[10^{-4}, 10^{-2}]$	-
LSTM	128, 512	2	0.25	0.	He normal	Linear	64, 128	SGD + Momentum	$[10^{-3}, 10^{-1}]$	0.6, 0.8
GRU	64	2	0.25	0.	He normal	Sigmoid	672	RMSProp	$[10^{-5}, 10^{-2}]$	0.2

Table 9.10: Hyperparameters summary.

train a generic model for all the sectors in the area or if new sectors are aggregated to the existing area new models could be created from the knowledge already learnt by the existing models of the old sectors.

Another interesting scenario is the one of Antwerp where sectors belong to the downtown of the city and they are restricted by pricing policies and time of use. In those sectors one can find through exponential smoothing or by the visualizations obtained from seasonal plots that at each hour of the day the average occupancy is more or less the same and that the variance explains the effect of the restrictions.

9.1.4 NN hyperparameters

During the realization of this dissertation a computational and time effort has been made in order to explore the hyperparameters space of different NN methodologies, namely MLP, LSTM and GRU for parking time series forecasting. The obtained results are of great interest for practitioners in the area as they can be used as guidelines in order to create new models in such type of data by means of reducing the search space. The reduction in the search space makes much more feasible the creation of new models because the task of finding a good subset of values for the hyperparameters its one of the most time-consuming steps of the modeling process with NN.

Table 9.10 presents the most common values for the considered hyperparameters that have been found through the random search algorithm accordingly to the results presented in Chapter 7.

Based on the values of the hyperparameters we checked if there is a relationship between those and the sectors characteristics. Evidences about that relationship can indicate that it is possible to use the same model for those sectors or even transfer the knowledge from the model trained in one sector in order to train a new model on a new sector based on the common features between the two sites.

In order to do the comparisson, we have selected sectors that are very similar between them, like in the Wattens scenario. Observing the values of the hyperparameters for the models trained with data from the Wattens sectors shown in Table 7 it is posible to notice that in the case of RNN, the Inside and Outside models seems to have very similar hyperparameters.

For the sake of finding the elements that define the relationship between the hyperparameter values and the sectors characteristics, Table 9.12 shows the hyperparameter values of the GRU models trained for similar sector sizes. Observe that the hyperparameters values of the models are quite different between them. Similarly is shown in Table

Experiment name	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer
MLP_endogenous_wattens_inside	128	2	0.25	He normal	ReLu	256	RMSProp
MLP_endogenous_wattens_outside	256	5	0.5	He normal	Sigmoid	256	RMSProp
LSTM_endogenous_wattens_inside	16	2	0.25	He normal	Linear	256	SGD
LSTM_endogenous_wattens_outside	32	2	0	He normal	Sigmoid	64	SGD
GRU_endogenous_wattens_inside	32	2	0.25	He normal	Sigmoid	672	RMSProp
GRU_endogenous_wattens_outside	64	2	0.25	He normal	Sigmoid	128	RMSProp

Table 9.11: Hyperparameters values from Wattens models.

Experiment name	Places	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer
GRU_endogenous_antwerp_pelikaanstraat	35	32	1	0.25	He normal	Sigmoid	672	RMSProp
GRU_endogenous_LA_Permit	31	8	2	0.25	He normal	Linear	32	RMSProp
GRU_endogenous_antwerp_vanwesenbekestraat2	12	256	3	0.25	He normal	Sigmoid	672	RMSProp
GRU_endogenous_barcelona_regular	11	256	2	0.25	Glorot normal	Sigmoid	64	RMSProp
GRU_endogenous_antwerp_vanwesenbekestraat	10	16	2	0.25	He normal	Linear	256	RMSProp
GRU_endogenous_antwerp_vanschoonhovestraat1	8	128	3	0.25	He normal	ReLu	128	RMSProp
GRU_endogenous_antwerp_vanschoonhovestraat3	8	8	1	0	Glorot normal	Linear	32	RMSProp

Table 9.12: Hyperparameters values related to sectors size.

9.13 where the models have been considered by their type and use like the Barcelona Regular and the Wattens Outside sector that are private and belong to a company and have similar number of places.

It is hard to discern if there is a relationship between hyperparameter values and sectors characteristics because it is possible that the variance shown in the results for similar sectors can have its source in the inherent stochasticity of the NN methodologies (random initialization, multiple local optimas, etc.), this is reaffirmed when one checks the obtained ARIMA candidate models and sees that sectors with similar characteristics have similar hyperparameters as shown in Table 9.14 (although this can also be because the simplicity of the hyperparameter space of the ARIMA methodology).

9.2 Main Contributions

The main contributions of this thesis are: exhaustive characterizations, studies, modeling and real time parking occupancy forecasts for on-street parking systems in fifteen different scenarios (on-street parking sectors) with unique characteristics.

This section briefly summarizes other contributions presented in this thesis, by order of their appearance in the document:

Experiment name	Places	Number of neurons	Number of layers	Dropout %	Weights initializer	Activation function	Batch size	Optimizer
GRU_endogenous_wattens_outside	18	2	0.25	0.25	He normal	Sigmoid	128	RMSProp
GRU_endogenous_barcelona_regular	11	256	2	0.25	Glorot normal	Linear	32	RMSProp

Table 9.13: Hyperparameters values by sector type.

Experiment name	ARIMA model
ARIMA_antwerp_pelikaanstraat	$SARIMA(0,1,2)(0,1,1)_{24}$
ARIMA_antwerp_vanwesenbekestraat	$SARIMA(1,0,0)(0,1,1)_{24}$
ARIMA_antwerp_vanwesenbekestraat2	$SARIMA(1,0,0)(0,1,1)_{24}$
ARIMA_antwerp_vanschoonhovestraat1	$ARIMA(1,0,0)$
ARIMA_antwerp_vanschoonhovestraat2	$ARIMA(1,0,0)$
ARIMA_antwerp_vanschoonhovestraat3	$ARIMA(1,0,0)$
ARIMA_barcelona_regular	$SARIMA(2,1,2)(0,1,1)_{24}$
ARIMA_wattens_inside	$SARIMA(2,1,2)(0,1,1)_{24}$
ARIMA_wattents_outside	$SARIMA(2,1,2)(0,1,1)_{24}$
ARIMA_LA_standard	$SARIMA(1,1,2)(0,1,2)_{24}$
ARIMA_LA_permit	$SARIMA(1,1,1)(0,1,1)_{24}$
ARIMA_LA_disabled	$SARIMA(0,1,1)(0,1,1)_{24}$
ARIMA_LA_zipcar	$SARIMA(0,1,0)(0,1,1)_{24}$
ARIMA_riyadh_sector1	$SARIMA(1,1,1)(0,1,2)_{24}$
ARIMA_riyadh_sector10	$SARIMA(0,1,1)(1,1,1)_{24}$

Table 9.14: ARIMA models hyperparameters.

- Guidelines for analyzing parking time series by means of statistical methods and visualizations. Moreover, this methodology can also be applied to other data that consist of time series realizations of hourly frequency processes.
- A characterization of different and heterogeneous parking systems according to realizations of parking occupancy over time. This characterization will be profitable for future works that take into account information and clues about modeling these systems.
- The usage of the Box Jenkins methodology for creating ARIMA models through the characterization of parking data, namely time series. By providing insights into the strong and weak points of applying this methodology for the problem at hand, this work can help create services using this technology.
- The modeling of parking systems using different NN methods, namely MLP, LSTM and GRU. This study provides insights into how to organize the data and which are the best ranges of hyperparameters, as well as into how these hyperparameters are related to the heterogeneous characteristics of the parking systems. This information can be of great interest for future research on the topic. Moreover, the information provided can help create services using this technology.
- Exogenous variables, which are information that is external to the parking systems, have been studied under an NN framework in order to extract and interpret their influence on parking. Moreover, guidelines have been provided on how to integrate such data into NN models. We have quantified the errors of NN models with and without exogenous variables and provided insights into the effects of the exogenous variables.

- An exhaustive comparison of the obtained models that highlights the strong and weak points of each method and how they can be improved by further research.
- NN models for time series data using new methods and strategies for NN, which have been developed from 2012 to 2020. At the moment of writing this thesis, the area of NN research focuses mainly on computer vision problems and natural language processing. This work studies the proposals and advancements made in those domains while adapting them to the context of time series.
- Part of the data used in this work is publicly available under an open access license (Creative Commons Attribution 4.0 International) so that it can be used as a benchmark for parking occupancy forecasting. The data corresponds to the cities of Antwerp, Barcelona and Wattens. This data consists of the raw sensor readings, occupation by hour, weather, dates, auxiliary plots, and documentation on the data realizations. All of these formed a part of the data analysis performed in this thesis, which was developed in the context of the European H2020 project FastPrk-2: Enhanced On-Street Parking Management. The data can be found in the following references Hernandez-Ramirez [2018a,b,c].

From the listed contributions, we want to highlight the originality of this dissertation by means of:

1. Methods:
   a) An step-by-step data analysis on time series data that offers interpretability and information from the observations. This permits to identify key points for modeling such data.
   b) An step-by-step description on how to model time series data using NN of any type. Specifically, this thesis provides information about how to preprocess the data, how to divide the train-validation-test sets, advices on how to obtain good candidates values for the hyperparameters of the model, visualizations on the data and how to interpret them in order to solve and/or avoid typical problems that can occur in such models.
   c) Analysis of exogenous variables and its effect by means of using NN techniques like an study of the formed embeddings or the outputs of the models when the endogenous variable is conditioned to each one of the exogenous variables.
2. Findings:
   a) Identification of variables that affect the behaviour of parking systems and that can help in the decision making when dealing with them. These variables are parking usage restrictions (end user, pay per use, park-n-ride, etc.), socio-cultural factors, surroundings (the influence of nearly parking sites and points of interest) and the total number of places.
   b) Parking sectors are highly dependent on the number of parking spots that make it up. Found that those parking sectors with low numer of parking places behave like a random walk process but more regular patterns arise as the number of parking places increases.

c) Mathematically and visible evidences that the recent realizations are the most influential ones in a parking system and that a daily seasonality is present in the process, ensuring the property of weakly stationarity (mean and variance are independent of time). This permits to model such systems with interpretable models like ARIMA or with more complex models like NN.

d) Dense populated areas of sensors divided in neighbourhoods present very similar realizations (with an undetermined ϵ that may be reduced to random noise but this needs to be further investigated) between them and with the area not divided if all the area shares the same values for the variables stated at point a). This has been observed in the Riyadh scenario and can be useful as an assumption in order to simulate parking behaviours in cities.

e) Identified that for a high number of parking systems the searching space of the hyperparameters can be reduced, helping practitioners by reducing the computational needs.

f) Based on the NN method used, some key hyperparameters like the optimizers have very different behaviours when traversing through the searching space looking for optimal optimal values in a time series context. This thesis has identified which optimizers works better with which NN method and can be used with similar time series.

g) GRU models performs better in the endogenous setting but as the complexity grows (increased the space of the data through the exogenous variables) the LSTM models has been shown to be the best candidates.

h) Exogenous variables such as weather and calendar provide useful information to the modeling. Specifically, because parking systems are highly dependent on human daily behaviour, a change on them is reflected as a change on the parking processes. But, some of the information can be redundant and add noise to the modeling process so of all the variables considered only a subset of them are relevant in order to provide better forecasts than the one provided by models that only make use of the endogenous variable. As calendar information directly affects human daily behaviour by providing information about if it is working day or not, so variables of interest are day of the week, holiday and month. On the side of the weather variables, those with a direct effect in human daily behaviour are related to the precipitation type, accumulation and intensity. Bear in mind that these subsets of variables are dependent of the region of the world where the parking sites are located.

3. Innovations:

a) Discern the similarity of parking time series from the same area by means of clustering techniques. Providing a way to summarize the process of the whole area by just using the realization of a subset of the time series that are part of the process. As we know, this method is new in the area of research.

b) How embeddings formed by categorical exogenous variables in a NN reflect aspects of the process modeled and can help in deciding which exogenous variables are useful and also interpret their effect. Although the study of the embeddings has been studied in NLP, as far as we know, there are no studies about the embeddings formed by categorical exogenous variables in the context of time series.

c) How to discern visually the impact of continuous exogenous variables to the endogenous variable by means of using NN. This makes possible to identify which continuous exogenous variables contribute the most in the model. We found no prior usage of NN methods in order to assess the influence of exogenous variables in the context of time series.

d) Proposed NN architectures that can handle exogenous variables for the MLP and RNN cases.

Aside from these main contributions, we will answer the research questions presented in Chapter 1.

- *Forecasting the availability of parking places is not a new problem in the literature. Over the past 20 years, efforts have been made to provide solutions, although without much progress due to the lack of data. However, because we currently have real-time parking status made available by advances in IoT and increasing computational capacity, we can train machine learning models that are capable of learning and identifying complex patterns in the data. Considering these advantages, is it now possible to provide a better solution to the problem of forecasting parking availability?*
  Due to technological constraints, works prior to 2014 could measure parking information only by making use of off-street parking data and data coming from pay-per-parking places in the city. The arrival of better technology for sensors made it possible to collect parking data through platforms of sensors. One of the first works that took advantage of these new sources of on-street parking system data is Vlahogianni et al. [2016], who used well-known NN architectures like MLP and Time Delay NN (a modification of the MLP, in which temporal delays of the data are manually fine-tuned). One-and-a-half years later, no new research has emerged on this topic; so the present work begins from there by applying ARIMA, MLP and RNN architectures in order to study, characterize and model parking systems.
  The results show that it is possible to achieve low-error models that learn recurrent parking patterns and whose forecasts can be used to provide different services while also improving parking system management.
  It is difficult to quantify whether the results presented here are better than the results from other works, as parking data is still owned by a few private companies and the data is usually not shared with the research community.
  Thus, the answer to the question is that this work provided models with low error rates in their parking occupancy forecasts, but it is not possible to quantify by

how much the proposed methods are better than those of other proposals in the literature, as the problem is bound to the parking systems from which the data comes, and this data is usually not available.

- *Creating models is affordable using two different but related disciplines: classical statistical methods and machine learning. Both try to solve the same problems with different philosophies. Specifically, both provide algorithms that enable models to forecast time-dependent data. Which approach would be the most suitable in fulfilling our requirements?*
  This work provides results from both of the mentioned disciplines: ARIMA as a classical statistical family of methods and NN as a machine learning approach. The results from the models show that both methods have fulfilled the requirements of providing forecasting models with low error rates and at different time steps (from one to six hours). The differences between the methods make it evident that the forecast quality of their models depends on the characteristics of the parking system being modeled.
  In the results presented here, both methods have proven to be good candidates for the problem. Specifically, ARIMA has proven to be superior to NN in short-term forecasts and with parking systems of low size. In addition, the ARIMA modeling allows obtaining interpretability and estimation of the uncertainty in the forecasts. On the other hand, NN models are best suited when a huge amount of data is used or for easily including exogenous information that could have an impact on the forecasts. This exogenous information could be easily integrated into the model as time series or as independent input variables. In addition, NN methods have demonstrated less increase in error as the forecasted time step increases. Therefore, they would be of interest for dealing with data at a lower frequency (for example, forecasting minute-by-minute frequency data for one hour to obtain 60 different forecasts).
- *Intuition leads us to question whether sensor data is enough for inferring patterns that govern a parking system. Thus, it is natural to consider other data sources. What other sources can be exploited to improve parking forecasts and what effect do they have?*
  Exogenous data from weather and calendar sources have been considered, as parking/traffic behaviours clearly change according to such information. The results with exogenous variables have shown that they have an effect on the parking systems, providing an improvement in nearly all cases over the models that used only the endogenous information.
- *Models are trained with historical data, but a new installation in a new system (city) lacks this fundamental prerequisite. In order to use the forecast service for new installations, the proposed algorithm must provide models with some sort of transfer learning in order to share knowledge of patterns between models in similar situations. Is this feasible?*
  From the observations presented in this work, the data realizations of parking systems depend on a wide variety of factors pertaining to the parking places'

characteristics (users, time usage constraints, numbers of places, etc.), to their location (near the town hall, leisure spots, points of interest, industrial zones, etc.), and to cultural behaviours in the parking system's country. Although some patterns are common to all parking systems, all of these factors lead to huge variability and make it impossible to transfer knowledge from trained models to new ones. Thus, it would be better to train the models from scratch. Despite that, this work provides enough information to create new models that will provide good forecasts without any need to explore different architectures and hyperparameter combinations when using the methods presented here.

- *Which of the current NN methods is more suitable for handling data with temporal dependencies?*
  Although previous works have used NN methods for time series problems, modern techniques and strategies focus mainly on two areas of research, namely computer vision and natural language tasks. Natural language problems are usually handled by RNN methods, as these allow working with sequences of varying sizes in order to learn language patterns. The starting point of this work has taken a similar approach to those of the natural language domain, because the natural order of time series elements is of great importance, just as it is in language sequences. Based on this, the work presented here has explored two of the most common RNN architectures: LSTM and GRU. Although GRU is a modification of LSTM, we have found that it provides better results than LSTM and slightly better results than an MLP architecture, in which the order of time is not taken into account.

9.3 Future Research

Although much has been accomplished by this work, it is just a starting point for this area of research on the study of parking systems. Moreover, this work contributes to a further expansion of using NN methods in the time series domain. In this section, we discuss the directions for future research that will naturally follow the research described in this thesis.

- It has been shown that, apart from time dependencies, some of the parking systems studied have location dependencies (for example, in the Riyadh sectors). It would be of interest to characterize and model how the occupation patterns are correlated between sectors and their distances.
- As stated previously, some of the parking systems studied have spatial dependencies. We propose three ways for future research in order to exploit such dependencies:
  1. Adapt the NN models presented in this work by using as exogenous variables the actual and past data realizations of nearby sectors where significant correlations has been detected through cross-correlation.

2. Vector-valued and multidimensional series. A generalization of "classical" time series approach like ARIMA where a vector of mean value functions and a matrix of correlations (diagonal values are autocorrelations and the others are cross-correlations) are estimated from a collection of time series related to a stochastic process of interest. This framework makes posible then to use vector auto-regressive models or kriging models (Gaussian process regression).
3. Model the parking systems as a graph and use Graph Neural Networks (GNN) Veličković et al. [2018] in order to forecast the evolution of the graph in time by taking into account the dynamics that happen between the nodes and vertices.

- This work used MLP and RNN to provide forecasts, but the problem of long-term dependency is far from being solved and has an active area of research where new methods are constantly proposed. Recently there have been two new approaches that offer promising results, namely convolutional neural networks van den Oord et al. [2016] and attention models Vaswani et al.. In particular, attention models have become the state of the art in the NLP research area through the Transformer model. This model has been designed in order to handle sequences but it can be extended in order to be used with time series data by exploiting the properties of such type of data. These new approaches can improve the presented work by means of the need of less computational resources, better handle of the long term dependencies or added interpretability to the models.
- This work used ARIMA and NN methods in order to modelize time series data from different parking systems but, as exposed in Chapter 4, there exists other methodologies for time series with a well-established theoretical background that can provide new insights or be more appropiate when facing specific scenarios. We consider that state space models (Kalman filter, particle filter, etc) can enhance further the parking forecasting research area.
- Parking has an impact on traffic (and vice versa) and this has not been explored in this thesis; so, in addition to improving models for parking occupancy forecasting, characterizing the relationship between the two processes could also help improve models that study and forecast traffic speed, density or flow in urban areas.
- Parking data comes from sensors that can be subject to noise and provide missing data or data with errors. Detecting, quantifying and correcting these could be the next step in incentivizing the usage of such technology.
- As stated previously, the limitation to the usage of data coming from sensor networks in specific areas makes that the used data comes from very limited and closed systems that are far to represent the global dynamics that happens in the parking processes of a whole city. The study of parking data from other sources like GPS, vanets or connected cars can extend this work and also be extended by taking inspiration by the ideas presented here.
- Low error rates from sensor realizations and from model forecasts make it possible

to use operational research techniques to optimize security and the pay-per-parking routes of guards.

- It has been shown that different parking sites have common properties that can be exploited in order to transfer the learnt knowledge from a trained model in a sector with a good amount of historical data to create new models for new sectors with very little historical data. Specification and experimentation about the findings provided in this work related to transfer learning can be of great interest for the parking forecasting area of research.
- It has been observed that the behaviour of the occupancy at parking sectors is mainly influenced by the unique characteristics of each one of them but it is hard to discern the contribution of each characteristic to the process. It would be of great interest to quantify the effect of these characteristics in the time series realizations of different parking sectors.
- A methodology about how to detect the influence of the exogenous variables into the endogenous variables by means of non-linear estimators (NN) have been presented in Chapter 8. Although it has been useful in order to discern which exogenous variables have an influence on the parking processes, it does not provide measurables quantifications about the influence and so, as future research, it would be very interesting to extend this methodology in order to provide quantifications (sign and magnitude) of the influence of the exogenous variables onto the endogenous variable.

9.4 Related Publications, Presentations, Teaching Activities and Projects

In this final section we enumerate the activities developed in relation to this thesis, all of which were carried out during the same period of time (2017–2020).

Publications

- Arjona, J., Linares, M.P., Casanovas, J., Vázquez, J.J. (2020). "Improving Parking Availability Information Using Deep Learning Techniques". Transportation Research Procedia, 47, 385-392. Elsevier.
https://doi.org/10.1016/j.trpro.2020.03.113
- Arjona, J., M.P. Linares, Casanovas, J. (2019). "A deep learning approach to real-time parking availability prediction for smart cities". DATA '19: Proceedings of the Second International Conference on Data Science, E-Learning and Information Systems. Dubai. December 2019, Article No.: 16, pp 1–7.
https://doi.org/10.1145/3368691.3368707

- Vázquez, J.J., Arjona, J., Linares, M.P., Casanovas, J. (2020). "A Comparison of Deep Learning Methods for Urban Traffic Forecasting using Floating Car Data". Transportation Research Procedia, 47, 195-202. Elsevier. https://doi.org/10.1016/j.trpro.2020.03.079

Submitted publications

- Arjona, J., Linares, M.P., Casanovas , J. "A Neural Network Approach to Long-Term Parking Availability Forecast Using Real-Time Sensor Data, Weather Conditions and Calendar Effect" to the Journal of Advanced Transportation (October 2019).
- Arjona, J., Linares, M.P., Casanovas , J. "Characterizing parking systems from sensor data through a data-driven approach" submitted to the Transportation Letters journal (March 2020).
- Arjona, J., Linares, M.P., Casanovas , J. "A data-driven approach to model parking systems from real-time sensor data, calendar effects and weather conditions" submitted to the Transportation Research Part C journal (April 2020).

Presentations at International Conferences

- Arjona, J., Linares, M.P., Casanovas, J., Vázquez, J.J. (2019). "Improving Parking Availability Information Using Deep Learning Techniques". The 22nd Euro Working Group on Transportation Meeting, September 2019, Barcelona.
- Arjona, J., Linares, M.P., Casanovas, J. (2019). "A deep learning approach to real-time parking availability prediction for smart cities". DATA '19: Second International Conference on Data Science, E-Learning and Information Systems. December 2019, Dubai.
- Arjona, J., M.P. Linares, Casanovas, J. (2019). "Improving Parking Availability Information Using Deep Learning Techniques". Poster presentation at Deep Learning Barcelona 2019. December, 19–20th, 2019, Barcelona.
- Vázquez, J.J., Arjona, J., Linares, M.P., Casanovas, J. (2019). "A Comparison of Deep Learning Methods for Urban Traffic Forecasting using Floating Car Data". The 22nd Euro Working Group on Transportation Meeting, September 2019, Barcelona.
- Gardella, R., Arjona, J., Linares, M.P., Casanovas, J. (2020). "A Simulation-based Evaluation of Travel Time Prediction Algorithms on Freeways using Floating Car Data". Accepted for presentation in the 23rd Euro Working Group on Transportation Meeting, September 2020, Chipre.

Teaching activities

- Meet-upc: “Data science for smarter cities”. PyLadies Bcn Meet-up June, 14th, 2018, Barcelona.
- Seminar: "Data analytics in societal challenges modeling: smart mobility and other related fields" in the PATC: Big Data Analytics course in the Barcelona Supercomputing Center, February, 6th, 2020, Barcelona.
- Master course: "Ambit intelligence", Master’s degree in Automatic Systems and Industrial Electronics Engineering, Universitat Politècnica de Catalunya, February–June 2020, Barcelona.
- Supervision of three master’s theses:
  - “Traffic forecasting in smart cities”. (2019). Vázquez, J.J. http://hdl.handle.net/2117/131604
  - “Travel time forecasting using probe vehicle data on freeways”. (2019). Gardella, R. http://hdl.handle.net/2117/170076
  - (in process) “Modelling and forecasting bus passenger occupancy using data-driven methods”. (2021). Recio, G.

Competitive European Projects

- FastPrk-2: Enhanced On-Street Parking Management (2016-2018). Call: H2020-SMEINST-2-2016-2017. Topic: SMEInst-10-2016-2017 - Small business innovation research for Transport and Smart Cities Mobility. Developed task: Artificial Intelligence algorithms for automatic recommendation and predictive applications of on-street parking.
- BIG IoT - Bridging the Interoperability Gap of the Internet of Things (2016-2019). Call: H2020-ICT-2015. Topic: ICT-30-2015 Internet of Things and Platforms for Connected Smart Objects. Developed task: Enforcement service for parking spots in Barcelona pilot area.

Bibliography

Mohamed S Ahmed. Analysis of Freeway Traffic Time-Series Data by Using Box-Jenkins Techniques. *Transportation Research Board*, (722):1–9, 1979. URL http://onlinepubs.trb.org/Onlinepubs/trr/1979/722/722-001.pdf.

Ding AiLing, Zhao XiangMo, and Jiao LiCheng. Traffic flow time series prediction based on statistics learning theory. *IEEE 5th International Conference on Intelligent Transportation Systems. Proceedings, 3-6 Sept. 2002*, (60073053):727–730, 2002. doi: 10.1109/itsc.2002.1041308. URL http://dx.doi.org/10.1109/ITSC.2002.1041308.

Constantinos Antoniou, Moshe Ben-akiva, and Haris N Koutsopoulos. Kalman Filter Applications for Traffic Management. *Intech*, (May), 2010.

Constantinos Antoniou, Jaume Barceló, Martijn Breen, Manuel Bullejos, Jordi Casas, Ernesto Cipriani, Biagio Ciuffo, Tamara Djukic, Serge Hoogendoorn, and Vittorio Marzano. Towards a generic benchmarking platform for origin/destination flows estimation / updating algorithms : Design , demonstration and validation. *Transportation Research Part C*, 66:79–98, 2016. ISSN 0968-090X. doi: 10.1016/j.trc.2015.08.009. URL http://dx.doi.org/10.1016/j.trc.2015.08.009.

Luigi Atzori, Antonio Iera, Giacomo Morabito, and A Diee. The Internet of Things: A survey. *Computer Networks xxx*, 2010. doi: 10.1016/j.comnet.2010.05.010. URL https://www.cs.mun.ca/courses/cs6910/IoT-Survey-Atzori-2010.pdf.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. jul 2016. URL http://arxiv.org/abs/1607.06450.

Claudio Badii, Paolo Nesi, and Irene Paoli. Predicting Available Parking Slots on Critical and Regular Services by Exploiting a Range of Open Data. *IEEE Access*, 6:44059–44071, 2018. ISSN 2169-3536. doi: 10.1109/ACCESS.2018.2864157. URL https://ieeexplore.ieee.org/document/8430514/.

Baher Abdulhai, Himanshu Porwal, and Will Recker. Short Term Freeway Traffic Flow Prediction Using Genetically-Optimized Time-Delay-Based Neural Networks. (iv), 1998.

James Bergstra and Yoshua Bengio. Random Search for Hyper-Parameter Optimization Yoshua Bengio. *Journal of Machine Learning Research*, 13:281–305, 2012. doi: 10.1.1.306.4385. URL http://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf.

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Phil Blythe, Yanjie Ji, Weihong Guo, Wei Wang, and Dounan Tang. Short-term forecasting of available parking space using wavelet neural network model. *IET Intelligent Transport Systems*, 9(2):202–209, 2015. ISSN 1751-956X. doi: 10.1049/iet-its.2013.0184. URL `http://digital-library.theiet.org/content/journals/10.1049/iet-its.2013.0184`.

G. E. Box and D. R. Cox. An analysis of transformations revisited, rebutted. *Journal of the American Statistical Association*, 77(377):209–210, 1982. ISSN 1537274X. doi: 10.1080/01621459.1982.10477788.

G E P Box, G M Jenkins, and G C Reinsel. *Time Series Analysis: Forecasting and Control*. 1976. ISBN 8131716333. doi: 10.1016/j.ijforecast.2004.02.001.

Leo Breiman. Bagging Predictors. Technical report, 1994.

Leo Breiman. Statistical Modeling: The Two Cultures. Technical Report 3, 2001. URL `http://www.recognition.su/wiki/images/8/85/Breiman01stat-ml.pdf`.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral Networks and Deep Locally Connected Networks on Graphs. In *ICLR2014*, 2013. URL `https://arxiv.org/pdf/1312.6203.pdf`.

Felix Caicedo, Carola Blazquez, and Pablo Miranda. Prediction of parking space availability in real time. *Expert Systems With Applications*, 39:7281–7290, 2012. doi: 10.1016/j.eswa.2012.01.091. URL `https://www.sciencedirect.com/science/article/pii/S0957417412001042`.

Murat Caliskan, Andreas Barthels, Bjorn Scheuermann, and Martin Mauve. Predicting Parking Lot Occupancy in Vehicular Ad Hoc Networks. *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*, pages 277–281, 2007. ISSN 1550-2252. doi: 10.1109/VETECS.2007.69. URL `http://ieeexplore.ieee.org/document/4212497/`.

Enrique Castillo, José María Menéndez, and Santos Sánchez-Cambronero. Predicting traffic flow using Bayesian networks. *Transportation Research Part B: Methodological*, 42(5):482–509, 2008. ISSN 01912615. doi: 10.1016/j.trb.2007.10.003.

Hao a Chen and Hesham A. a Rakha. Real-time travel time prediction using particle filtering with a non-explicit state-transition model. *Transportation Research Part C: Emerging Technologies*, 43:112–126, 2014. ISSN 0968090X. doi: 10.1016/j.trc.2014.02.008. URL `http://www.sciencedirect.com/science/article/pii/S0968090X14000400`.

Zhirong Chen, Jianhong Cecilia Xia, and Buntoro Irawan. Development of fuzzy logic forecast models for location-based parking finding services. *Mathematical Problems in Engineering*, 2013. ISSN 1024123X. doi: 10.1155/2013/473471.

Xingyi Cheng, Ruiqing Zhang, Jie Zhou, and Wei Xu. DeepTransport: Learning Spatial-Temporal Dependeny for Traffic Condition Forecasting. In *2018 International Joint Conference on Neural Networks*, page 17, 2018. URL `https://arxiv.org/pdf/1709.09585.pdf`.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014. ISBN 9781937284961. doi: 10.3115/v1/W14-4012. URL `http://arxiv.org/abs/1409.1259`.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann Lecun. The Loss Surfaces of Multilayer Networks. In *Proceedings of Machine Learning Research 38*, volume 38, pages 192–204. Microtome Publishing, 2015. URL `http://proceedings.mlr.press/v38/choromanska15.html`.

Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. Traffic Graph Convolutional Recurrent Neural Network: A Deep Learning Framework for Network-Scale Traffic Learning and Forecasting. 2018. URL `https://arxiv.org/ftp/arxiv/papers/1802/1802.07007.pdfhttps://github.com/zhiyongc/Graph{_}Convolutional{_}LSTM`.

Justin Dauwels, Aamer Aslam, Muhammad Tayyab Asif, Xinyue Zhao, Nikola Mitro Vie, Andrzej Cichocki, and Patrick Jaillet. Predicting traffic speed in urban transportation subnetworks for multiple horizons. *2014 13th International Conference on Control Automation Robotics and Vision, ICARCV 2014*, pages 547–552, 2014. doi: 10.1109/ICARCV.2014.7064363.

Alysha M. de Livera, Rob J. Hyndman, and Ralph D. Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*, 106(496):1513–1527, 2011. ISSN 01621459. doi: 10.1198/jasa.2011.tm09771.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Advances in Neural Information Processing Systems 29 (2016)*, page 9, 2016. URL `https://github.com/mdeff/cnn{_}graphhttps://github.com/xbresson/spectral{_}graph{_}convnets`.

El Periodico. El 17\% del tráfico de Barcelona son coches en busca de aparcamiento, 2012. URL `https://www.elperiodico.com/es/barcelona/20121228/el-17-del-trafico-de-barcelona-son-coches-en-busca-de-aparcamiento`.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990. ISSN 03640213. doi: 10.1016/0364-0213(90)90002-E.

Eurostat. *Energy, transport and environment indicators. 2018 edition.* 2018. ISBN 978-92-79-96509-8. doi: 10.2785/94549. URL `htttp://dx.doi.org/10.2785/94549`.

Petr Fedchenkov, Theodoros Anagnostopoulos, Arkady Zaslavsky, Klimis Ntalianis, Inna Sosunova, and Oleg Sadov. An Artificial Intelligence Based Forecasting in Smart Parking with IoT. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, number November, chapter ruSMART: N, pages 33–40. Sant Petersburg, 2018. ISBN 9783030011680. doi: 10.1007/978-3-030-01168-0_65.

Rui Fu, Zuo Zhang, and Li Li. Using LSTM and GRU neural network methods for traffic flow prediction. In *Proceedings - 2016 31st Youth Academic Annual Conference of Chinese Association of Automation, YAC 2016*, pages 324–328, 2017. ISBN 9781509044238. doi: 10.1109/YAC.2016.7804912.

Eric Gantelet and Amélie Lefauconnier. The time looking for a parking space: strategies, associated nuisances and stakes of parking management in France. 2008. URL `http://www.mit.edu/course/11/11.951/OldFiles/oldstuff/albacete/Other{_}Documents/EuropeTransportConference/traffic{_}engineering{_}an/the{_}time{_}looking{_}f1580.pdf`.

Dipankar Ghosh and C.H. H Knapp. Estimation of traffic variables using a linear model of traffic flow. *Transportation Research*, 12(6):395–402, 1978. ISSN 00411647. doi: 10.1016/0041-1647(78)90027-8. URL `http://www.sciencedirect.com/science/article/pii/0041164778900278`.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010. URL `http://www.iro.umontreal.`

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep Sparse Rectifier Neural Networks. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA, 2011. PMLR. URL `http://proceedings.mlr.press/v15/glorot11a.html`.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, page 9, Montreal, Canada, 2014. MIT Press. doi: 2969033.2969125. URL `http://www.github.com/goodfeli/adversarial`.

Vincent Le Guen and Nicolas Thome. Shape and Time Distortion Loss for Training Deep Time Series Forecasting Models. (NeurIPS):1–13, sep 2019. URL `http://arxiv.org/abs/1909.09020`.

Cheng Guo and Felix Berkhahn. Entity Embeddings of Categorical Variables. Technical report, 2016. URL `https://www.kaggle.com/c/rossmann-store-sales`.

Robert C. Hampshire and Donald Shoup. What Share of Traffic is Cruising for Parking? *Journal of Transport Economics and Policy*, 52(3):184–201, 2018.

Robert Cornelius Hampshire, Kats Sasanuma, Tayo Fabusuyi, Robert C Hampshire, Victoria Hill, and Katsunobu Sasanuma. A Predictive Model and Evaluation Framework for Smart Parking: The Case of ParkPGH. 2011. URL `https://www.researchgate.net/publication/268128841`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034. Microsoft Research, IEEE, dec 2015. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.123. URL `http://ieeexplore.ieee.org/document/7410480/`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 770–778, 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.90. URL `http://image-net.org/challenges/LSVRC/2015/`.

Mikael Henaff, Joan Bruna, and Yann Lecun. Deep Convolutional Networks on Graph-Structured Data. Technical report, 2015. URL `https://arxiv.org/pdf/1506.05163.pdf`.

Francisco Hernandez-Ramirez. Fastprk2 - Data (pilot Barcelona - UPC). jul 2018a. doi: 10.5281/ZENODO.1313667. URL `https://zenodo.org/record/1313667`.

Francisco Hernandez-Ramirez. Fastprk2 - Data (pilot Antwerp). jul 2018b. doi: 10.5281/ZENODO.1313673. URL `https://zenodo.org/record/1313673`.

Francisco Hernandez-Ramirez. Fastprk2 - Data (pilot Wattens). jul 2018c. doi: 10.5281/ZENODO.1313810. URL `https://zenodo.org/record/1313810`.

Sepp Hochreiter. *Fundamental Deep Learning Problem*. PhD thesis, Technische UniversitÂšat Munchen, 1991. URL `http://people.idsia.ch/{~}juergen/SeppHochreiter1991ThesisAdvisorSchmidhuber.pdf`.

Sepp Hochreiter and Jj Urgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. URL `http://www.bioinf.jku.at/publications/older/2604.pdf`.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 08936080. doi: 10.1016/0893-6080(89)90020-8.

Eric Hsueh, Chan Lu, and Chen-Hao Liao. *A Parking Occupancy Prediction Approach Based on Spatial and Temporal Analysis*. Springer, Cham, 2018. doi: 10.1007/978-3-319-75417-8_47.

IBM. IBM News room - 2011-09-08 IBM Global Commuter Pain Survey: Traffic Congestion Down, Pain Way Up - United States, 2011. URL `https://www-03.ibm.com/press/us/en/pressrelease/35359.wss`.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *32nd International Conference on Machine Learning, ICML 2015*, 1:448–456, feb 2015. URL `http://arxiv.org/abs/1502.03167`.

Andrei Ionita, Michael Cochez, Tobias Meisen, and Stefan Decker. Where to Park? Predicting Free Parking Spots in Unmonitored City Areas. 18, 2018.

doi: 10.1145/3227609.3227648. URL http://users.jyu.fi/{~}miselico/papers/parking{_}paper.pdf.

Sherif Ishak, Prashanth Kotha, and Ciprian Alecsandru. Optimization of Dynamic Neural Network Performance for Short-Term Traffic Prediction. *Transportation Research Record: Journal of the Transportation Research Board*, 1836:45–56, 2003. ISSN 0361-1981. doi: 10.3141/1836-07. URL http://trrjournalonline.trb.org/doi/10.3141/1836-07.

Hongxia Jin. The application of neuro-FDT in urban short-term traffic flow prediction. *Proceedings - Third International Conference on Natural Computation, ICNC 2007*, 5 (Icnc):499–503, 2007. doi: 10.1109/ICNC.2007.710.

Michael Jones, Aftab Khan, Mahesh Sooriyabandara, Parag Kulkarni, and Pietro Carnelli. ParkUs 2.0: Automated Cruise Detection for Parking Availability Inference. 2018. doi: 10.1145/3144457.3144495. URL https://www.researchgate.net/publication/322387223.

Shinobu Kawashima, Kajiro Watanabe, and Kazuyuki Kobayashi. Traffic Condition Monitoring by Laser Radar for Advanced Safety Driving. pages 299–303, 1995.

Eamonn Keogh, Jessica Lin, and Ada Fu. HOT SAX: Finding the most Unusual Time Series Subsequences: Algorithms and Application. *Icdm*, 2005. ISSN 1550-4786. doi: 10.1109/ICDM.2005.79.

Abhirup Khanna and Rishi Anand. IoT based smart parking system. In *2016 International Conference on Internet of Things and Applications, IOTA 2016*, 2016. ISBN 9781509000449. doi: 10.1109/IOTA.2016.7562735.

Thomas N Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR '17)*, page 14, 2016. URL https://arxiv.org/pdf/1609.02907.pdf.

Andreas Klappenecker, Hyunyoung Lee, Jennifer L Welch, Klappenecker Andreas, Lee Hyunyoung, and L Welch Jennifer. Finding Available Parking Spaces Made Easy. 2010. URL https://www.sciencedirect.com/science/article/pii/S157087051200042X.

Andrew Koster, Allysson Oliveira, Orlando Volpato, Viviane Delvequio, and Fernando Koch. Recognition and recommendation of parking places. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8864:675–685, 2014. ISSN 16113349. doi: 10.1007/978-3-319-12027-0_54.

Amir O. Kotb, Yao Chun Shen, and Yi Huang. Smart Parking Guidance, Monitoring and Reservations: A Review. *IEEE Intelligent Transportation Systems Magazine*, 9 (2):6–16, 2017. ISSN 19411197. doi: 10.1109/MITS.2017.2666586.

Mark A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, 1991. ISSN 15475905. doi: 10.1002/aic.690370209.

Thomas J Laetz. Predictions and Perceptions Defining the Traffic Congestion Problem. 1990. URL https://www.sciencedirect.com/science/article/pii/0040162590900746.

Lawrence W Lan and April Y Kuo. Journal of the Eastern Asia Society for Transportation Studies, Vol. 6, pp. 1518 - 1533, 2005. 6:1518–1533, 2005.

Jorge Lanza, Luis Sánchez, Verónica Gutiérrez, José Galache, Juan Santana, Pablo Sotres, and Luis Muñoz. Smart City Services over a Future Internet Platform Based on Internet of Things and Cloud: The Smart Parking Case. *Energies*, 2016. ISSN 1996-1073. doi: 10.3390/en9090719.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the Loss Landscape of Neural Nets. In S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 6389–6399. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7875-visualizing-the-loss-landscape-of-neural-nets.pdfhttps://github.com/tomgoldstein/loss-landscape.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations (ICLR '18)*, page 16, 2017. URL https://arxiv.org/pdf/1707.01926.pdfhttps://github.com/liyaguang/DCRNNhttp://arxiv.org/abs/1707.01926.

Trista Lin, Herve Rivano, and Frederic Le Mouel. A Survey of Smart Parking Solutions. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3229 – 3253, 2017. ISSN 15249050. doi: 10.1109/TITS.2017.2685143.

Seppo Linnainmaa. Taylor expansion of the accumulated rounding error. *Bit*, 16(2): 146–160, 1976. ISSN 15729125. doi: 10.1007/BF01931367.

J W C Van Lint, C Hinsbergen, J W C Van Lint, and F Sanders. Short Term Traffic Prediction Models. *14th World Congress on Intelligent Transport Systems, ITS 2007*, 7, 2007. URL https://www.researchgate.net/publication/228882378.

Los Angeles Times. L.A. program aims to make parking easier - Los Angeles Times, 2010. URL https://www.latimes.com/archives/la-xpm-2010-aug-22-la-me-express-park-20100822-story.html.

Jorge Ludlow and Walter Enders. Estimating non-linear ARMA models using Fourier coefficients. *International Journal of Forecasting*, 16(3):333–347, 2000. ISSN 01692070. doi: 10.1016/S0169-2070(00)00048-0.

Mohammad M. Hamed, Hashem R. Al-Masaeid, and Zahi M. Bani Said. Short-term prediction of traffic volume in urban arterials. *Division of UrbanTraffic*, I(June):249–254, 1995.

David J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.3.448. URL http://www.mitpressjournals.org/doi/10.1162/neco.1992.4.3.448.

Niall McCarthy. The World's Worst Cities For Traffic Congestion [Infographic], 2019. URL https://www.forbes.com/sites/niallmccarthy/2019/06/05/the-worlds-worst-cities-for-traffic-congestion-infographic/{#}2769841e12bc.

Marcelo C Medeiros, Timo Teräsvirta, and Gianluigi Rech. Building Neural Network Models for Time Series: A Statistical Approach. 2006. doi: 10.1002/for.974. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/for.974.

Lyudmila Mihaylova, René Boel, and A Hegyi. An Unscented Kalman Filter for Freeway Traffic Estimation. *Journal of Chemical Information and Modeling*, pages 31–36, 2006. ISSN 1098-6596. doi: 10.1017/CBO9781107415324.004.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In C J C Burges, L Bottou, M Welling, Z Ghahramani, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013. URL https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

Bandu Nandekar, Ranjana Raut SGB Amravati, and Pravin Dakhole YCCE Nagpur. Parking System with Space Modeling to Improve Parking Efficiency and Parking Revenue. 2015. doi: 10.1109/ICCN.2015.51.

H Nicholson and C D Swann. The prediction of traffic flow volumes based on spectral analysis. *Tra˜zspn Rus*, 8:533–538, 1974. URL https://www.sciencedirect.com/science/article/pii/0041164774900306.

Iwao Okutani and Yorgos J. Stephanedes. Dynamic prediction of traffic volume through Kalman filtering theory. *Transportation Research Part B*, 18(1):1–11, 1984. ISSN 01912615. doi: 10.1016/0191-2615(84)90002-X.

Catriona Queen Osvaldo Anacleto and Casper J . Albers. Multivariate forecasting of road traffic flows in the presence of heteroscedasticity and measurement errors. *Journal, Source Statistical, Royal Series, Society Statistics, C Applied*, 62(2):251–270, 2013.

Nicola Piovesan, Leo Turi, Enrico Toigo, Borja Martinez, and Michele Rossi. Data analytics for smart parking applications. *Sensors (Switzerland)*, 2016. ISSN 14248220. doi: 10.3390/s16101575.

John W Polak, Ian C Hilton, Ian Hilton Consultancy, Kay W Axhausen, and William Young. Parking Guidance Systems: Current Practice and Future Prospects. 1990. URL http://webarchiv.ethz.ch/ivt/vpl/publications/reports/o9.pdf.

Nicholas G. Polson and Vadim O. Sokolov. Deep learning for short-term traffic flow prediction. *Transportation Research Part C: Emerging Technologies*, 2017. ISSN 0968090X. doi: 10.1016/j.trc.2017.02.024.

Tooraj Rajabioun and Petros Ioannou. On-Street and off-street parking availability prediction using multivariate spatiotemporal models. *IEEE Transactions on In-*

telligent Transportation Systems, 16(5):2913 – 2924, 2015. ISSN 15249050. doi: 10.1109/TITS.2015.2428705.

Tooraj Rajabioun, Brandon Foster, and Petros Ioannou. Intelligent parking assist. *2013 21st Mediterranean Conference on Control and Automation, MED 2013 - Conference Proceedings*, pages 1156–1161, 2013. doi: 10.1109/MED.2013.6608866.

Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Quoc V Le, Alexey Kurakin, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-Scale Evolution of Image Classifiers. *CoRR*, abs/1703.0, 2017. URL `http://arxiv.org/abs/1703.01041`.

Felix Richter, Sergio Di Martino, and Dirk C. Mattfeld. Temporal and Spatial Clustering for a Parking Prediction Service. *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, 2014-Decem:278–282, 2014. ISSN 10823409. doi: 10.1109/ICTAI.2014.49.

F. Rosenblatt. The Perceptron - A Perceiving and Recognizing Automaton, 1957.

Royal Society. Machine Learning: The power and promise of computers that learn by example. 2017. URL `https://royalsociety.org/topics-policy/projects/machine-learning/`.

Sebastian Ruder. An overview of gradient descent optimization algorithms. pages 1–14, 2016. ISSN 0006341X. doi: 10.1111/j.0006-341X.1999.00591.x. URL `http://arxiv.org/abs/1609.04747`.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, oct 1986. ISSN 0028-0836. doi: 10.1038/323533a0. URL `http://www.nature.com/articles/323533a0`.

Abolfazl Safikhani, Camille Kamga, Sandeep Mudigonda, Sabiheh Sadat Faghih, and Bahman Moghimi. Spatio-temporal modeling of yellow taxi demands in New York City using Generalized STAR models. (November), 2017.

Maria Scalabrin, Matteo Gadaleta, Riccardo Bonetto, and Michele Rossi. A Bayesian forecasting and anomaly detection framework for vehicular monitoring networks. In *IEEE International Workshop on Machine Learning for Signal Processing, MLSP*, 2017. ISBN 9781509063413. doi: 10.1109/MLSP.2017.8168151.

SFPark. Technical Manual, 2014.

Wei Shao, Yu Zhang, Bin Guo, Kai Qin, Jeffrey Chan, and Flora D. Salim. Parking Availability Prediction with Long Short Term Memory Model. In *Green, Pervasive, and Cloud Computing. GPC 2018. Lecture Notes in Computer Science, vol 11204.* Springer, Cham, 2018. ISBN 978-3-319-57185-0. doi: 10.1007/978-3-319-57186-7. URL `http://link.springer.com/10.1007/978-3-319-57186-7`.

Donald Shoup. The High Cost of Free Parking. 1997. URL `https://www.researchgate.net/publication/235359727www.tstc.org`.

Donald Shoup. Instead of Free Parking. 1999. URL `http://shoup.bol.ucla.edu/InsteadOfFreeParking.pdf`.

Donald Shoup. PERFORMANCE PARKING PRICES. 2011. URL `https://www.accessmagazine.org/wp-content/uploads/sites/7/2016/01/access38{_}free{_}parking{_}markets.pdf`.

Donald Shoup. The Right Price for Curb Parking. 2016. URL `https://www.delawarevalleysmartgrowth.org/wp-content/uploads/2016/05/The-Right-Price-for-Curb-Parking-April-21.pdf`.

Donald Shoup. *Parking and the City*, volume 29. Planners Press, Los Angeles, 1 edition, 2018. ISBN 9781138497030. doi: 10.1080/07293682.1991.9657508.

Donald C Shoup. Cruising for parking. *Transport Policy*, 13:479–486, 2006. doi: 10.1016/j.tranpol.2006.05.005. URL `http://shoup.bol.ucla.edu/Cruising.pdf`.

Robert H. Shumway and David S. Stoffer. *Time Series Analysis and Its Applications*. Springer Texts in Statistics. Springer International Publishing, Cham, 2017. ISBN 978-3-319-52451-1. doi: 10.1007/978-3-319-52452-8. URL `http://link.springer.com/10.1007/978-3-319-52452-8`.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ruslan Salakhutdinov, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. URL `http://jmlr.org/papers/v15/srivastava14a.html`.

Anthony Stathopoulos and Matthew G Karlaftis. A multivariate state space approach for urban traffic flow modeling and prediction. 2003. doi: 10.1016/S0968-090X(03)00004-4. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.2521{&}rep=rep1{&}type=pdfhttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.2521{%}7B{&}{%}7Drep=rep1{%}7B{&}{%}7Dtype=pdf`.

Shiliang Sun, Changshui Zhang, and Guoqiang Yu. A Bayesian network approach to traffic flow forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 2006. ISSN 15249050. doi: 10.1109/TITS.2006.869623.

Dusan Teodorovic and Panta Lucic. Intelligent parking systems. *European Journal of Operational Research*, 175(3):1666–1681, 2006. ISSN 03772217. doi: 10.1016/j.ejor.2005.02.033.

UK Government. Parking guidance and information, 2003. URL `http://www.dft.gov.uk/pgr/roads/tpm/tal/its/parkingguidanceandinformation`.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. pages 1–15, 2016. URL `http://arxiv.org/abs/1609.03499`.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Mascha Van Der Voort, Mark Dougherty, and Susan Watson. Combining Kohonen maps with ARIMA time series models to forecast traffic flow. *Transportation Research Part C: Emerging Technologies*, 4(5):307–318, 1996. ISSN 0968090X. doi: 10.1016/S0968-090X(97)82903-8.

J. W.C. van Lint, S. P. Hoogendoorn, and H. J. van Zuylen. Accurate freeway travel time prediction with state-space neural networks under missing data. *Transportation Research Part C: Emerging Technologies*, 13(5-6):347–369, 2005. ISSN 0968090X. doi: 10.1016/j.trc.2005.03.001.

V N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, 10(5):988–999, 1999. ISSN 1045-9227. doi: 10.1109/72.788640. URL `http://www.ncbi.nlm.nih.gov/pubmed/18252602`.

Ashish Vaswani, Google Brain, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. Technical report. URL `https://arxiv.org/pdf/1706.03762.pdf`.

Petar Veličković, Arantxa Casanova, Pietro Liò, Guillem Cucurull, Adriana Romero, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, 2018.

Eleni I Vlahogianni and Matthew G Karlaftis. Testing and Comparing Neural Network and Statistical Approaches for Predicting Transportation Time Series. *Transportation Research Record Journal of the Transportation Research Board Transportation Research Board of the National Academies*, (2399):9–22, 2013. doi: 10.3141/2399-02. URL `https://trrjournalonline.trb.org/doi/abs/10.3141/2399-02`.

Eleni I. Vlahogianni, John C. Golias, and Matthew G. Karlaftis. Short term traffic forecasting: Overview of objectives and methods. *Transport Reviews*, 24(5):533–557, sep 2004. ISSN 0144-1647. doi: 10.1080/0144164042000195072. URL `http://www.tandfonline.com/doi/abs/10.1080/0144164042000195072`.

Eleni I. Vlahogianni, Matthew G. Karlaftis, and John C. Golias. Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach. *Transportation Research Part C: Emerging Technologies*, 2005. ISSN 0968090X. doi: 10.1016/j.trc.2005.04.007.

Eleni I. Vlahogianni, Matthew G. Karlaftis, and John C. Golias. Statistical methods for detecting nonlinearity and non-stationarity in univariate short-term time-series of traffic volume. *Transportation Research Part C: Emerging Technologies*, 2006. ISSN 0968090X. doi: 10.1016/j.trc.2006.09.002.

Eleni I Vlahogianni, Matthew G Karlaftis, and John C Golias. Temporal Evolution of Short-Term Urban Traffic Flow: A Nonlinear Dynamics Approach. *Computer-Aided Civil and Infrastructure Engineering*, 23:536–548, 2008.

Eleni I. Vlahogianni, Matthew G. Karlaftis, and John C. Golias. Short-term traffic forecasting: Where we are and where we're going. *Transportation Research Part C: Emerging Technologies*, 2014. ISSN 0968090X. doi: 10.1016/j.trc.2014.01.005.

Eleni I. Vlahogianni, Konstantinos Kepaptsoglou, Vassileios Tsetsos, and Matthew G. Karlaftis. A Real-Time Parking Prediction System for Smart Cities. *Journal of Intelligent Transportation Systems*, 20(2):192–204, mar 2016. ISSN 1547-2450. doi: 10.1080/15472450.2015.1037955. URL https://www.tandfonline.com/doi/full/10.1080/15472450.2015.1037955.

Jin Wang and Qixin Shi. Short-term traffic speed forecasting hybrid model based on Chaos-Wavelet Analysis-Support Vector Machine theory. *Transportation Research Part C*, 27:219–232, 2013. doi: 10.1016/j.trc.2012.08.004. URL https://www.sciencedirect.com/science/article/pii/S0968090X1200109X.

Menglin Wang, Baisheng Lai, Zhongming Jin, Yufeng Lin, Xiaojin Gong, Jianqiang Huang, and Xiansheng Hua. Dynamic Spatio-temporal Graph-based CNNs for Traffic Prediction. Technical report, 2018. URL https://arxiv.org/pdf/1812.02019v3.pdf.

T. Warren Liao. Clustering of time series data - A survey. *Pattern Recognition*, 38(11):1857–1874, 2005. ISSN 00313203. doi: 10.1016/j.patcog.2005.01.025.

Rachel R. Weinberger and Robert Cornelius Hampshire. Parking-Cruising Caused Congestion. *SSRN Electronic Journal*, 2017. doi: 10.2139/ssrn.2906528.

Billy M. Williams and Lester A. Hoel. Modeling and Forecasting Vehicular Traffic Flow as a Seasonal ARIMA Process: Theoretical Basis and Empirical Results. *Journal of Transportation Engineering*, 129(6):664–672, nov 2003. ISSN 0733-947X. doi: 10.1061/(ASCE)0733-947X(2003)129:6(664). URL http://ascelibrary.org/doi/10.1061/{%}28ASCE{%}290733-947X{%}282003{%}29129{%}3A6{%}28664{%}29.

Eric Hsiao Kuang Wu, Jagruti Sahoo, Chi Yun Liu, Ming Hui Jin, and Shu Hui Lin. Agile urban parking recommendation service for intelligent vehicular guiding system. *IEEE Intelligent Transportation Systems Magazine*, 6(1):35–49, 2014. ISSN 19391390. doi: 10.1109/MITS.2013.2268549.

Shuguan Yang, Wei Ma, Xidong Pi, and Sean Qian. A deep learning approach to real-time parking occupancy prediction in spatio-termporal networks incorporating multiple spatio-temporal data sources. Technical report, Pittsburgh, 2019. URL https://arxiv.org/pdf/1901.06758.pdf.

Yanjie Duan, Yisheng Lv, and Fei-Yue Wang. Travel time prediction with LSTM neural network. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016. ISBN 978-1-5090-1889-5. doi: 10.1109/ITSC.2016.7795686.

Bing Yu, Mengzhang Li, Jiyong Zhang, and Zhanxing Zhu. 3D Graph Convolutional Networks with Temporal Graphs: A Spatial Information Free Framework For Traffic Forecasting. page 7, mar 2019a. URL http://arxiv.org/abs/1903.00919.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. ST-UNet: A Spatio-Temporal U-Network for Graph-structured Time Series Modeling. Technical report, 2019b. URL `https://arxiv.org/pdf/1903.05631v1.pdf`.

E.S. Yu and C.Y.R. Chen. Traffic prediction using neural networks. *Proceedings of GLOBECOM '93. IEEE Global Telecommunications Conference*, (January 1994):991–995, 1993. doi: 10.1109/GLOCOM.1993.318226. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=318226`.

Haiyang Yu, Zhihai Wu, Shuqin Wang, Yunpeng Wang, and Xiaolei Ma. Spatiotemporal Recurrent Convolutional Networks for Traffic Prediction in Transportation Networks. *Sensors*, 17(7):1501, jun 2017. ISSN 1424-8220. doi: 10.3390/s17071501. URL `http://www.mdpi.com/1424-8220/17/7/1501`.

J M Zamarreño and P Vega. State space neural network. Properties and application. *Neural Networks*, 11(6):1099–1112, 1998. ISSN 0893-6080. doi: http://dx.doi.org/10.1016/S0893-6080(98)00074-4. URL `http://www.sciencedirect.com/science/article/pii/S0893608098000744`.

Hongjun Zhang, Stephen G Ritchie, and Zhen-Ping Lo. Macroscopic Modeling of Freeway Traffic Using an Artificial Neural Network. 1997. URL `https://pdfs.semanticscholar.org/2374/8b6173dbcb6cc724371d06417f7ce893466e.pdf`.

Yanxu Zheng, Sutharshan Rajasegarar, Christopher Leckie, and Marimuthu Palaniswami. Smart car parking: Temporal clustering and anomaly detection in urban car parking. In *IEEE ISSNIP 2014 - 2014 IEEE 9th International Conference on Intelligent Sensors, Sensor Networks and Information Processing, Conference Proceedings*, 2014. ISBN 9781479928439. doi: 10.1109/ISSNIP.2014.6827618.

Yanxu Zheng, Sutharshan Rajasegarar, and Christopher Leckie. Parking availability prediction for sensor-enabled car parks in smart cities. In *2015 IEEE 10th International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP 2015*, 2015. ISBN 9781479980550. doi: 10.1109/ISSNIP.2015.7106902.

Barret Zoph, Quoc V Le Google Brain, and Quoc V Le. Neural Architecture Search with Reinforcement Learning. *CoRR*, abs/1611.0, 2016. URL `http://arxiv.org/abs/1611.01578`.