Predicting airline passengers with deep multi-task learning

Schoonderbeek, J.

Award date:
2020

Technische Universiteit Eindhoven University of Technology

Department of Industrial Engineering and Innovation Sciences
Information Systems Research Group

Predicting Airline Passengers with Deep Multi-Task Learning

J. Schoonderbeek (1246682)

In partial fulfillment of the requirements for the degree of Master of Science in Operations Management and Logistics

Supervisors:
dr. Y. (Yingqian) Zhang - Eindhoven University of Technology
dr. K. (Kalliopi) Zervanou - Eindhoven University of Technology
MSc. Tom de Ruijter - KLM Royal Dutch Airlines/ BigData Republic

1.0

Eindhoven, Tuesday 31st March, 2020

Abstract

Airline passenger forecasting has been studied for many years and has served airlines with critical information for their planning processes. This thesis distinguish six different type of passengers: economy and business class travellers, departure and arrival transfer passengers and local departure and arrival passengers. These different type of passengers follow different type of processes and give therefore more information during the planning phase. Furthermore, these type of passengers share the same data structure and the same features, therefore multi-task learning can be an interesting approach. With multi-task learning the model leverages shared learning among the different type of passengers which in this case improved the performance by 7.4% in comparison with independent single task models. In addition, the proposed model has outperformed all benchmark model studied in this thesis. Therefore, it can be concluded that this novel approach is very promising for industrialization and future research in the field of airline passenger forecasting and ML.

Keywords - Airline Passenger Forecasting, Machine Learning, Deep Multi-Task Learning

Preface

This thesis is a result of a master project for the program Operations Management and Logistics within the Industrial Engineering department at Eindhoven University of Technology. The project has been conducted at KLM Royal Dutch Airlines.

Throughout the program I had support of some people and I would like to use this preface to thank those that particularly helped me and had a contribution to this thesis.

First of all, I would like to thank dr. Y. (Yingqian) Zhang for her guidance throughout my master program. She served as my mentor and was involved in making important decision throughout the program. Furthermore, she supervised my master thesis and was very helpful during certain phases of the project. Thank you for your time, effort and feedback to help me improve myself. Second, I would like to thank dr. K. (Kalliopi) Zervanou for being my second supervisor and taken the time to provide me from feedback on my master thesis as well.

Furthermore, I would like to thank Guido Santegoeds of KLM for giving me the opportunity at KLM. Without Guido this collaboration would probably not have taken place. In particular I want to speak out my appreciation to Tom de Ruijter, who was my supervisor at KLM. I learned a great deal of you and you amazed me many times. Thank you for sharing your experience and coaching my throughout the process. Conducting my first ML project within a company under your supervision made me a very lucky person.

Last, I would like to thank my parents and girlfriend Julia. Moving from Amstelveen to Eindhoven was a big step but you always supported and helped me where you could. In the end, this choice was absolutely the best one in terms of my education.

Jeroen Schoonderbeek

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Airline passenger forecasting has been a popular research topic over the years. This chapter describes how this has become the motivation for this master thesis. In addition it will outline some basic information regarding the context of the study and specify the scope of the research. Subsequently, the corresponding research questions and thereby structure of the thesis will be explained and elaborated. At the end the scientific contribution of this thesis will be highlighted and discussed.

1.1 Research Motivation

This study is conducted in collaboration with KLM Royal Dutch Airlines. KLM is the oldest operating airline flying under the name and was established in 1919. About 100 years later it transports more than 3,000,000 passengers a year and has over 33,000 employees.

Over the years, the aviation sector has become a very competitive market to operate in. Nowadays, customers have a lot more choices between different airlines. Besides, there are more different types of airlines in terms of service and prices they offer. Furthermore, the available flight space has become more occupied and airports have become restricted in their capacity expansion, this also limits the growth perspective of an airline. Taken this all into account, it has become critical for an airline to excel in cost control and customer satisfaction in order to stay competitive (KLM, 2018).

In order to minimize cost and maximize customer satisfaction, operating strategies of an airline are crucial. Determining this operating strategies require a lot of planning and planning is highly complex due to many dependencies and unknowns. The planning process of an airline and specifically the one of KLM, can be outlined as followed:

Figure 1.1: Airline Planning Process

In a nutshell, an airline starts with a long-term strategy. In this phase the direction of the company will be established. Normally, this covers a time horizon between 10, 5 or 3 years. Shortly after this phase, the network planning follows. Network planning is strongly influenced by the company's strategic goals. Based on the strategic direction an airline has set, network planning decisions are made about fleet capacity, growth expectations and flight destinations. However, one has to check for operational feasibility in terms of infrastructure, manpower and materials. If

the planned network schedule is feasible, the airline will start calculating prices for the products and services they are planning to offer. This products and services are then made available on the desired channels within the Distribution & Sales step. Thereafter, Revenue Management is trying to optimize revenue by getting the airplanes as full as possible while selling the tickets for the highest possible price. Lastly, within the Operation & Service, planning is done in terms of resource planning, infrastructure and safety.

This planning process can be significantly improved by leveraging data such as using optimization techniques, descriptive analytics, apps and machine learning (ML). This study will focus on the ML aspect. One way of applying ML is to predict what could happen in future such that better planning can be done. A factor that plays a crucial role in planning is knowing how many passengers to expect. To require this information airlines use a passenger forecast.

It is important to make a clear distinction between two types of passenger forecasts. A passenger forecast can be made to predict the expected demand, hence how many people are interested to fly or a passenger forecast can be made to predict the number of passengers that will board the airplane of an upcoming flight. Predicting the expected demand is for this thesis out of scope and will not be further discussed. This thesis will focus on the number of passengers that will actually board the airplane of an upcoming flight. This means that cancellations and no-shows are excluded because these type will not board the airplane.

The passenger forecast that predicts how many passengers will board the airplane has many benefits for an airline. For example, the fleets within the airline capacity can be allocated such that as many passengers as possible can fly. Because by knowing how many passengers to expect, one can choose the airplane configuration that suits best. This may benefit an airline in unnecessary allocating an airplane that is too small, which means less tickets can be sold or it could be that too much aircraft capacity is allocated which lead to economical inefficiencies. Another example is that an airline must perform feasibility checks. When an airline produces a new network schedule it has to be checked if there is sufficient staffing, if safety regulations are met and if there is enough infrastructure in place such as check-in desks, baggage conveyors etc. To evaluate this, one must know the expected number of passengers per time unit and the capacity. Besides this feasibility check, an airline can also plan resources based on this information. For example, for the check-in process of the departure hall. If one uses the expected number of passengers for a departing flight in combination with an arrival profile, one knows how many passengers to expect per time unit. By knowing how long each process activity takes one can determine how many passenger to expect per time unit per process activity. With this information waiting times can be calculated, hence costs be controlled and customer satisfaction can be maximized.

However, looking back at the planning process of figure 1.1, each process step has different data available and requires different outputs of the passenger forecast. Therefore different type of passenger forecast models are required. Based on KLM's request, this study will focus on predicting passengers per flight between 125-360 days prior to departure. This time horizon is relevant for two phases of the planning process. For 'Network' to improve the aircraft allocation process and execute feasibility checks. For 'Operation & Service' to improve their resource planning of the processes.

In terms of outputs, Network wants a passenger forecast segmented in cabin class travellers. In case of KLM, this is economy and business class. With this information, network can choose the optimal fleet configuration per cabin class. Operating & Service wants a passenger forecast segmented in cabin class, local and transfer passengers. A transfer passenger is a passenger type that will use the airport as a hub to fly to the next destination. In contrary, a local passenger is one that actually arrives or departs from the concerned airport. This different type of passengers follow different processes and therefore it is relevant for resource planning. For the feasibility check all six types of passengers are required because different type of passengers follow different type of processes.

In the current situation at KLM, there is already a passenger forecast system in place. However, due to complaints of its performance there is a clear motivation to investigate a new type of

passenger forecast with the objective to outperform the current system. The current system has a horizon of 180 days prior to the departure, where the planning process needs a horizon of 360 days prior to departure. Besides, planners experience that the error margin of the prediction model is too large and therefore, they do not trust it anymore. For all the above mentioned reasons, a mandate has been given in the form of a master thesis, to research a better working prototype as a proof of concept.

1.2 Research Questions

In this section the research motivation will be addressed in a structured fashion by means of research questions and its scientific contribution. First the main research question will be introduced, where after, the sub-research questions will follow. Lastly an outline will be given how this master thesis adds value to the current science in the field of ML, the airline sector and the field of operations & logistics.

Main research question:

How to build a forecasting model based on historical airline data to predict the number of passengers on flights departing from a specific airport in specific flight categories? Specifically, local and transfer passengers as well as different cabin classes with a time horizon of 125-361 days prior to departure.

In order to answer the main research questions, it is split up in 5 sub-questions.

Sub-research questions:

1. *What methods do already exist that concerns passenger forecasting on flight level?*
2. *Which features can be selected and, if required, be constructed or transformed, to build the prediction model?*
3. *What baseline can be used to compare the model performance with?*
4. *Which machine learning techniques should be used to build a prediction model that produces the best results?*
5. *How can the prediction model be evaluated and tested?*

1.3 Scientific Contribution

In existing literature on airline passenger forecasting mainly two approaches are taken. A econometric time series approach or an ML approach. This research extents the ML studies. It is common to build a ML model per output you want to predict. For this case we want to predict multiple splits in outputs at the same time, hence multiple type of passengers at the same time. These type of passengers share the same data structure and the same features, therefore multi-task learning (MTL) may suits this problem. With MTL, the model leverages shared learning among the different type of passengers which can improve the performance. For this thesis project, MTL is applied with Deep Neural Networks (DNN). To the best of my knowledge this is first time MTL with DNN is applied in predicting the number of passengers that will board an airplane for a specific upcoming flight. This could lead to new perspectives on the study of passenger forecasting. Additionally, it enriches the work on the application field of MTL. Furthermore, ML explainability theories such as SHAP and partial dependency plots are used to provide insights in differences between single-task learning (STL) and MTL. Moreover, the problem of passenger forecasting is extended to a more detailed level, hence, instead of predicting the total number

of passengers per flight, different types of passengers will be predicted. This supports the businesses side to further improve planning and optimization be knowing more specifically when to expect which type of passenger. Lastly, this is the first time an official research is conducted in the Netherlands concerning airline passenger forecasting because there are no Netherlands based publications available.

Chapter 2

Literature Review

Nowadays, big data is almost everywhere. The airline sector and in particular passenger forecasting benefits from this. How this has evolved and influenced the industry will be discussed in this chapter. Subsequently, the existing methods on passengers forecasting will be outlined in order to prove that MTL is never used before on this topic. In the end, MTL will be further explained in theory and with examples.

2.1 Machine Learning in the Airline Industry

In 2017, a survey was held within the airline industry and reported that half of the global airlines plan to make significant investments in ML and AI capabilities (Baker, 2018). In essence, ML is a set of statistical models that find patterns of predictability (Fedyk, 2016). Most of the time large amounts of data benefits ML in terms of performance. If large amounts of data is improves ML than airlines such as KLM are an excellent environment for it. KLM has over 3,000,000 passengers a year, around 33,000 employees and about 330 flights per day; as one can imagine this produces a lot of data.

In general, airlines generate huge amounts of data. However, this data is not always used as information. Airlines such as Delta Airlines highlighted in their 2017 investor day presentation, that they are going to expand ML and AI capabilities to better utilize these huge amount of data such that it can be used as information for decision making (Delta, 2017). Or Lufthansa Group who recently invested in an alliance with Hopper to expand their ML capabilities (Lufthansa, 2019). Besides the huge amount of data and investments that are made, there is a very wide range of possibilities how ML can be applied within airlines.

Supervised learning is by far the most popular approach in the airline industry. For example, a recent award winning paper (Leeuwen et al., 2020) applied Gradient Boosting Decision Trees to predict mishandled baggage while transferring to a connecting flight. In addition, they used process analysis to better understand the business domain, which resulted in discovering new features regarding process times. Furthermore, it was proven that ML showed better results than business rules in terms of accuracy coupled with a marked increase in precision and recall.

A less popular method in the airline sector is unsupervised learning. Most of the time it serves a supervised problem. An example, are the Dutch researchers (Wotawa et al., 2019), who applied unsupervised learning techniques to detect fraudulent bookings of online travel agencies. They argue that their anomaly detection can be a valuable tool to identify false bookings which hurts tickets sells.

Recently, reinforcement learning (RL) has become a popular ML technique applied for airline decision making processes. Most papers consider airline revenue management use cases. For

example, (Lawhead. R.J, 2019) used an airline revenue management case to apply RL. It shows that especially, when the state space is large, the algorithm delivers encouraging computational behaviour. In addition it outperforms a well-known industrial benchmark heuristic. Or (Shihab et al., 2019), who used RL to find the optimal policy for Seat Inventory Control and Overbooking in order to maximize revenue for each flight. It was found that the proposed policy came very close to the theoretical optimal solution. This promising results has triggered airlines to further study this type of techniques.

To conclude, ML is not new in the airline business. In contrary, there are a lot of different use cases to apply all different sort of ML techniques. This research will focus on predicting the number of passengers that will board a specific upcoming flight. This is a typical supervised learning problem, to be more specific, a regression problem. In the next section a review will be given about the published literature on this problem.

2.2 Existing Methods for Airline Passenger Forecasting

2.2.1 Time Series Models

For forecasting airline passengers a popular choice is the Seasonal Autoregressive Integrated Moving Average (SARIMA) model. This method is a combination of two models, the Moving Average (MA) and Autoregressive (AR). These time series models are in existence since the early 1900's. The MA gives a prediction based on the average of a certain consecutive subsets of time series. The AR is based on stochastic calculations in which prediction are based on weighted sum of past values. The extra value of combining those models is that they capture non-stationary time series data in comparison to MA and AR. (Box and Jenkins, 1970) introduced Seasonal ARIMA (SARIMA), which therefore made it relevant for forecasting airline passengers. Based on a literature review on econometric models applied in airline passenger forecasting (Fildes et al., 2011), it appears that the SARIMA model and its derivatives are most frequently applied and show reliable results.

For example, (Andreoni and Postorino, 2006) experimented with uni-variate and multivariate time series models to estimate passengers demand in the South of Italy on yearly basis. In addition, an interesting explanatory variable is introduced, the researcher applied hedonic pricing theory to determine the fare price such that it can be used as variable. In conclusion, the SARIMA model with an explanatory variable (SARIMAX) showed the best performance in comparison with the other time series models.

Another example is (Tsui et al., 2014), who used SARIMA and ARIMA with explanatory variables (ARIMAX) to forecast passengers throughput at the airport of Hong Kong on monthly level. They were the first to apply the Box-Jenkins ARIMA methodology for the region Hong Kong. Different time series models are applied and again is this study the SARIMA and SARIMAX outperformed the other time series models.

An overlapping assumption of the time series models discussed so far, is that they assume a linear dependency between the prediction and the previous data points. There are variants that try to capture non-linear dependency such as Autoregressive Conditional Heteroskedasticity (ARCH) by (Engle, 1982) or Generalised Autoregressive Conditional Heteroskedasticity (GARCH) by (Bollerslev, 1986). However, these models are meant to model volatility of shocks of financial time series and are therefore not relevant for passenger forecasting. In contrast, machine learning models have shown relative good performance in capturing non-linear dependencies. Subsequently, the relevant machine learning algorithms for this topic will be discussed in the next section.

2.2.2 Machine Learning Models

In the literature mostly Artificial Neural Networks (ANN) are applied to forecast airline passengers. In some cases Support Vector Machine (SVM) are also applied but mostly in combination with an

additional type of model. Furthermore, there are some interesting cases where tree-based models where applied.

One of the first to apply ANN for forecasting airline passengers was (BaFail, 2004), who applied ANN to forecast the number of airline passengers in Saudi Arabia. The forecast was split up in two outputs predicting international and domestics passengers. For each output a separate independent neural network was applied. They found that the most influential factors where oil gross domestic product and capital income in the domestic and international sector.

(Mohie El-Din et al., 2017), applied back propagation NN to forecast the airline passenger in the region of Egypt. The contribution of this research paper was to use genetic algorithms (GA) to enhance the ANN. The GA adapts the weights in the model which is proven to show better performance in this case. Features that were used are: population size, employed population, per capita income(PCI), Gross domestic product (GDP), gross national product(GNP), economic growth rate an foreign exchange rate.

(Mostafaeipour et al., 2018) also improved the ANN with another algorithm. In this research the ANN was optimized with the Bat and Firefly algorithm. In this case study the air travel demand for Iran was predicted by considering elasticity and population size in each zone of the country. By means of the optimization algorithm the results show an improvement of the adaptation rate of the ANN.

(Laik et al., 2014), took a different approach by predicting the load factor of flights by means of decision trees. This research was conducted in the area of Singapore. Features that were used are: aircraft type, airline type, day of the week, month of the year, hour of the day, country, destination. The aim was mainly to predict the demand for the day in order to improve resource planning.

In the paper of (Godfrey and Gashler, 2018) a new model is introduced. They present a neural network technique for the analysis and extrapolation of time series data. This technique is defined as Neural Decomposition. In general, layers with a sinusoidal activation function serve to decompose the training data into a sum of sinusoids. In addition, layers with non-periodic activation functions are applied to capture non-periodic components such as trends etc. This model was tested on multiple datasets among which monthly international airline passengers. The results show that a simple model generalizes well on different time series. Each time it outperforms the popular models such as LSTM, echo state networks, ARIMA, SARIMA, SVR with a radial basis function, and Gashler and Ashmore's model.

2.2.3 Conclusions

This work looks at the problem of passenger forecasting, specifically for airlines. Existing and previous approaches have mostly focused on modelling parts of this problem, such as (Tsui et al., 2014) with a classical time series approach for total number of passengers on specific time horizons, (Mohie El-Din et al., 2017) with ANN for predicting on flight level or (Laik et al., 2014) who have chosen the approach of modelling passengers based on load factors.

These approaches don't scale to the operational requirements of KLM, where e.g. time series modelling would involve 1200 separate models for the different flights and dimensions required. This adds complexity to deployment, maintenance and retraining. Even regression models per flight would still require training different models for each passenger type – e.g. business, economy etc, resulting in a separate model for each passenger type.

Instead, this work approaches the issue through joint modelling of the different passenger groups per flight simultaneously using deep learning multi-task learning resulting in a single model forecasting all passenger types for the entire network and fleet.

Recent work by (Nekrasov et al., 2019) has shown this approach to work well in the computer vision domain. (Gao et al., 2019) has translated this approach to the regression domain with structured data, yielding better performance using a single model rather than modelling the sub-problems separately.

As is with most current deep learning approaches, practice precedes theory. There is no strong theoretical framework yet as to why this works well, however (Ruder, 2017) provides several

compelling intuitive arguments. In the next section, this approach will be explained in more detail.

2.3 Multi-Task Learning

2.3.1 Introduction to Multi-Task Learning

Typically, with ML the aim is optimize a certain metric. This could be a business Key Performance Indicator (KPI) or a score on a benchmark. In case of this research, the aim is optimize a regression performance measure per passenger type such as Mean Absolute Error (MAE) or Mean Squared Error (MSE). Commonly, a single model per output is trained to perform this task. Subsequently, the model will be optimized by tuning parameters until its performance does not further improves. In most cases this process performs well, but it could be that by being single-focused on one output, relevant information of other outputs might be ignored. (Ruder, 2017) argues that by sharing representations between multiple related outputs, the model will generalize better and therefore, in cases, performs better than the original single output model. This approach is defined as MTL.

MTL is part of transfer learning. According to (Sinno J., 2010), it is defined as inductive transfer learning which aims at finding good feature representation to minimize domain divergence and classification or regression model error. To formalize the definition of MTL, I would like to refer to (Zhang and Yang, 2017):

"Given m learning tasks $\tau_i, i = 1, \ldots, m$, where all the tasks or a subset of them are related, multi-task learning aims to help improve the learning of a model for task τ_i by using the knowledge contained in the m tasks.

MTL has been mostly successfully applied in Natural Language Processing (NLP), speech recognition, computer vision and drug discovery. This thesis however, argues that it can also be successful for multi-target regression problems with structured data.

2.3.2 Why Multi-task learning can work?

There are multiple reasons why MTL might work. Looking from a biological cognitive perspective, it is in some ways similar as how humans learn. Often, if we learn a new task, knowledge of related acquired tasks are applied. For example, someone who plays field-hockey and tries to learn golf, will probably learn quicker than someone who has never exercised a sport with a stick.

From a technical perspective, MTL is a form of inductive transfer learning. Inductive transfer may benefits the model performance by introducing an inductive bias. Consequently, the model will prefer some hypotheses over others. For example, Ridge regularization results in inductive bias, where the preference is for sparse solutions (Ruder, 2017). In terms of MTL, the inductive bias is created by auxiliary outputs. This might result in that the model will prefer hypotheses that explain multiple outputs instead of one output and therefore will generalize better.

Although some intuitive reasons are already mentioned such as biological cognitive perspective and inductive bias, there are some underlying mechanism that were researched by (Caruana, 1997). Caruana is one the first to introduce MTL and can be seen as the pioneer in the field. In the paper four different mechanism are mentioned why MTL might work. Furthermore it is assumed that there are two related tasks A and B, which both have a common hidden layer denoted as F.

First, a MTL approach increases the training data. Each output has probably some meaningless data in it, which is defined as noise. If a model is trained on output A, the objective is to learn an optimal representation F, that ignores noise and generalizes well. Assuming that different outputs have different noise patterns, a MTL model learns two outputs simultaneously and might therefore learn a more general representation. Only learning output A increases the risk of overfitting output A. Jointly, learning output A and B enables a better generalization by averaging the noise patterns.

Second, if the training data contain a lot of noise, is limited or high-dimensional, it might be harder for a model to learn which features are relevant. By MTL additional evidence may be gathered about which features are relevant due to related outputs.

Third, it might be that feature G is easier to for output A than output B. The reason for this could be that this feature G interacts in a more complex way or other features impeding the model to learn feature G. With MTL it can be possible for the model to eavesdrop, in other words, to learn feature G through output A for output B.

Fourth, MTL has a regularization function by introducing inductive bias to the model. Consequently, overfitting is reduced as well as the Rademacher complexity of the model, i.e. the models ability to fit random noise.

Furthermore, MTL could also be interesting for practical reasons. For instance, it is easier to deploy and industrialize one model instead of multiple models. After deployment, less time is required to keep up the maintenance and monitoring the model. And lastly, tuning the hyper-parameters has to be done only once instead of doing this per model.

However, there are also reasons why MTL may not be favorable. It requires more effort in the development phase. Building the model will become more complex, training time will increase and understanding what is happening might be more difficult. In addition, it requires more hardware capacity and it is more difficult to add new outputs to the model instead of building a new extra model. Because by adding new tasks to the model, one has to reconsider the loss function, what parameters to share with the new task and how it affects the other tasks.

In conclusion, the benefits of MTL sounds promising and interesting to test. Therefore, predicting different types of passenger can be an interesting use case for the application field of MTL. Next, a more in-depth analysis will be given on MTL and how it works. This will be done in the context of Deep Learning.

2.3.3 Methods for MTL in Deep Learning

MTL with Deep Learning is the most common application in the field. It is typically done with hard or soft parameter sharing between the hidden layers. Both will be explained separately.

Hard Parameter Sharing

Hard parameter sharing was first introduced in (Caruana, 1993). This approach is most common and intuitive to apply. It basically, shares the hidden layers between all the outputs and has a separate output layer per output, figure 2.1.

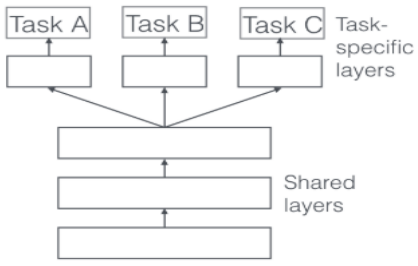


Figure 2.1: Hard parameter sharing for multi-task learning in deep neural networks

By sharing the layers, the chance of overfitting will be significantly reduced (Yang and Hospedales, 2017). In this paper, it is argued that the risk of overfitting is an order N smaller than overfitting a model per output. In this case, N is denoted as the number of outputs. The reason

for this might be that the model needs to represent all outputs and therefore has to capture the more general relationships instead of adding output specific relationships.

Furthermore, hard parameter sharing requires specifying the weights given to the task's loss. In this way one, can for example give more priority to a specific task. However, if the weight becomes too high, it could be that the other task's losses starve and therefore will not perform well. Choosing these extra parameters can be difficult and equalling the weights can be a good default setting. Finding the right weights is also very time consuming, therefore, for this thesis a default with equal weights will be chosen initially.

A good example of hard parameter sharing is the paper (Kokkinos, 2017). In this paper seven computer vision problem are solved with hard parameter sharing. The authors showed that the computational cost decreases but experience a degradation in performance by adding more tasks to the model. Other examples of hard parameter sharing are (Nekrasov et al., 2019), (Dvornik et al., 2017), (Kendall et al., 2018), (Bilen and Vedaldi, 2016), (Pentina and Lampert, 2017).

Soft Parameter Sharing

With soft parameter sharing, each output has its own hidden layers and parameters, figure 2.2. The distance between the parameters is then regularized such that they become more similar. This could be done with for example Lasso regularization (Duong et al., 2015).

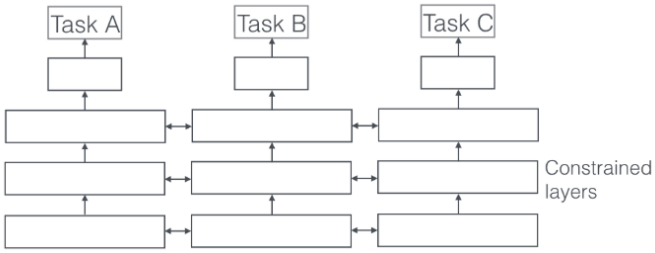


Figure 2.2: Soft parameter sharing for multi-task learning in deep neural networks

This approach has shown to be more robust towards learning output specific features, which make sense because each output has its own model. However, it is less popular in the existing literature. Examples of soft parameter sharing are (Dai et al., 2016) and (Tessler et al., 2017).

Auxiliary tasks

A classic way to improve MTL performance is by adding a related task to the model as an auxiliary output. For example, (Caruana, 1997) tries to predict the steering direction of a self-driving car and adds as an auxiliary task different characteristics of the road. In (Song and Xiao, 2014) landmark detection was predicted with head pose estimation and facial attribute inference as auxiliary tasks. The authors of (Girshick, 2015) build a prediction model where the class and the coordinates of an object in an image are learned. (Liu et al., 2015), jointly used query classification with web search to improve performance. And lastly, (Arik et al., 2017), combined the phoneme duration and frequency profile for text-to-speech. However, it still is relative unclear what auxiliary task may be beneficial in practise. For now, finding an auxiliary task is based on the assumption that it is in some way related to the main task. In addition, there is no standard notion defining when two task are related to each other. (Caruana, 1997) defines two task as related when they use the same features for decision making. (Baxter, 2000) states theoretically if two task share a common optimal hypothesis class, i.e. share the same inductive bias, they are related. In the paper of (Xue, 2007), it is argued that two task are related if their parameter vectors in the classification boundaries are close to each other. Although there is some progress in understanding task relatedness, there is not much advances in recent literature. In conclusion, it can be stated that task relatedness is not binary. In general, similar task should improve the

learning progress of the model. Where in the contrary less similar task should make learning more difficult. (Ruder, 2017) states that in case of allowing the models to learn what to share might temporarily circumvent the lack of theory and make models better even tough the auxiliary task is loosely related. In conclusion, there should be a clear notation for task relatedness in MTL such that we know what tasks are related with each other.

2.4 Position of this Research in the Literature

In this chapter literature is discussed on three subjects: the increasing trend of ML in the aviation industry, the existing methods on airline passenger forecasting and the theory around MTL. From the literature study it can be concluded that MTL is a new and exciting method to forecast airline passengers. Although many methods are tried to forecast airline passengers, MTL is a completely new approach and seems a suitable enhancement on the current literature. The different type of passengers share the same features and it is assumed they are somehow related. According to the theory this could improve the performance of the prediction model and is therefore interesting to test. Furthermore, this thesis introduces an extension of the problem by predicting multiple types of airlines passengers in order to better plan for airline processes. With this thesis a Deep MTL model will be introduced that is optimized with limited time and resources. Hopefully, this will lead to new perspectives on the study of passenger forecasting and enriches the work on the application field of MTL.

Chapter 3

Methodology

The methodology for building a passengers forecast applied in this thesis is two-folded, where both are focused on a proof of concept approach. First, the abstract methodology of the thesis will be discussed. Second, a more in-depth methodology for the specific use case of the thesis will elaborated upon.

3.1 Abstract Methodology

The central theme of the approach is the process of demonstrating concepts in MTL by making use of DNN. These techniques are divided in different model architectures by experimenting with different paradigms such as, single task models, auxiliary inputs and grouping tasks. The final proposed model can be adopted by any organization or person who wants to apply MTL on a multi-target regression problem using structured data. Therefore, the purpose of this section is to extract an abstraction of the methods used in this thesis. This is based on the design science principle of (Van Aken, 2005), such that the abstract knowledge can be used by professionals for similar problems they are trying to solve. In (Peffers, 2007), a design science research methodology (DRSM) is introduced for specific information system problems. This methodology is adhered to in this thesis and comprises the following process steps: problem identification and motivation, definition of the objectives of a solution, design and development, demonstration, evaluation, and communication.

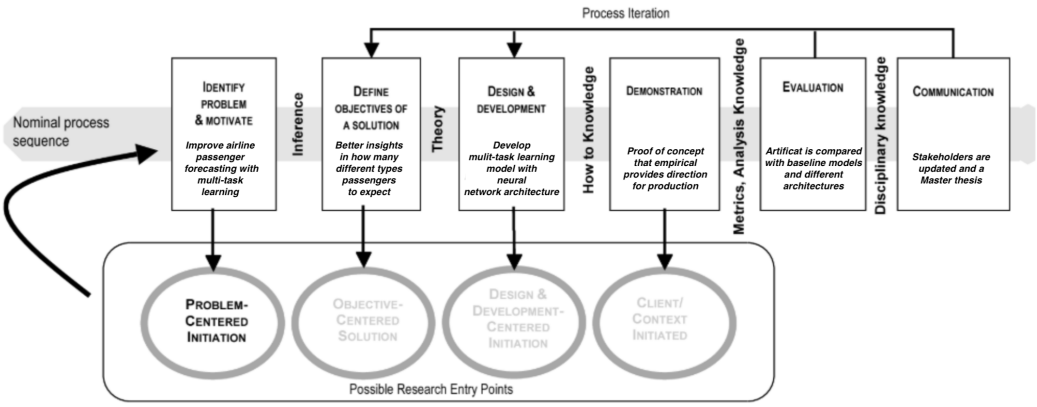


Figure 3.1: DRSM for airline passenger forecasting at KLM

Figure 3.1 provides a summary of the DRSM process model pertained to this thesis. The underlying problem that motivated this research, is to improve the current airline passenger prediction model of KLM. The need for better insights about different types of passengers triggered the development of a MTL neural network model.

1. **Identify problem & motivate:** The airline industry is highly complex with many dependencies within the operation. Therefore, efficient and effective planning of the operations is crucial for airlines to stay competitive. Passenger forecast can serve as important information in making decisions within the planning process. Forecasting too many passengers can lead to oversupply of services and products. Whilst forecasting too little passengers may lead to decreasing customer satisfaction and operational challenges. Because different types of passengers follow different operational processes, this thesis aims to predict the expected number of different types of passengers. One might expect that these passengers are somehow related, which makes it interesting to apply MTL for this problem.

2. **Define objectives of solution:** The objective for this research is to improve the current passenger forecast of KLM and to investigate if MTL might be a better solution compared to building a single-task model per required output. These required outputs should be available between 360-125 days prior to departure for any specific upcoming flight.

3. **Design and development:** The artifact of this thesis is a MTL model that can handle categorical and numerical features to predict for the required outputs. In addition, these inputs are transformed in a preprocessing pipeline to improve the prediction models performance. For the proposed model different types of architectures are experimented with which are inspired on literature findings. However, these findings were not directly applicable for this problem. Therefore, own applications of network architectures have been developed and tested.

4. **Demonstration:** A final proof of concept has been developed and proved to perform better than the current system of KLM. Next to this, it also proved that MTL can work better than building single task models.

5. **Evaluation:** The proposed model has been compared with baseline models and different architectures. The baseline models consist out of a simple prediction model, the current system used by KLM and a Gradient Boosting Decision Tree model, which is considered as a best in class model. The proposed model has shown to outperform the benchmarks model with 60%, 20% and 2%, respectively. Hence, artifact leads to improved performance and robust results.

6. **Communication:** The approach, experiments and results have been shared with the most important stakeholders. This was mainly done by presentations, providing slide decks and frequent meetings. Furthermore, a final detailed report is delivered in the form of a master thesis with a corresponding repository. This repository contains all the code necessary to reproduce the experiments and results.

3.2 Proof of Concept Approach

In this section a more in-depth methodology will be discussed which was used for understanding and experimenting the problem. The set-up was inspired by the CRISP-DM process model as shown in figure 3.2.

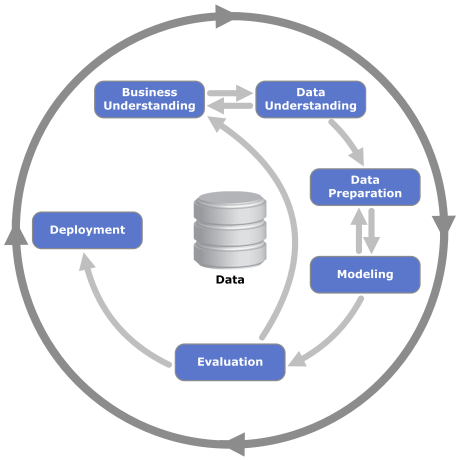


Figure 3.2: CRISP-DM Process Diagram for Data Mining

In essence, this thesis follows the same steps of the CRISP-DM process: business understanding, data understanding, data preparation, modelling, evaluation and deployment. However, because this thesis follows a proof of concept approach, deployment is out of scope. The first three steps will be extensively discussed in chapter 4, 5 and 6 respectively. In addition, the evaluation stage will be discussed in chapter 9 and provides a comprehensive overview of the results. However, one of the main focus points of this thesis is to develop a new architecture for predicting different type of airline passengers. Therefore, the remaining of this section will be dedicated to explain how the experimentation methodology for modelling is set up.

The experimentation methodology is based on the lean startup method. The general idea is to start simple and small. Then add complexity and test for improvement. In our case, the first step was to start with a simple baseline model. This was relatively easily to build and explain to stakeholders. It gives an understanding if the right predictions are made, how difficult the problem is and if the data is clean. For this relative simple model, a moving average (MA) per flight was used. This MA uses the three previous flights taken the query moment into account. For example, if we want to predict a specific upcoming flight with flight number KL 0001 and a query moment of 150 days before departure. Then we look 150 days back and take the average of the last 3 flights with flight number KL 0001. Second a more advanced approach is considered. For the second baseline model, a state of the art model has been studied, which is GBDT. This is then considered as best in class and the model to beat. Together with the current passenger forecast model of KLM, these 3 benchmark models are considered as the baseline models. They can be compared with the proposed models in order to evaluate the performance.

For the neural network it becomes more complex as there are a lot of parameters and options to explore. First of all, the sequence layers are out of scope for this research. Although it may improve performance, it is considered to broaden the scope of this research too much and therefore it might loose focus. Secondly, hyperparameters such as learning rate, batch size, drop out rate, regularization rates, epochs, activation functions and weight initialization are tested, but will not be extensively discussed. The reason is to keep focus on what is important for this thesis, namely the comparison of MTL with single task models.

In terms of MTL, first single neural networks with identical architectures per task will be build in order to compare with the MTL model. Second, a MTL with hard parameters sharing will be modelled where different sharing of parameters are tested, this will be done by grouping outputs

layers. This term 'groups' will be explained later on in chapter 8. In addition, auxiliary tasks will be added to the network to see how it affects the performance.

As evaluation, the models will be compared with a 2-way holdout method (train/test split), where the most recent year will be the test set. The data used for this thesis is over 140,000,000 records and is therefore considered as a large dataset. According to (Raschka, 2018), when a having a large dataset, a 2-way holdout method should be sufficient for performance estimations and evaluation of models. The evaluation is based on performance over the months of the year and day of the week. Furthermore, the query moment will be taken into account and residuals plots will be examined. But first, it is necessary to understand the problem and business challenges more in-depth. Therefore, the next chapter will zoom in on business understanding.

Chapter 4

Business Understanding

This section will provide a comprehensive overview about how the proposed model can support the business in several areas and in various planning and scheduling stages. Different use cases for passengers forecasting will be discussed and an overview will be given how the system is designed and fits into the business.

4.1 Business Use Cases

There are multiple instants before departure, where a passenger forecast can serve as an important information source for decision making. The following enumeration will outline when those moments occur in time before departure and where the passenger forecast numbers are used for.

- **10-3 year:** The strategy of the airline is determined in terms of fleet capacity and growth planning of KLM and Schiphol. In this case, long term growth estimates of the total number of expected passengers, is used to decide what direction to take with respect to the current capacity.
- **3-1 year:** Within this time horizon the network planning is scheduled. For this use case, the passenger demand is required for decisions about which destinations are economically attractive and which destinations might be less attractive.
- **365-1 days:** The airline is trying to decide on the optimal fleet allocation. This in combination with other factors depends strongly on how many passengers to expect for a specific upcoming flight.
- **240-125 days:** The first resource plannings are made and feasibility checks are performed to understand if the current capacity can cope with the expected upcoming demand. Furthermore, an airline has to check for customs & security compliance, for if so, do certain measures need to be taken in advance to guarantee safety. Amongst others, the expected number of passengers, serves as crucial information to estimate the total demand for the operation processes.
- **125-1 days:** In this stage, the airline plans for catering, cargo, final resource planning, customs & security, baggage handling and load control. Also here, the expected number of passengers, serves as crucial information to estimate the total demand.

Although there are many moments a passenger forecast is utilized before an airplane actually departs, the passenger forecast for this thesis focuses mainly on three important use cases.

1. **Aircraft allocation:** An airline needs to decide which airplane to allocate to which flight. The optimal airplane allocation depends strongly on how many passengers are expected. An

airline does not want to allocate an airplane too small such that less tickets can be sold. On the other hand, an airline do not want to use an airplane with too much capacity which can lead to more economic inefficient airplanes in terms of fuel consumption and other airplane size related costs. Therefore, different scenarios must be considered and changes in bookings, airplane maintenance and crew possibilities must be monitored. The expected number of passengers is not the only factor to consider but in the end it is important in determining which aircraft to allocate to which flight.

2. **Resource planning:** All planners in KLM rely heavily on a passenger forecast to schedule their resources. Their goal is optimize service while minimizing costs. The better the passenger forecast is, the better the resources can be scheduled. In this case, a bad forecast results in under or over staffing, which leads to customer dissatisfaction or unnecessary costs, respectively. Although, the final resource planning is not made within this passenger forecast time window, it is used for the initial resource plannings. It is beneficial to make the initial planning as accurate as possible, so later on not too many stirring changes are needed. These changes could lead to extra work, misunderstanding, errors and gives third parties who are involved less time react, which may cause unforeseen problems. For example, at the departure hall of KLM ¦ % of the employees are a temporal hire. Some of the jobs in these processes require special training and skill sets. It is not a given that a temporal hire contains the necessary skills, therefore, third parties like employment agency need time to recruit and/or train this type of employees.

3. **Feasibility analysis:** During planning the network schedule, operational capabilities are taken into account in terms of feasibility. Network planning provides a tentative flight schedule. Based on this flight schedule the passenger forecast is used to estimate the number of passengers to expect. Typical aspects considered are; is there enough manpower, is there sufficient equipment and can the infrastructure handle it. If for example the passenger forecast provides a wrong prediction, which follow in an advice that the operation cannot cope with proposed network, then the proposed network schedule is rejected. This cost revenue. Subsequently, network planning has to use their resources again to produce a new alternative flight schedule. This flight schedule could be seen as sub-optimal, assuming the first schedule was the best option.

Taking the above use cases into account, it is evident that a passenger forecast is of significant importance for an airline. Understandable, KLM has such a system already in place, so why do they want to replace it? There are two main reasons for this. First, there is a system in place called PTRA, that can predict between 1 and 180 days before departure. However, the business needs a forecasts earlier than 180 days before departure. To cope with this need, employees have build their own passenger forecast system. So it appears, KLM requires a comprehensive flexible system that can predict from 360 days prior to departure to 1 day before departure. Secondly, the PTRA system seems to be outdated. It does not use state of the art techniques and users have complained that especially on the 'long term' predictions i.e., between 125 en 180 days before departure, the outcomes are not reliable.

It can be concluded that there are clear business drivers to design a new forecasting system, embracing state-of-the-art techniques, to produce the best possible passenger forecast, which is the focus of this thesis. And in order to prove that the proposed model works better than the current system of KLM, PTRA will be adressed as a benchmark model. The next paragraph will focus on the system design, taking into account the underlying business value chain.

4.2 System Design

This section has the following structure. First, the data value chain will be explained in order to understand how the system will provide value for the business. Second, the business requirements of the system are outlined. Third, the required outputs the system must generate are enumerated. Fourth, the available inputs are discussed and briefly explained. Fifth, a discussion about performance metrics for different stakeholders is provided. Lastly, a final system design set-up is offered.

4.2.1 Data Value Chain

Before diving into the passenger forecast model, we first need to understand the data value chain. Figure 4.1 provides an overview on the data flows and how it eventually creates value for KLM.

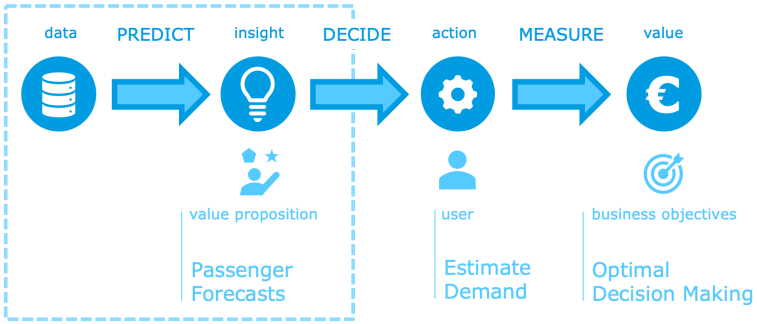


Figure 4.1: Data Value Chain for KLM

To elaborate, there is data, maintained in a specific data lake or data warehouse. A prediction model is build using this data. This model predicts the required outputs used by the planners to estimate the demand. For example, if a planner of a departure hall needs to know the demand, one has to know how many passengers to expect per time unit. To achieve this, a planner can multiply the expected number of passengers departing locally with an arrival profile. Based on this demand, actions can be taken in terms of resource planning, feasibility checks and security compliance. Finally, these actions can be measured and analyzed to evaluate how good the decisions were to determine the business value it has created. To conclude, the value proposition of this thesis is to build an accurate and reliable passenger forecast to provide insights for the users. The users eventually need to transform the insights into actual business value. How these insights must be delivered will be specified in the next subsection.

4.2.2 System requirements

There are mainly three important system requirements for the passengers forecast.

1. It should predict for any specific upcoming flight of KLM.
2. It should predict any moment between 360-125 days prior to departure.
3. The forecast should be more accurate than the output of the current KLM system.

To clarify requirement 1, an example could be that there is an upcoming flight with flight number 'KL 1234', flying from 'Schiphol airport' to 'London Heathrow airport' with flight date '2021-01-31', which is almost a year later than now. Than the passenger forecast model must predict for this certain flight the six different type of passengers that going to board.

These requirements are taking into account while modelling the passenger forecast. Although industrializing is out of scope, for industrializing the proposed model, extra requirements should be added such as:

1. Real-time data as input for predictions; more recent data could lead to better predictions
2. On-time predictions; predictions at the right (query) moments
3. 100% coverage; no missing predictions
4. Monitoring and alerting; know the performance of the system, the predictions and when things break
5. Automatic re-training of the model; new and more data can improve the model.

In addition, it could also be considered to look at the pre-processing pipeline and prediction model to make it more efficient such that the running time of the code becomes faster.

4.2.3 System Output

The are multiple ways a passengers forecast can generate outputs from which the business may benefit. For example, one can choose to predict distributions such that uncertainties are included. One could also choose to predict directly the expected demand. For example in the departure hall, instead of combining the expected number of passengers with an arrival profile, one can choose to predict how different passenger types flow through different processes. However, these different outputs needs extra effort in terms of change management. To minimize the change needed, it was decided, in collaboration with KLM, to stick with predicting the expected number of passengers. As discussed in the introduction, there are 6 types of passengers that are interesting for the use cases:

1. Economy class passengers
2. Business class passengers
3. Local arriving passengers (Passengers who's end station is the destination airport)
4. Local departing passengers (Passengers who's first station is the arrival airport)
5. Transfer arriving passengers (Passengers who use the arrival airport as a hub)
6. Transfer departing passengers (Passengers who use the departure airport as a hub)

This segmentation is based on the fact that these different type of passengers follow different processes. For example, a local departing passenger probably needs to check-in and drop off luggage. In comparison, a transfer departing passenger, needs to walk from one gate to the other. Or for example, business class passengers have different privileges compared to economy class passengers, requiring different resources.

4.2.4 System Inputs

For the prediction model, only inputs can be used that are available at the required moments. Moments we want to predict, are also referred to as query moments. Data that is available on the required query moments are:

- Current state of bookings for the cabin classes
- Locations: arrival and departure station
- Aircraft type
- Aircraft capacity for the cabin classes
- Scheduled date-time information for departure and arrival

- Flight number

All, except the bookings data, are logged in the Official Aviation Guide (OAG) and should be available for airlines. This guide provides a track record for over 900 airlines. Taken into account that each airline knows it own state of bookings, we assume that all airlines have these input data available. Therefore, the proposed model can be generalized for all airlines. These inputs will be further examined and discussed in chapter 5.

4.2.5 Performance Metrics

In this section, we will clarify what an accurate passenger forecast means. Accurate has different meanings for different type of stakeholders. In this case, we distinguish 3 types of stakeholders: Business, users and the model. In figure 4.2, an overview is given who those stakeholders are and what is important for them.

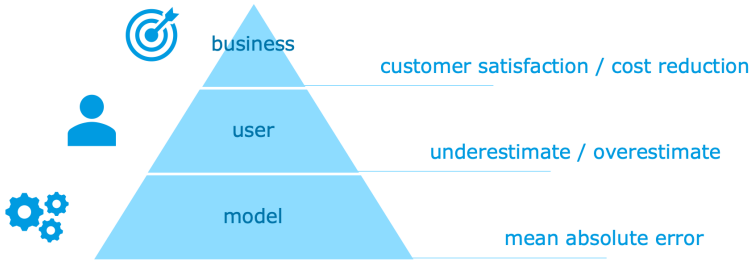


Figure 4.2: Performance Metrics Passenger Forecast

As mentioned in the introduction, customer satisfaction and cost control are of the utmost importance for an airline to stay competitive. For the business and users there are 3 aspects to consider in terms of customer satisfaction and cost reduction.

First, by optimal aircraft allocation the maximum number of passengers can board. If for example, a flight gets more passengers than the aircraft capacity and not a larger one is allocated, not all passengers can board which will decrease customer satisfaction and future loyalty. Furthermore the airline has to re-book the excess passengers, which cost extra manpower. Sometimes a fine such as few hundred euro's to the passenger has to be paid and/or provide a business class ticket instead of economy class. If necessary the airline also has to pay for hotel accommodation.

Second, network planning provides a tentative flight schedule. Based on this flight schedule, Operations estimates the demand. The demand is estimated by using the passenger forecast per flight. Based on this demand, KLM has to check if the operations can cope. Typical aspects considered are; is there enough manpower, is there sufficient equipment and can the infrastructure handle it. If for example, the passenger forecast provides a wrong prediction, which follow in an advice that the operation cannot cope with proposed network, then the proposed network schedule is rejected. This may potentially cost revenue. Subsequently, network planning has to use their resources again to produce a new alternative flight schedule. This flight schedule could be seen as sub-optimal, assuming the first schedule was the best option

Third, resource plannings are made based on the estimated demand, which is based on the expected number of passengers. By estimating too much demand unnecessary costs are made, by underestimating the expected demand customer service will be hurt.

In terms of modelling, the mean absolute error (MAE) is chosen instead of the mean squared error (MSE), because it is closely related to the business problem, as will be explained hereafter. With MAE, there is no preference for over-under estimation and it is not necessary to penalize bigger mistakes. Because by penalizing bigger mistakes, flights with larger capacity will probably get favoured since there absolute difference will be larger. The formula for MAE with an example can be seen in figure 4.3

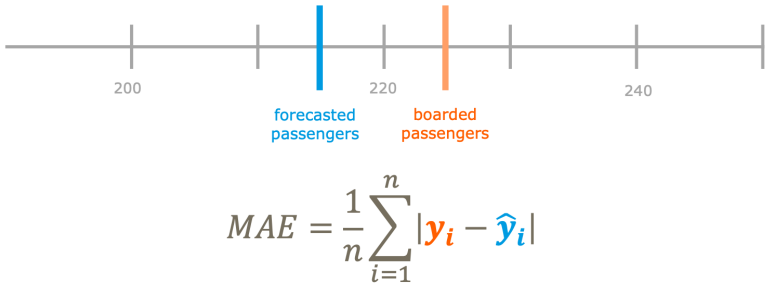


Figure 4.3: Mean Absolute Error

The MAE will be sliced and calculated per output we want to predict. To give an idea how the performance might be visualized and evaluated, an example is given in figure 4.4.

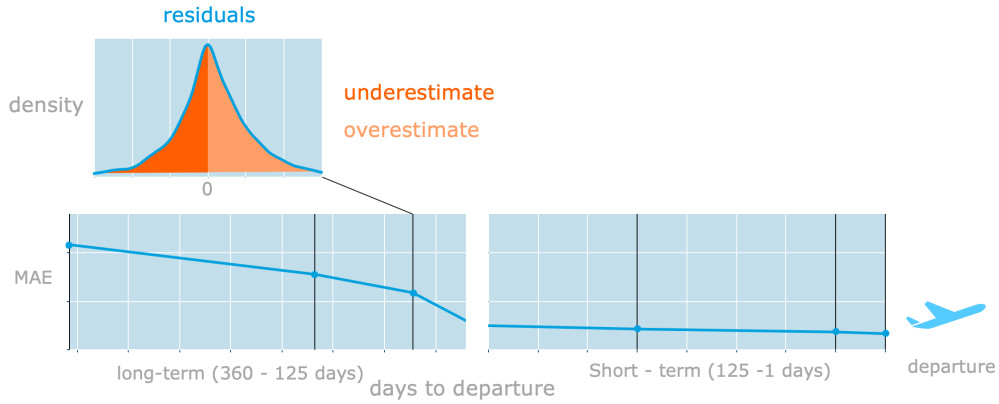


Figure 4.4: Performance Visualization

The MAE is on the y-axis, representing the model performance and on the x-axis the query moments prior to departure. Per query moment a residual plot can be made to determine if the model over or underestimates. In order to be trustworthy, it is important for the model performance to show stable prediction errors over the query moments. If users experience variation in the performance, it could be that they find that one time it does give good predictions and the next time, the prediction are way of target. This instability could lead to distrusting the prediction model. Furthermore, there are more ways to visualize and evaluate the performance of the model. This will be comprehensively discussed in chapter 9.

4.2.6 System Set-up

Finally, this section describes how the system could be designed and which steps should be taken in the process. Figure 4.5 gives an overview of this process.

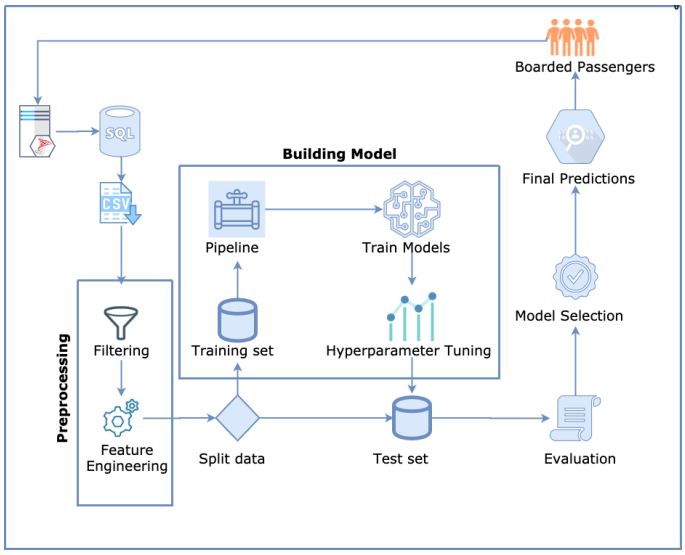


Figure 4.5: System Set-up

This is a general framework showing how the system is designed. First data is extracted from the local server. Subsequently, prepossessing is done with an automated pipeline that filters the data and generates new features. When the data is cleaned, a split in train and test data is made for validation procedures. These steps will be extensively discussed in chapter 6. With the train data, different models are trained and hyperparameters are optimized. For finding the right hyperparameters the test set is also used as validation set. The different architectures of the models will be evaluated on the test set. Then, the best performing model will be chosen to make predictions. An the end, one can choose to build an auto retrain for this process or to do it manually after substantial new additional data is available. However, this is out of the scope for this thesis, and will not be further considered here.

Chapter 5

Data Understanding

This section describes the data that is used for predicting different types of airline passengers between 126-365 days before departure. The data used is from 2016 till the end of 2019 and contains 144,195,921 records. The remaining of this chapter will describe which filters are used for data cleaning, then descriptive analytics will be provided about the target variables and then on variables that are used for predictions.

5.1 Cleaning and Filtering

In general, two filter rules are applied in the preprocessing pipeline in order to generate a clean dataset.

1. Drop record where no passengers boarded

2. Drop record where the capacity of the specific airplane is missing

In consultation with the KLM stakeholders it has been decided, that if no passengers boarded, the record is false. This is because for KLM it would be very unlikely to execute a flight with no passengers as it will generate no revenue except from the cargo. In total this affected only 0.0112% of the dataset. Furthermore, the capacity information of a specific flight is important for the prediction model. Because of its importance, rows with no information about the capacity have been dropped as well. In total this affected 0.02% of the dataset.

5.2 Target Variables

For the specific problem at KLM, there are six target variables that the model needs to predict for each specific flight number, as discussed in section 4.2.3.

Lets first look at how these different type of passengers differ over the flights. To visualize this, a histogram is plotted in figure 5.1 till figure 5.6. On x-axis the number of passengers on a flight are outlined, where on the y-axis the number of occurrences are displayed. Based on the economy and business class figures, it can be observed that most of the flights consist of smaller airplanes because most of the occurrences happens for relative smaller number of passengers. Furthermore, comparing the transfer and local passengers, it can be seen that there are more transfer passengers than local passengers. For the local passengers it happens often that are no arriving or departing passengers.

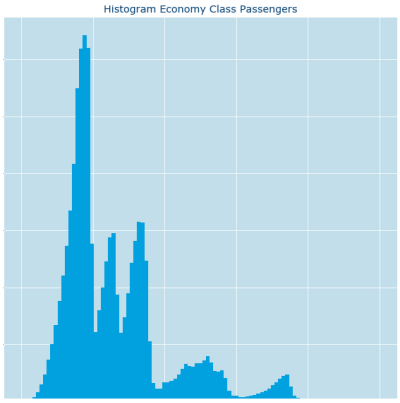


Figure 5.1: Histogram Economy class passengers

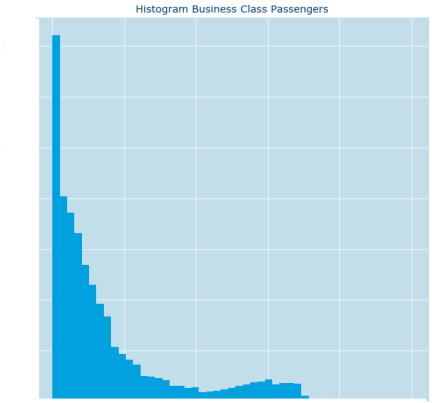


Figure 5.2: Histogram Business class passengers

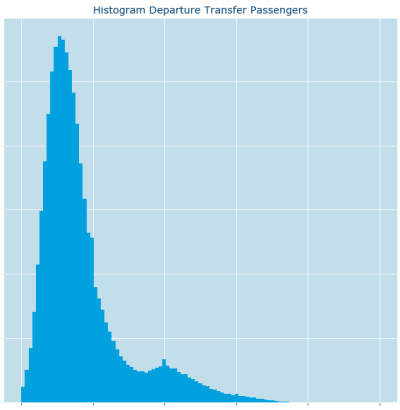


Figure 5.3: Histogram Departure transfer passengers

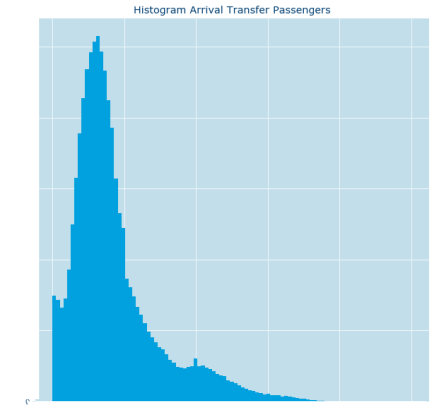


Figure 5.4: Histogram Arrival transfer passengers

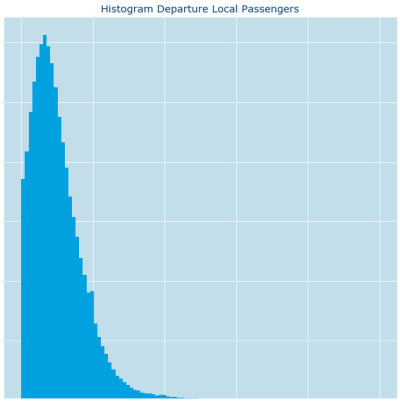


Figure 5.5: Histogram Departure local passengers

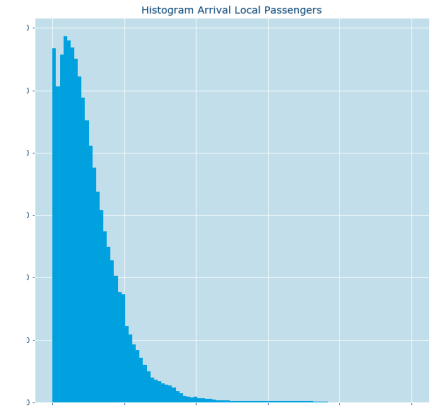


Figure 5.6: Histogram Arrival local passengers

How these target variables have evolved over time is plotted as time series in figure 5.7 till figure 5.12. To start with, there is a general upward trend for all targets variables. This implies that the number of passengers travelling with KLM is growing. According to there annual report (KLM, 2018) this is correct. Furthermore, in all figures a clear seasonality can be observed. In the first months of the year the least amount passengers are flying, where in the summer months, June, July and August the most passengers are travelling. In terms of passenger forecasting, these summer months are the most important to predict accurately. This is because the more passengers are travelling, the more important planning becomes. In those months operations is working at almost their full capacity and the effect of an percentage wise improvement becomes in absolute numbers larger. This is also strongly mentioned by the business side of KLM. This raises the question, if everything is running almost maximal, why not predict that each airplane is full with passengers?

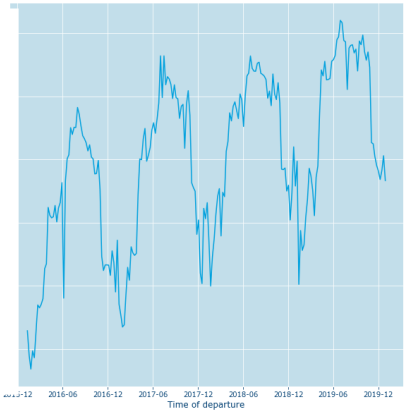


Figure 5.7: Economy class passengers over time

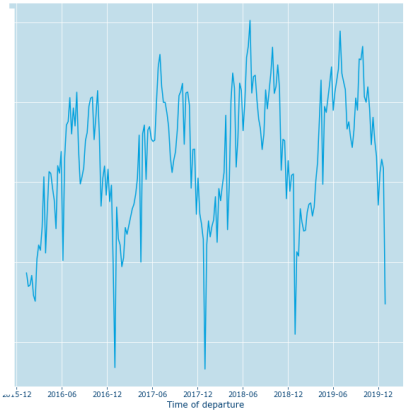


Figure 5.8: Business class passengers over time

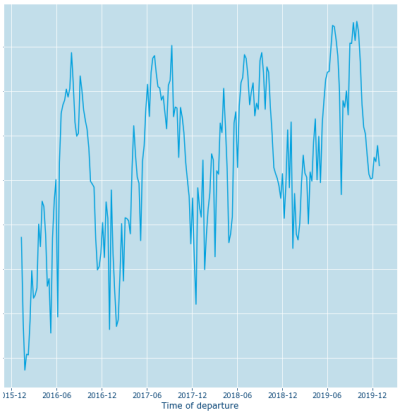


Figure 5.9: Departure transfer passengers over time

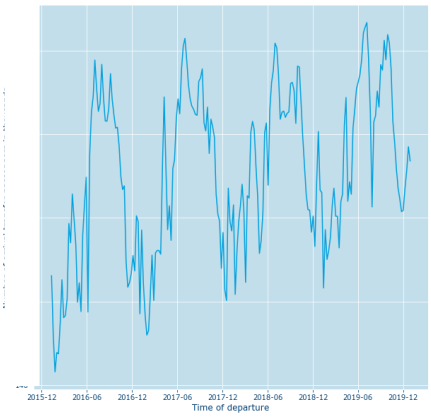


Figure 5.10: Arrival transfer passengers over time

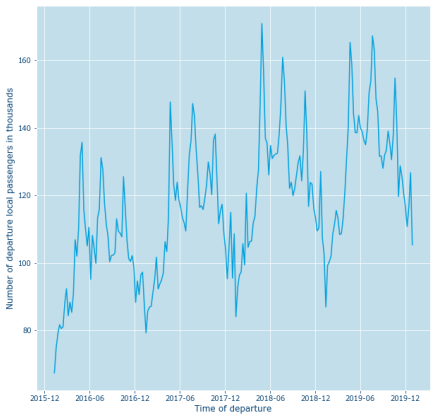


Figure 5.11: Departure local passengers over time

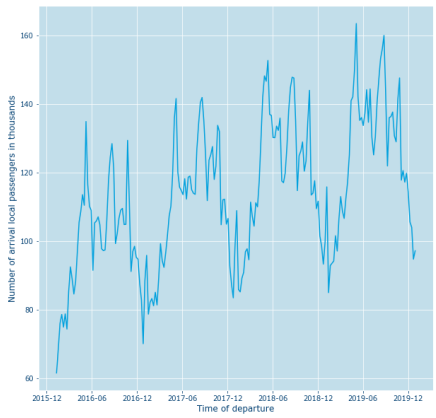


Figure 5.12: Arrival local passengers over time

Suppose you would predict that every airplane is full, it means that the load factor is one for both cabin classes. The load factor for example, for economy class is the number of economy class passengers divided by the capacity for economy class. Figure 5.14 shows how the mean load factors is over time.

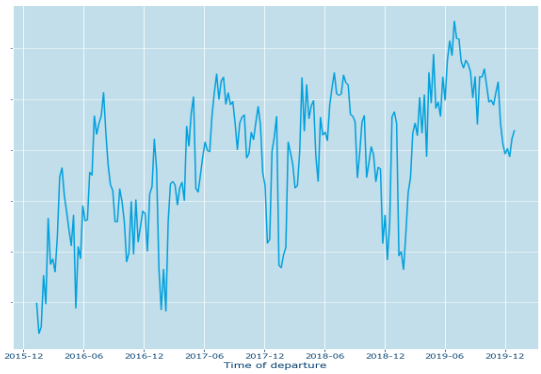


Figure 5.13: Load factor economy class passengers over time

T load factor for economy class comes close to one in the summer months, where in the earlie nonths of the year it is considerably lower. These load factors are strongly influenced by a partment within KLM named Revenue Management. Revenue Management is focused on optimizing profit by selling as many seats for the best possible price. For economy class this means, if the airplane is becoming full with bookings, they increase the ticket price, which therefore could result in less tickets sold. For the business class cabin it is a completely different game.

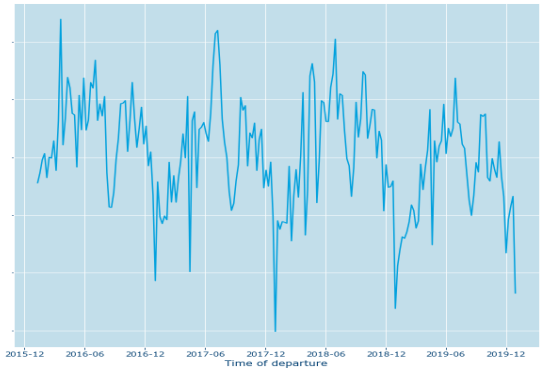


Figure 5.14: Load factor business class passengers over time

The load factor for business class is around 0.5 throughout the year. Revenue Management follows a different kind of strategy for this cabin class. They want to create a feeling of privilege for the customer. Thus if a potential customer wants to book business class, they want him/her to see that the cabin is almost empty so that they feel special. Within Revenue Management there are a lot of strategies, algorithms and business rules. They are constantly playing with the ticket prices and load factors to optimize revenue. This all is very interesting but out of scope for this thesis. In the end, predicting that every airplane has a load factor of one will not give a good performance. In addition, it does not happen often, that an airplane is fully booked.

Furthermore, in practise, the number of bookings does not equal the number of passengers. Passengers do not show up, employees of KLM join the flight without an official booking or too many seats are booked. Figure 5.15 and figure 5.16, show the mean absolute error between the total number of cabin class passengers and the total number of bookings for this class on flight level.

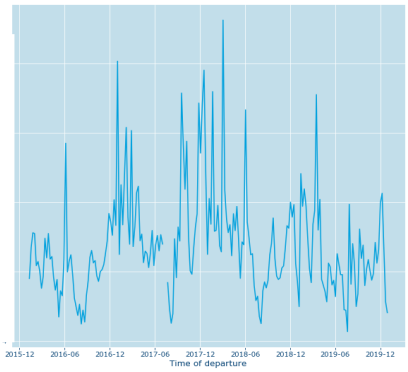


Figure 5.15: MAE economy class passengers and bookings over time

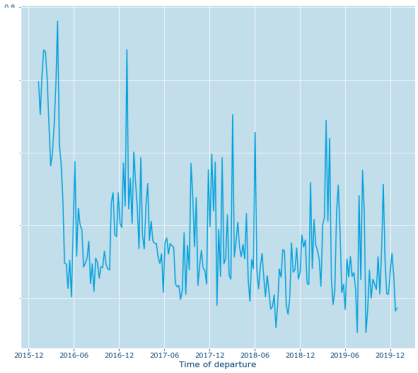


Figure 5.16: MAE business class passengers and bookings over time

As can be seen in the figures is that the number of bookings never equals the number of passengers that have boarded. Therefore, it can be concluded that predicting the number of passengers is different than predicting the number of bookings. By predicting the number of bookings instead, it will result in an additional error in the bias of the model.

Furthermore, it was observed that all target variables show an upward trend with the same seasonality. This might indicate the target variables are related to each other. Figure 5.17, shows a heat-map of the correlations between the target variables.

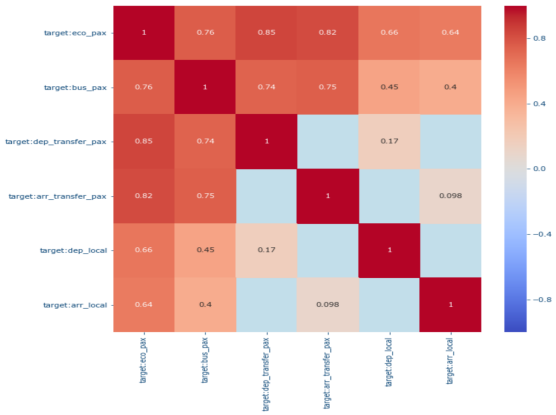


Figure 5.17: Correlation Matrix, target variables

In some cases, it is not possible to calculate a correlation because for some flights the target variable is zero, hence, not all flights have transfer or local passengers on board. There are some strong correlations between the cabin classes, which makes sense, because the larger the airplane the more capacity is available. Furthermore, there are some strong correlations between the transfer passengers and cabin class passengers. According to the business almost 70% of the passengers use Schiphol as a hub. Taking into account that 50% of all flights start from or have Schiphol as their final destination, this might clarify the correlations. Furthermore, the local passengers show weak or no correlation with the other target variables.

5.3 Predictors

The system inputs have been clarified in previous chapter. In this section they will be discussed in more detail. Table 5.1 presents the inputs by name, which data type it has, an explanation plus an example and what the source of the data is.

Table 5.1: Overview predictors

Name Variable	**Data Type**	**Information**	**Example**	**Source**
Flight number	String	The flight number of the flight leg	KL 1234	OAG
Aircraft subtype	String	The type of the aircraft	77W	OAG
Arrival station	String	The arrival airport of the flight leg	AMS	OAG
Destination station	String	The departure airport of the flight leg	CDG	OAG
Scheduled departure datetime	Datetime	Departure date time of the flight leg according to schedule	2019-21-01	OAG
Scheduled arrival datetime	Datetime	Arrival date time of the flight leg according to schedule	2019-22-01	OAG
Bookings economy class	Integer	Current number of bookings made for economy class	55	KLM
Bookings business class	Integer	Current number of bookings made for business class	10	KLM
Capacity economy class	Integer	Number of technical economy class seats	100	KLM
Capacity business class	Integer	Number of technical business class seats	20	KLM

Categorical Predictors

The categorical variables flight number, arrival and destination station are high in cardinality. They have 1029, 176, 175 unique categories respectively. Furthermore, Amsterdam Schiphol Airport is the most frequent airport in the data because it is the home airport of KLM. It covers roughly half of the data records for either arrival or departure station. In terms of aircraft subtypes, mostly small airplanes are used. There are small and large body airplanes. The small body airplanes are: E95, E90, 74E, 73W, 333 and 73J. This type of airplanes are mostly used for European flights. The large body airplanes can also transfer cargo, this types are: 772, 789, 74E and 781. Figure 5.18 shows how many times each aircraft sub type occurs in the total dataset.

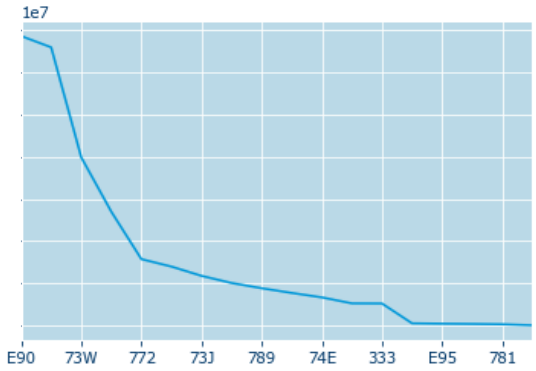


Figure 5.18: Value counts of aircraft types

In terms of transforming the categorical features three options have been considered: one-hot encoding, ordinal and label encoding. One-hot encoding transforms the categories to binary dummy variables. Hence, each category gets its own feature and then with a boolean it is indicated if the row corresponds to this feature with "1" and if not with "0". With the many categories the dataset has, this is not ideal because the dimensionality of the dataset will explode. The trade-off between ordinal and label encoding is contingent of the existence of any ordinality in the categories. In our case, there is no ordinality. For example, it does not matter for destination feature if Amsterdam Schiphol airport comes first or London Heathrow airport, hence label encoding has been chosen. With label encoding each unique category gets its own unique number assigned. This number then replaces the string in the cell.

5.3.1 Numerical Predictors

Besides the four categorical inputs, there are six numerical inputs. This could be extended by feature engineering. There are a lot of possibilities with feature engineering. However, for this thesis it is kept simple because the goal is more focused on the modelling part instead of finding new features. Subsequently, only the timestamp features are used to extract new features:

- Minute of the hour
- Hour of the day
- Day of the week
- Week of the year
- Month of the year
- Year
- Days before departure (Query moment)

Examples of features that could possibly add value and could be examined in the future are: last time the flight occurred, how many times a week does the flight takes place or how many flights with the same destination take place the same day etc.. These types of features could be further investigated as follow up work of this thesis, however for now they are out of scope.

For the numerical features it could be interesting to investigate if and how they are correlated. To visualize this, a correlation heat map is plotted in figure 5.19.

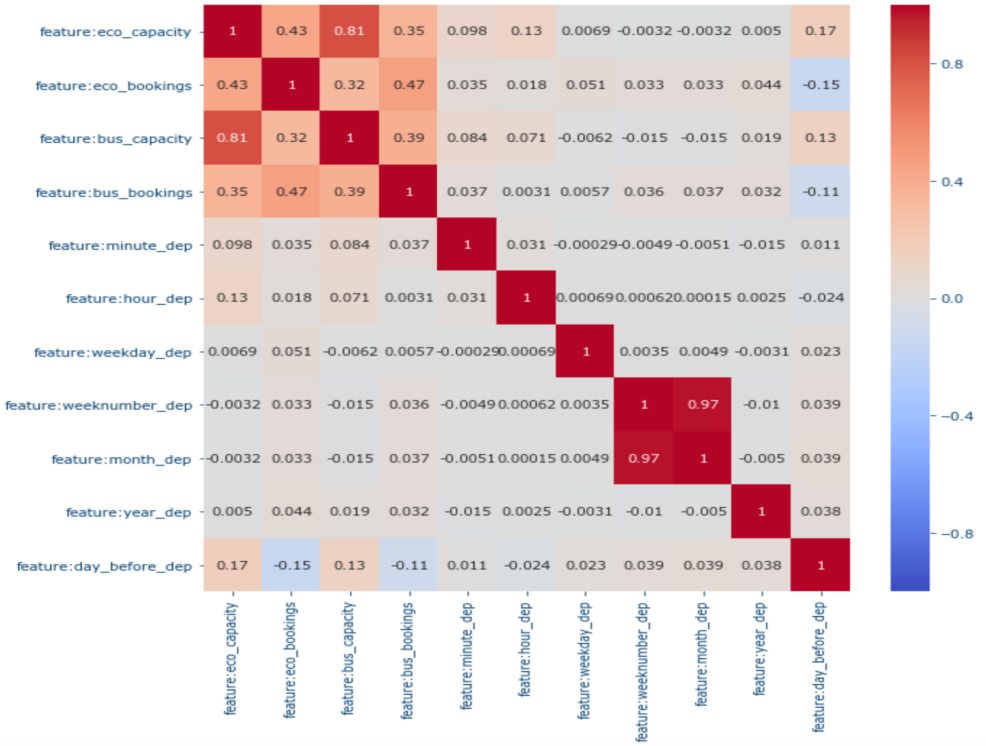


Figure 5.19: Correlation Matrix, predictors

In general, there are no correlations between date and time features and other predictors. Furthermore, there is a strong correlation between the economy class capacity and the business class capacity. This makes sense because the larger the airplane the more capacity there is available for both classes.

What is also interesting to notice is that over time airplanes are swapped during the planning period. This means that the capacity configuration changes for a certain flight over the query moments. Figure 5.20 shows a histogram displaying how many times the capacity has changed for a certain flight in the time period between 360 and 125 days before departure.

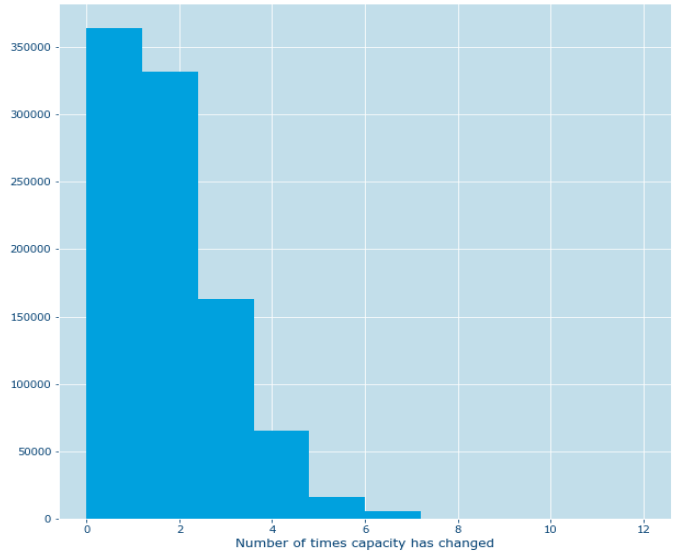
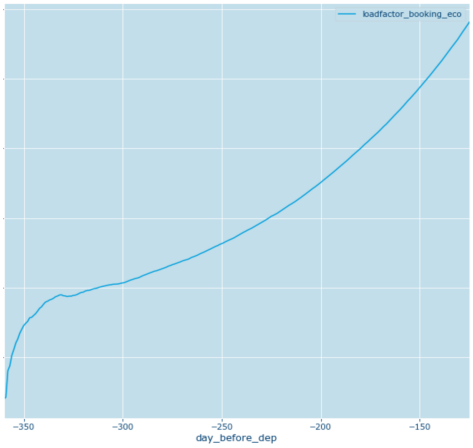


Figure 5.20: Correlation Matrix, predictors

Most of the time the configuration changes at least once, but a nine times change in configuration during the planning occurred as well. These changes make it more difficult to predict because the capacity information is not a constant.

Besides these aircraft swaps and the days before departure, there is another temporal feature, namely, the current status of the bookings. Figure 5.21 and figure 5.22 show how the load factor of the number of bookings has evolved on average over the days before departure. The load factor for the bookings is the number of bookings divided by its capacity. For example for economy class it is the number of economy class bookings divided by the economy class capacity. In the figure it can be seen that the longer before departure the lower the load factor. The closer to departure the more information is available about this load factor.



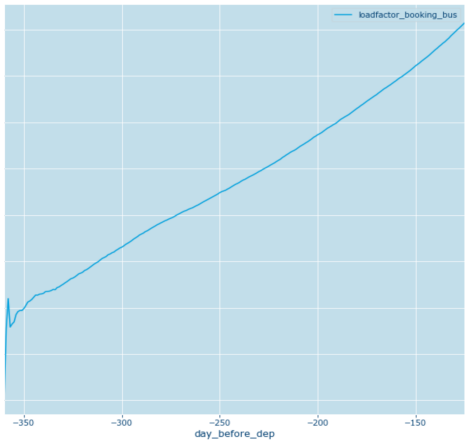


Figure 5.21: Load factor economy class bookingsFigure 5.22: Load factor business class bookings

For transforming the numerical features a scaling approach has been chosen. Scaling benefits the learning of the model especially with neural networks. By having the same scales among

features, the model does not prefer to learn large weights for large values. Large weights values can lead to instability in the network which results in poor performance during learning. For the scaling two options are considered: normalizing and standardization. The normalizer transforms the data such that the minimum value is "0" and maximum value is "1". This is done by using the following formula:

$$x_{new} = \frac{x - x_{min}}{x - x_{max}} \tag{5.1}$$

Where x_{new} denotes the new scaled value, x the original value, x_{min} the minimum value of the single numerical feature and x_{max} the maximum value of the numerical feature.

With standardization, the data is transformed such that the mean is zero and the standard deviation is 1. This is done with the following formula:

$$z_i = \frac{x_i - \overline{x}}{s} \tag{5.2}$$

Where z_i denotes the z-score, x_i the data point, $\overline{x}$ the sample mean and s the sample standard deviation. Standardization might especially be useful for models that rely on Gaussian processes. Furthermore, there are different arguments to use the one or the other. However, in this case, a performance comparison has been done. The comparison test is based on comparing the MAE performance on the test set. Out of this test it was concluded that the normalizer performs almost 10% better. Therefore, the normalizer is chosen to scale the numerical features.

5.3.2 Relationship between predictors and target variables

For the categorical predictors a Eta squared is applied to test the association between the target variables and the categorical predictors. Table 5.2 shows the results of this test. If for example the value is 0.52 such as with flight number on economy class passengers, it means 52% of all variance for economy class passenger is attributable to flight numbers.

Predictor/ target	Eta squared					
	Economy class	Business class	Departure transfer	Arrival transfer	Departure local	Arrival local
Flight number	0.52	0.724	0.668	0.669	0.706	0.706
Aircraft subtype	0.732	0.007	0.255	0.237	0.157	0.129
Arrival airport	0.499	0.349	0.003	0.002	0.168	0.078
Departure airport	0.522	0.327	0.001	0.003	0.053	0.152

Table 5.2: Eta squared test for categorical features on target variables

For the numerical features a Pearson correlation test is applied to quantify the association. The results are presented in table 5.3.

Pearson Correlation test						
Predictor/ Target	Economy class	Business class	Departure transfer	Arrival transfer	Departure local	Arrival local
Economy capacity	0.921	0.771	0.342	0.330	0.271	0.284
Business capacity	0.777	0.822	0.322	0.307	0.201	0.207
Economy bookings	0.450	0.374	0.153	0.185	0.146	0.176
Business bookings	0.355	0.422	0.136	0.149	0.121	0.128
Minute of departure	0.101	0.095	0.091	-0.026	0.067	0.014
Hour of departure	0.168	0.086	0.114	-0.075	0.070	0.072
Day in the week	0.024	0.024	0.005	0.003	0.014	0.023
Week of the year	-0.011	-0.012	-0.015	-0.004	0.005	0.007
Month of the year	-0.012	-0.012	-0.014	-0.004	0.002	0.007
Year	0.019	-0.03	-0.005	-0.003	0.027	0.023
Days before departure	0.140	0.130	0.066	-0.061	0.038	0.028

Table 5.3: Pearson correlation between numerical features and target variables

Although logically the features should indicate how many of which type of passengers are going to board, the correlation metrics do not show any specific predictive features. Therefore, a more complex model is needed to model the underlying complexities and non-linear relationships. This will be extensively discussed in the two upcoming chapters.

5.4 Validation Procedure

The data available covers the years 2016 to 2019. As shown in the previous chapter there is a clear seasonality in the dataset. Therefore, a split in training and test data should comprise all seasons in order to capture the different patterns. For this reason it has been decided to take the last year(2019), the most representative one, as the test data. The remaining data will be part of the training set. Figure 5.23 shows the data split visually.

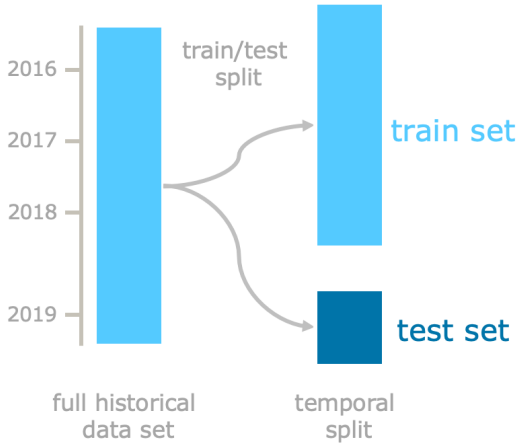


Figure 5.23: Validation procedure, data split

The choice for this split is based on (Raschka, 2018). This paper recommends that when having large datasets, a 2-way holdout method (train/test split) should be sufficient for performance estimation for the models. After this split, the train set contains 104,531,717 records and the test set 39,664,204 records. Furthermore, a rolling window method is considered, but because of the huge amount of data used for this thesis, this procedure is unfortunately out of scope as it would have resulted in very long training and testing times. Due to the time constraints and shortage of

resources for this project, this would have limited the number of experiments to try. For future research, given sufficient resources, it is recommended to include a rolling window validation to be more confident about the results of the experiments.

Finally, before going on to the modelling part, lets have a look at how the final data frame looks like after filtering, feature engineering and scaling.

Feature name	**Example**	**Data type**
Flight Number	5	Integer
Aircraft subtype	3	Integer
Arrival airport	143	Integer
Departure airport	87	Integer
Economy capacity	0.644	Float
Business capacity	0.575	Float
Economy bookings	0.052	Float
Business bookings	0.033	Float
Minute of the hour	0.12	Float
Hour of the day	0.478	Float
Day of the week	0.5	Float
Week of the year	0.788	Float
Month of the year	0.818	Float
Year	0.667	Float
Days before departure	0.004	Float
Economy class passengers boarded	221	Integer
Business class passengers boarded	27	Integer
Departure transfer passengers boarded	87	Integer
Arrival transfer passengers boarded	13	Integer
Departure local passengers boarded	44	Integer
Arrival local passengers boarded	17	Integer

Table 5.4: Final data frame for modelling

Chapter 6

Multi-task Learning with Deep learning

This chapter will explain the proposed model for predicting different type of airline passengers simultaneously. First the model architecture will described. Second, a comparison between single task neural networks and MTL will be given. Third, experiments regarding what to share in the network and auxiliary tasks will be discussed. Last, conclusions will be given about the final choices of the proposed model architecture.

6.1 Model Architecture

In this section the architecture of the proposed model will be explained. Out of the limited experiments done, the best established architecture in terms of MAE performance on the total loss function of the test set, is presented in figure 6.1. All neural networks are implemented with the Keras library using the functional API. For MTL it is required to use the functional API since it gives much more flexibility in modelling. Furthermore, all models are trained on a local server of KLM with 8 GPU's and memory of 36 GB.

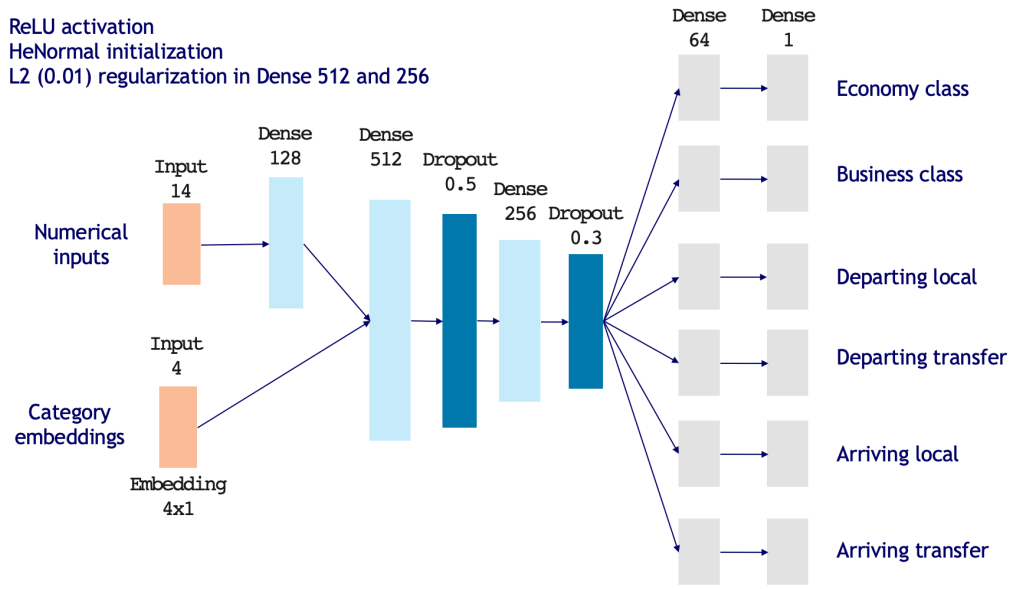


Figure 6.1: Multi-task learning final deep neural network architecture

In total there are 18 inputs for the model, whereof 4 categorical features and 14 numerical features. Each categorical feature has its own embedding layer. After the numerical inputs a fully-connected dense layer of 128 neurons is placed. Each dense layer has a 'ReLu' activation function and 'HeNormal' initialization. The relu activation uses the following formulate:

$$f(x) = max(0, x) \tag{6.1}$$

Where x denotes the input of a neuron. The 'HeNormal' initialization draws samples from a uniform distribution within [-limit, limit] where limit is $sqrt(6/fan\_in)$ and where fan_in is the number of input units in the weight tensor. This based on the paper of (He, 2015).

The embedding layers and dense layers are concatenated to a dense layer of 512 neurons. For regularization an Ridge l2 regularizer and dropout is added to network. The network follows a diverge structure and in the end, before each output layer, a separate dense layer of 64 neurons is placed. Without going into much detail on every parameter, there are 5 elements that have a relative big impact on the performance. These will be discussed in the next subsections.

6.1.1 Loss Function

As discussed earlier, the evaluation metric of the prediction model is MAE. This however does not have to be the loss function during training per se. It could for example be that one chooses MSE as loss function for optimization and uses MAE for the final evaluation. Therefore, as loss function the MSE and MAE are considered. These two have been compared by running exactly the same experiments, with the same data and same model architecture. MAE as loss function resulted in 5% better performance in comparison with MSE as loss function. In this respect, performance is defined as the MAE on the test set where each target is summed to a total MAE. Based on this results the MAE as loss function has been the final choice for this proof of concept. In addition, all outputs are weighted equally, therefore, during training they are considered equally important to learn. The loss function can be formulated as follow:

$$\sum_{j=0} L(\hat{y}_j, y_j) \tag{6.2}$$

Where j is denoted as the number of tasks, in this case there are six outputs to predict thus six tasks. $L(\hat{y}_j, y_j)$ is denoted as the a single loss function with MAE for task j, where $\hat{y}_j$ is the predicted value and y_j the value value.

Generally, this loss function is the mean of the overseen data of the absolute difference between the predicted and true value. Where N is denoted as the number of observations. The MAE loss function is less sensitive to outliers in comparison with MSE. In this case, it looks like this benefits predicting airline passengers for KLM. This is probably because the errors for large airplanes in comparison with small airplanes will weight more with MSE. Consequently, flights with large airplanes will be preferred during training. In practice most of the KLM flights are with small airplanes, therefore on average this would decrease the total performance if large airplanes are preferred more.

6.1.2 Embedding Layer

Each categorical input has its own embedding layer. This improved the total loss performance over the test set with approximately 5.1%, whilst keeping the rest of the network and data exactly the same. In general, the embedding layer compresses the input feature space into a smaller space and therefore embedding layers can only be used as the first layer in the model. In short, the embedding layer tries to learn the optimal mapping of each of the unique categories and converts it to a vector of real numbers. The size of this output vector can be seen as a hyper parameter and is equal to the output dimension. For now a common rule in NLP is used namely,

$$outputdimension = Min(64, \frac{N_{unique}}{2}) \tag{6.3}$$

Where N_{unique} is denoted as the number of unique values for the category. The output dimension could be further optimized by doing more research about how to find the optimal value. This however, is not further tested and investigated and could be done in future research.

6.1.3 Extra Dense Layer

A relative small improvement for the network is created by adding a dense layer after the numerical inputs. This dense layer contains 128 neurons and like the other dense layers has a 'ReLu' activation function with 'HeNormal' weight initialization. Overall, when keeping everything the same and only remove the dense layer after the numerical input, the total performance decreases approximately 1.2%. Although this is not a big number, it is still worthwhile because especially in the airline sector each percent counts. The dense layer adds value as it provides extra flexibility and space to learn the interactions between the numerical features. Possibly, there is some extra complexity in these interactions which can be absorbed better with this extra dense layer.

6.1.4 Regularization

Because the model tends to overfit relative fast, two regularization methods are tested. Both improved the network's performance which indicates regularization benefits predicting airline passengers on our query moments.

Dropout

The first regularization method discussed is the dropout method. The dropout method randomly ignores or 'droppes out' layers output. Consequently, each update to a layer, while training, gives a different view of the configured layer. This makes the training process more noisy and forces nodes that are not dropped out to learn more of the inputs. (Srivastava et al., 2014) argures that: "Units may change in a way that they fix up the mistakes of the other units. This may lead to complex co-adaptations. This in turn leads to overfitting because these co-adaptations do not generalize to unseen data.". Therefore, dropout can help to prevent overfitting and therefore can be an alternative as regularization method. By experimenting with dropout layers, it is found that the performance decreases 7.8% when the dropout layers are left out. Therefore, it is strongly advised to include dropout layers in the model. For further optimization of the prediction model it could be interesting to experiment with different parameters for the dropout rates.

Ridge regularization

The second regularization method tested are the Lasso, denoted as l1 and Ridge, denoted as l2 regularization. L2 regularization adds squared magnitudes of coefficients as a penalty to the loss. Where l1 adds absolute value of magnitude of coefficient as penalty to the loss. Both methods are applied on the dense layers with 512 and 256 neurons. Overall, the l2 regularization performed best and improved the total loss performance on the test set with approximately 6.2%. The l1 regularization improved the performance with approximately 3.1%. Therefore, the l2 regularization has been chosen for the two dense layers.

6.2 MTL versus Independent Neural Networks

6.2.1 Performance Comparison

Now the final model architecture has been clarified, the first interesting comparison to make is MTL versus independent single task neural network models. Let's examine how MTL performs compared to a single network for each required output, with the same architectures only one output layer instead of six. Table 6.1 shows the results of both approaches on the test set. The number per target variable are first grouped by query moment and then the average MAE score

is taken on that specific query moment. In that way each query moment is considered equally important and query moments with more data records will not out-weight query moments with less data records. This results in a data frame where each query moment has its own average MAE score. Table 6.1 show the average score of this dataframe scores. In addition, a last row is added to the table, where the total score can be seen. For the total MAE and standard deviation scores all MAE's and standard deviations of the target variables are summed.

Table 6.1: Comparison Multi-task learning versus Single-task models

Target Category		**Multi-task learning**	**Single-task learning**	**Difference**
Economy Class	MAE	**12.405**	12.760	0.355
	st.dev	**1.204**	1.308	0.511
Business Class	MAE	2.883	**2.765**	-0.118
	st.dev	0.346	**0.221**	0.266
Departure Transfer	MAE	8.445	**8.240**	-0.205
	st.dev	0.894	**0.790**	0.418
Arrival Transfer	MAE	**7.934**	8.378	0.444
	st.dev	**0.842**	1.008	0.554
Departure Local	MAE	**7.756**	7.888	0.132
	st.dev	**0.628**	0.728	0.368
Arrival Local	MAE	**6.941**	10.038	3.097
	st.dev	**0.543**	1.31	1.192
Total	MAE	**46.364**	50.069	3.705
	st.dev	**1.941**	2.376	1.371

As can be seen in the table, MTL performs overall better than single task models, in percentage MTL performs approximately 7.4% better in terms of MAE. To compare the MAE between the two models a Wilcoxon signed-rank test is applied. Each target variables shows a P-value of 0.000. Therefore, the differences between STL and MTL on MAE for each target variable are significant. In terms of variation of the query periods, MTL shows more stable predictions because the standard deviation is lower. Only the business class passengers and departure transfer passengers perform better with STL. That STL performs better than MTL is known as negative transfer. With negative transfer sometimes independent networks perform better. There are two reasons why negative transfer happens. First there is an optimization challenge, if gradients of one task interfering with the training of another task. Basically when you apply it, the network computes gradient for task one and computes gradient for task two. If gradient one hurts the weights for task two then optimization becomes more difficult. Furthermore, tasks learn at different rates. If one task is learning a lot faster than another task, it might end up learning task one very quickly and might get stuck trying to learn task two because it has already learned and does not want to learn something else. Essentially, the optimization gets stuck in a local optimum. The second issue might be a limited representational capacity. In general, MTL networks need to be much larger than a single task network. If the network is not large enough then it is going to under fit, which could be seen as a symptom of limited representational capacity. However, in this case, the exact same networks are trained, therefore it might not be the best performance, but is the most fair comparison.

Furthermore, within the loss function each task gets the same weight, hence each task is considered equally important. These weights can be adjusted to what task you find more important. Therefore, one could consider to give more weight to business class and departure transfer in order to match or even outperform the performance of STL. Changing the weights in the loss function is however a whole new research topic and requires more time and resources to investigate. For this thesis project it is out of scope but would be a recommend step to follow up separately. To better understand how MTL differs from STM, the next subsection will discuss the difference in feature importance, how individual predictions are made and the partial dependencies of the features.

6.2.2 Difference in Feature Importance

The feature importance is based on SHAP (SHapley Additive exPlanations) values and DeepLift, which is a method to explain predictions proposed by (Lundberg and Lee, 2017) and (Shrikumar et al., 2017), respectively. The feature importance for all target variables are calculated on 5000 records of the test data. According to (Lundberg and Lee, 2017), this number should be sufficient to get a general overview of the model and its feature importance on the total dataset. Each passenger category will be discussed briefly.

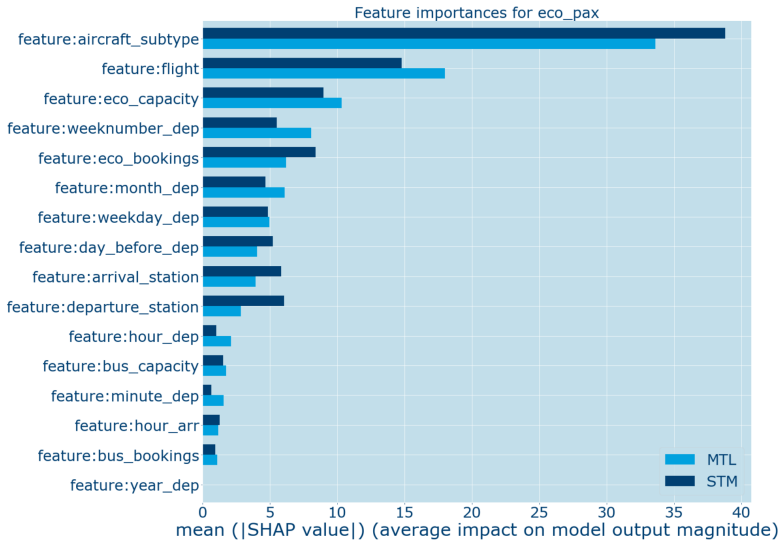


Figure 6.2: SHAP Feature Importance of Multi-task learning and Single task learning for Economy Class

For the economy class passengers it is interesting to see is that MTL gets more value out of the flight feature. This could be because with MTL it learns more examples of the same flight and therefore learns extra patterns throughout the other tasks it tries to predict. This is also the case for some more general features such a week number, month, hour and minute of departure. STM in contrast, learns more specific for the task by assigning more value to arrival and departure station. In addition, it uses the bookings data more for its predictions.

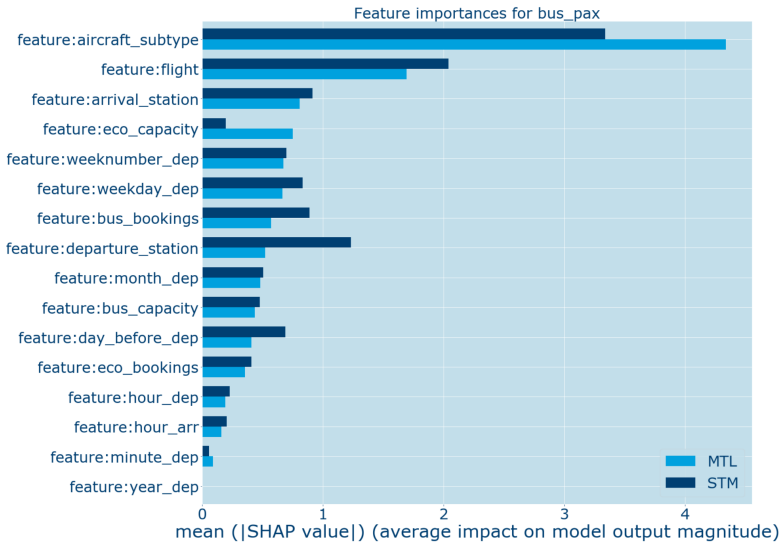


Figure 6.3: SHAP Feature Importance of Multi-task learning and Single task learning for Business Class

For business class, it is interesting to see that MTL uses the economy capacity features a lot more than the STM. This is not a task-specific feature for business class but still MTL uses it. However, taken into account that STM performs better on this output, it might not be the best way to learn for a neural network. Therefore, it seems that negative transfer has taken place where with learning this task this features was interfered by the other tasks. Furthermore, STM uses the departure station, bookings data, days before departure and flight number more. Taken the performance differences into account, it might be that more information is contained in these features.

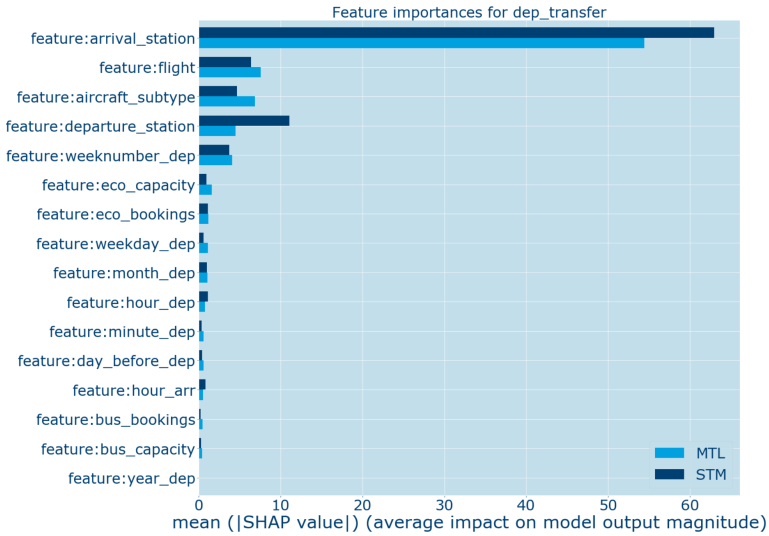


Figure 6.4: SHAP Feature Importance of Multi-task learning and Single task learning for Departure Transfer

Predicting the target departure transfer passengers is in terms of MAE performance better with STM. Looking at the feature importance, this is probably because it uses the departure and arrival station better. According to the business, there are typically stations that are used as hub

and that makes it a more task specific features. However, this could than also be expected for the target: departure local, arriving transfer and local.

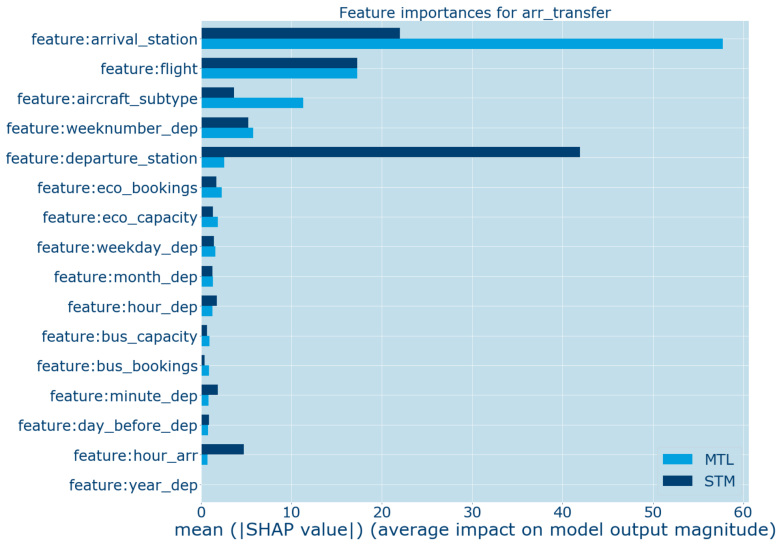


Figure 6.5: SHAP Feature Importance of Multi-task learning and Single task learning for Arrival Transfer

A large difference between MTL and STM can be observed between arrival and departure stations. Perhaps both balance each other out but this should be further investigated. This information alone is not enough to make further conclusions. Another relative large difference can be seen at the aircraft sub-type feature. In terms of MAE performance, MTL performs roughly 5% better. Besides the arrival and departure station, it looks like aircraft sub-type feature add extra value to the learning of the prediction model.

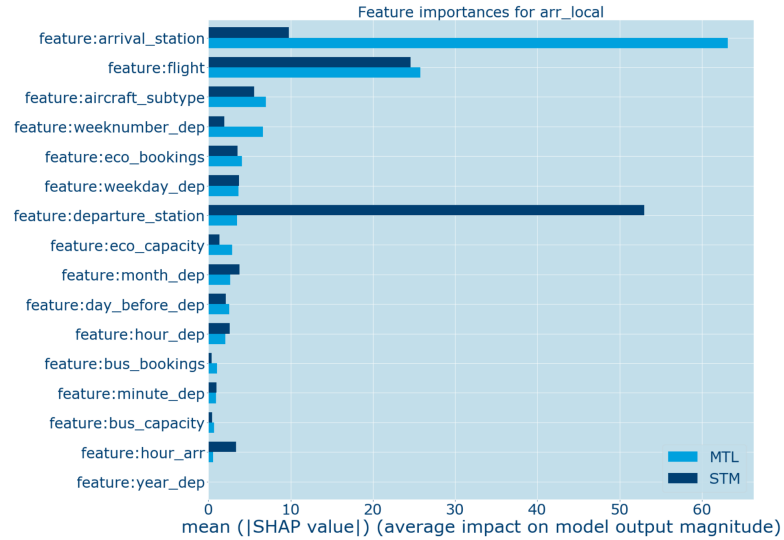


Figure 6.6: SHAP Feature Importance of Multi-task learning and Single task learning for Arrival Local

The same patterns can be observed for the arrival local passengers. Only, in this case the differences between the station are bigger. Interestingly, the difference in performance is also

much bigger. MTL performs in terms of MAE approximately 30% better. This might indicate that with MTL the shared learning helps to better understand the departure and arrival station.

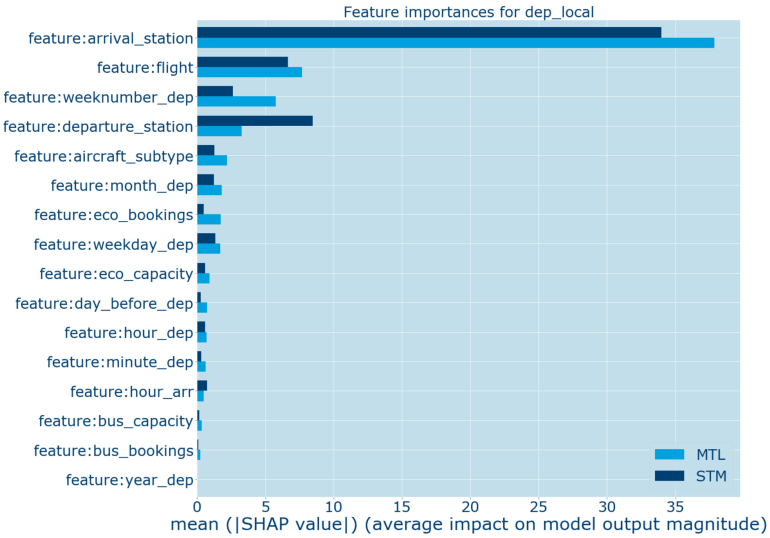


Figure 6.7: SHAP Feature Importance of Multi-task learning and Single task learning for Departure Local

Lastly, the departure local passengers will be discussed. Besides, the arrival and departure station, it seems that MTL uses more general features such as: month, economy bookings and days before departure. Overall, it can be observed that there a differences in feature importance

between STL and MTL, hence, the models uses the features differently. This may imply that the models learn differently. To better understand how this features interact and are distributed over the SHAP values, the next section will look into how individual flights are predicted with STM and MTL.

6.2.3 Difference in Individual Predictions

Figure 6.8 shows an example of how MTL makes a prediction for a certain flight. This example, is flight number 'KL 0569' flying to airport code 'AMS' with flight date '2019-08-12'.

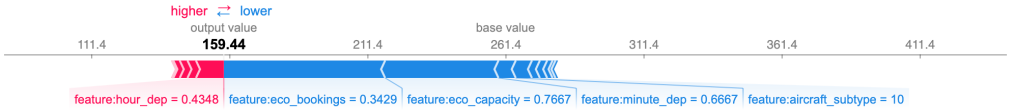


Figure 6.8: SHAP individual prediction for economy class with multi-task learning model

This figure can be interpreted as follows. The MTL models has predicted 159.44 economy class passengers. The base value is 261.4. This base value is the average model output over the training dataset that is passed. Features that increase the base value are colored in red and features that decrease the base value are in blue. The size of the visual representation of the feature represents the magnitude of its contribution. So in this case, the economy bookings with a feature value of 0.343, had the most impact on the prediction. Hence, subtracting the blue bars magnitude from the base value and adding the red bars magnitude, results in the output (prediction)value.

The next figures displays exactly the same flight only now predicted with STM.

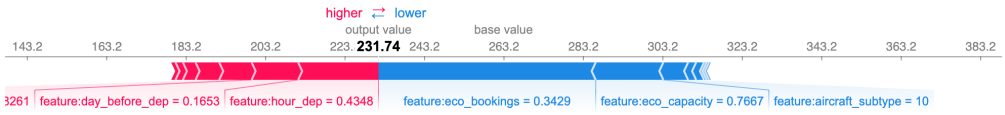


Figure 6.9: SHAP individual prediction for economy class with single-task model

The base value is roughly the same, the prediction however is different. Looking at the red part, STM uses more features with greater magnitudes to increase the prediction. Whereas the blue part is more or less similar to MTL, as expected and described in a previous section. The more general features are clearly used differently by MTL and STM.

In addition, figure 6.10 and figure 6.11 provide another example. This example, is flight number 'KL 0714' flying to airport code 'PBM' with flight date '2019-01-01'.

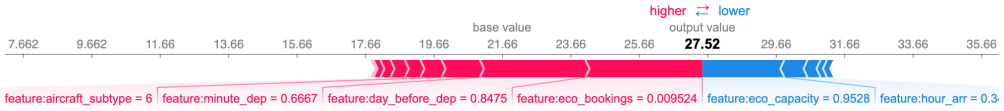


Figure 6.10: SHAP individual prediction for business class with multi-task learning model

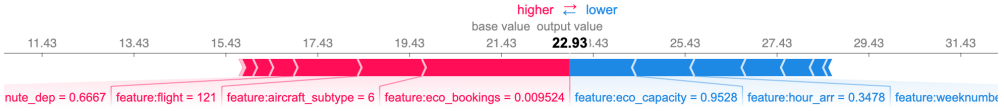


Figure 6.11: SHAP individual prediction for business class with single-task model

The base value is again roughly the same for both models. The red part has more or less the same magnitude which increases the prediction. However, the blue part differs. The difference lays mainly in the use of the general features such as hour of departure and week of the year. The same patterns occur for the other targets as well.

After this analysis of the importance of the different features with respect to the individual predictions, it can be concluded that features in STM and MTL are used differently and have different importance. This might indicate that they follow a different learning pattern. To get more insights in this matter, the next section will describe the partial dependencies of the features within the models.

6.2.4 Difference in Partial Dependencies

In this section, the target variable arrival local passengers will be used as example. In the previous section it was discussed that more general features such as number of days before departure or week of the year, differ in how they are used between MTL and STM. With partial dependence plots (PDP) it can be shown how these features affect predictions per model. It basically works as follows; there is a fitted and trained model in place, we take a data record of the test data and repeatedly alter the value of the feature we want to test. This results in a series of predictions, which can be plotted to visualize the pattern about how the model uses the feature. An example, is given for the feature: days before departure.

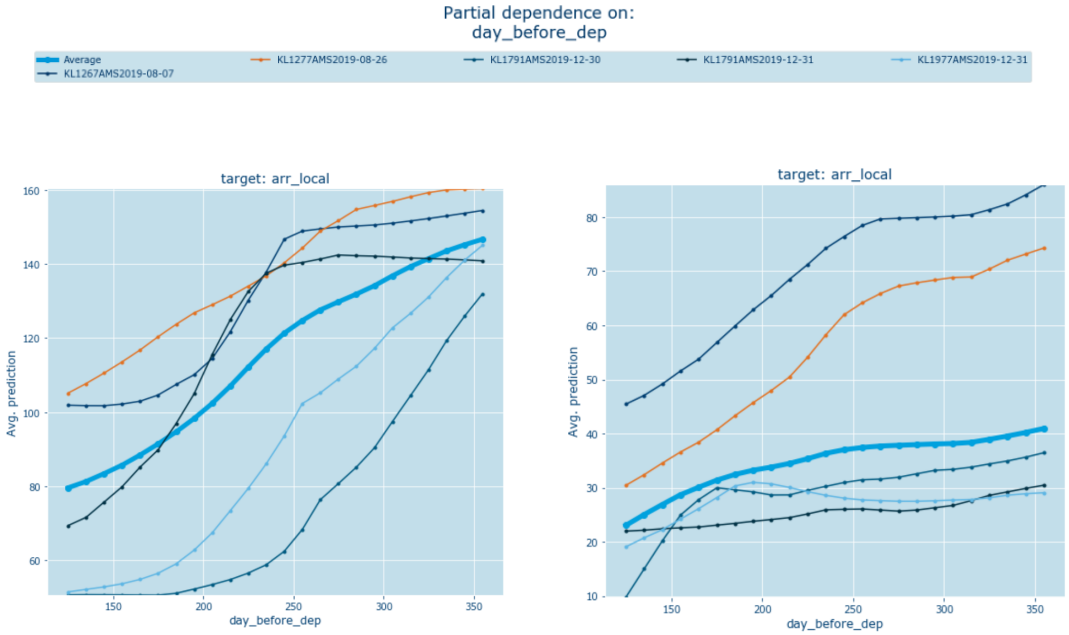


Figure 6.12: PDP, Single-task learning

Figure 6.13: PDP, Multi-task learning

In the top of the figure, a legend is provided which show which rowkeys are used. A rowkey makes a flight unique by, flight number, destination and date of flight. In this case, five unique flights are used for PDP. The tick blue line is the average of these five flights. By altering the feature value, and predicting with the different values for it, it can be observed how the models uses this feature. For example, figure 6.12 displays the single-task learning model and figure 6.13 the MTL model, both on the same records, with the same changes for the feature days before departure. Analysing these figures, show that STM reacts more (steeper slope lines) on changes in feature values than MTL. This confirms the theory discussed in chapter 2 that MTL has a regularization effect. Therefore, MTL is more robust for changes in feature values compared to STM. Furthermore, as explained in the previous section, regularization works positively for this problem, where arguments where given with drop out layers and Ridge regularization. Interestingly, some rowkeys show different learning patterns. For example, the light blue line belonging to 'KL1977AMS2019-12-31'. With STM, the value increases almost linearly with the number of days before departure. In case of MTL, it shows a more a logarithmic pattern. The same holds for rowkey, 'KL1791AMS2019-12-30'. This may confirm that between STM and MTL, different types of learning patterns of the features take place.

PDP, shows how a single feature impacts the predictions, but what they do not show is the distributions of effects. SHAP dependence contributions plots provide more or less similar insights but add more detail to it. Figure 6.14, shows in addition to the PDP a distribution plot of the feature 'days before departure'.

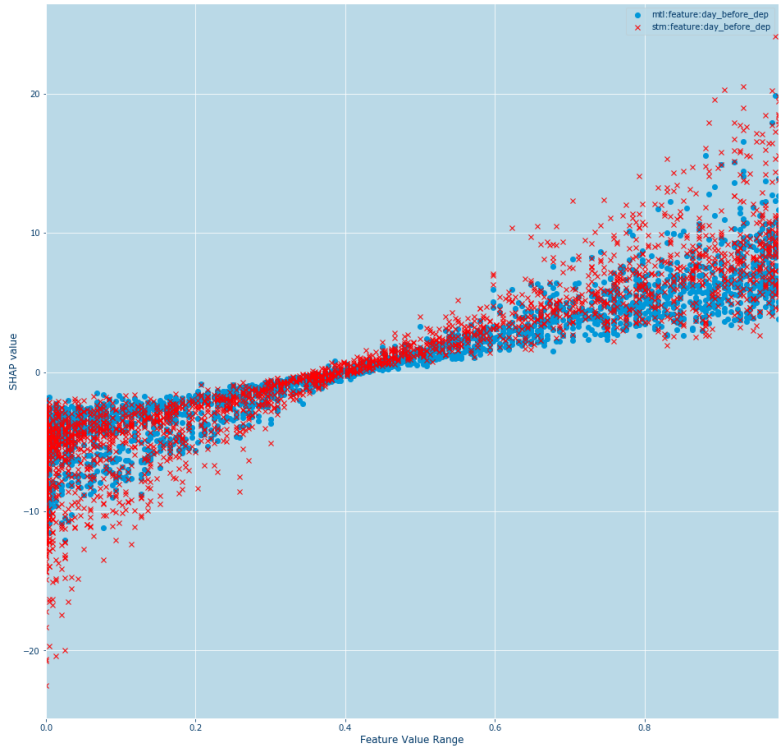


Figure 6.14: SHAP Dependence Contribution Plot

This figure shows, how different values for a feature influence the SHAP values, hence, the contribution to the prediction. The x-axis denotes the different (scaled) values for the features, where y-axis show the SHAP value. For this example, 5000 records are visualized on the target departure local. The red points indicate the STM and the blue dots MTL. The shape of both models is more or less the same. Getting closer to the departure day decreases the prediction, whilst longer before departure increases the prediction. Because higher values for the feature are towards right of the x-axis. The more to right, the higher the SHAP value, hence, the more value is added to the base value. Approximately, on the half of the x-axis, the contribution becomes negative, hence, value is subtracted from the base value. This is consistent with the PDP plots as shown in the previous paragraph. In these PDP plots it was also clear that MTL is less affected when changing the feature value. This can be seen even better in the SHAP contribution plot. The blue points are less wide spread than the red colored points, which confirms the expected regularization effect of MTL.

6.3 Grouping tasks and Auxiliary inputs

As for MTL, multiple experiments have been performed to better understand what layers to share in the model. In addition auxiliary inputs have been tested as well because according to the literature, it may improve the performance. Concerning what to share, three different groupings are tested. The first grouping is similar to the basic architecture as discussed in section 6.1, and referred to in figure 6.1. This is for now denoted as 'no-groups'. The second grouping contains 2 groups. In this example, the cabin classes are grouped and the local and transfer passengers are grouped. The architecture can be seen in figure 6.15.

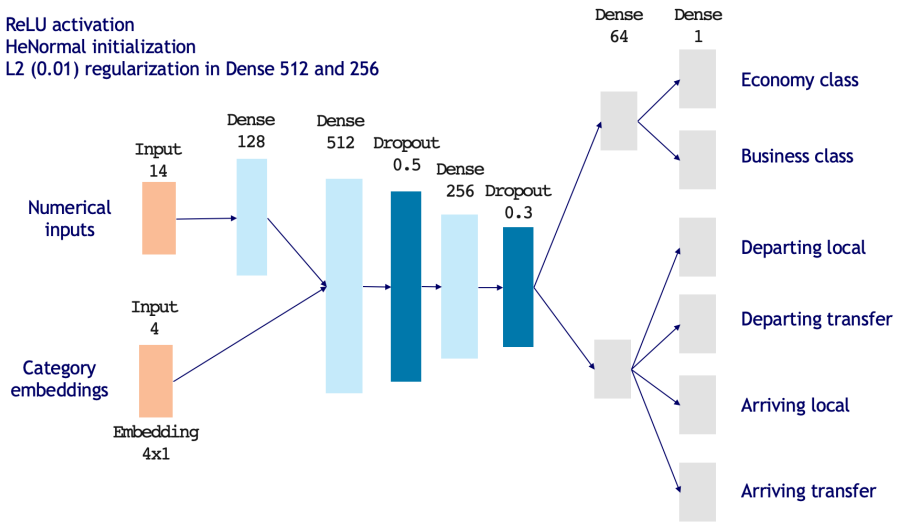


Figure 6.15: Multi-task learning final deep neural network architecture with 2 output groups

The third grouping contains 3 groups, where the cabin class is one group, the arrival passengers a group and the departure passengers is a group. The architecture can be seen in figure 6.1.

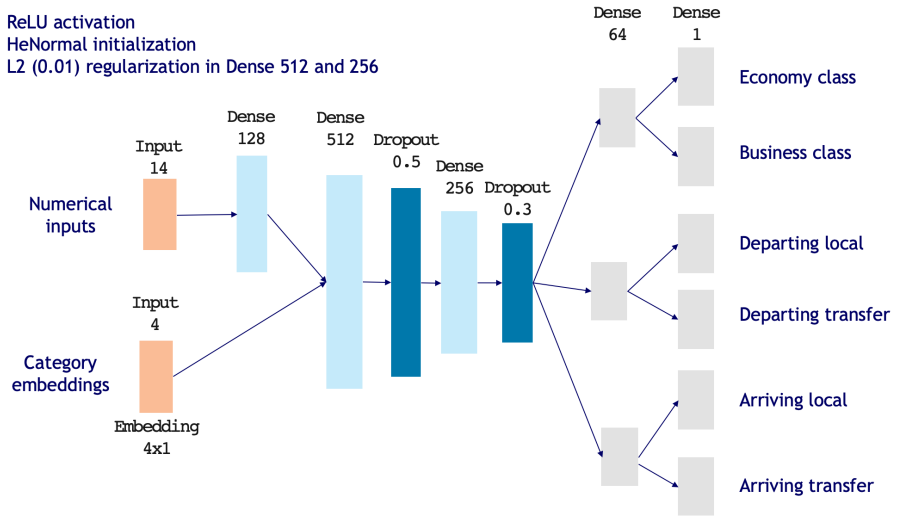


Figure 6.16: Multi-task learning final deep neural network architecture with 3 output groups

Next to the sharing part, an auxiliary task is also experimented with. For the auxiliary task, the total number of bookings for the cabin class is added to the network. The hypothesis is that this would be a related task to the economy and business class passengers and therefore might improve the performance of those target variables. Again, the same three groups are tried with only now the auxiliary task added. The auxiliary tasks where added to the cabin class groups. The results of the experiments can be seen in table:

Table 6.2: Results experiments about output groupings and auxiliary inputs

Target Category		**MTL: No groups**	**MTL: 2 groups**	**MTL: 3 groups**	**MTL-aux: no groups**	**MTL-aux: 2 groups**	**MTL-aux 3 groups**
Economy Class	MAE	12.405	12.413	12.373	12.593	**12.300**	12.981
	st.dev	1.204	1.134	1.509	**1.11**	1.144	1.212
Business Class	MAE	2.883	2.863	2.864	2.809	**2.806**	2.852
	st.dev	0.346	0.221	0.222	0.211	**0.210**	0.243
Departure Transfer	MAE	**8.445**	8.614	8.509	8.673	8.502	8.803
	st.dev	0.894	**0.807**	0.913	0.908	0.902	0.929
Arrival Transfer	MAE	7.934	8.148	**7.917**	7.986	8.142	8.505
	st.dev	0.842	0.889	0.846	**0.714**	0.973	0.779
Departure Local	MAE	**7.756**	8.035	7.933	7.836	7.958	8.064
	st.dev	0.628	0.804	0.796	0.766	**0.621**	0.692
Arrival Local	MAE	6.941	7.191	6.932	**6.930**	7.258	7.469
	st.dev	**0.543**	0.884	0.778	0.791	0.920	0.917
Total	MAE	**46.364**	47.264	46.528	46.827	46.966	48.674
	st.dev	1.941	2.050	2.262	1.955	**1,856**	2.078

First of all, at the total score, hence the last two rows, it can be seen that MLT with no groups performs best in terms of MAE and second in terms of variation. The models are compared with Wilcoxon signed-rank test in order test if there is significant difference. For example, comparing the two best models, MTL with no groups and MTL with 3 groups, on MAE. All target variables show a significant difference with P-values very close to 0.000.

Furthermore, lets discuss the grouping aspect i.e., what to share between the output layers. For the cabin classes there is not much difference between the groupings. Therefore, it can be concluded that grouping economy and business class, does not increase or decrease MAE that much. For the remaining target categories, it seems that creating two groups, decreases performance in terms of MAE. Probably, these tasks benefit more from having the flexibility of an extra own dense layer to learn more task specific features. In this respect, it can be concluded that in general, less sharing of output layers improves performance.

From the experiments it became clear that adding an auxiliary task, did not improve the performance. It was expected that this might improve the performance of both cabin class passengers. For the business class passengers, it did improve performance but for the economy class it only improved by having 2 groups. In terms of variation of the prediction, it improved in all cases except on having 3 groups. Therefore, it can be concluded that for more stable predictions, adding the number of bookings of the cabin classes as auxiliary task improves performance in most cases. However, for the remaining target categories, it decreases performance. Only the MTL with two groups seems to benefit slightly from it. The output of the other models do not improve at all with the auxiliary inputs. To conclude, it is not recommended to add the number of bookings for the cabin classes as auxiliary inputs.

6.4 Conclusions

For predicting different types of airline passengers, it can be stated that MTL works better than STM in four out of the six target variables. For KLM, the performance of the models in terms MAE is the most important. Therefore, it is recommended to use STL for business class and departure transfer passengers, because for those targets, STM performs better. In this cases STL performs better because it utilizes task and non-task-specific features better. For example, we saw for the business class that with MTL the feature economy class capacity has a high feature importance although this is non-task-specific feature. Furthermore, we saw for departure transfer passengers that task-specific feature such arrival and departure station has a higher feature importance, hence this features are better utilized. For the other four target variables, it is recommended to

use MTL because it has an lower MAE score. Furthermore, it was shown that there is a difference in how MTL and STM uses the features. First SHAP theory was used to show the difference in feature importance. Based on these differences, it can be concluded that MTL and STM allocate different importance to the same features, especially to the general features such as date and time features. Subsequently, it was shown that the features are also used differently in making individual predictions. The main difference lays in how the general features such as date and time features are utilized. By plotting the PDP, it was observed that MTL reacts less on changes in feature values. In addition, with the SHAP contribution plot, it can be concluded that MTL is more robust in how its uses and reacts to those general features in comparison with STM and thus has regularization effect. Given that regularization works well for this problem, it might be that for other problems where regularization benefits performance, MTL can outperform STM.

Furthermore, it seems that grouping output layer does not lead to any performance improvement. The same holds for adding the number of bookings of the cabin classes as auxiliary input. This could be further tested with different parameters however, given the current results, it seems unlikely it will result in any significant improvement.

Chapter 7

Validation

In this section, a comparison will be presented between the proposed DNN model with the benchmark models. The proposed DNN is based on the conclusions of previous chapter and represents the best neural network architecture found. For evaluation the models are compared on MAE at a few query moments i.e.: over the year and day of the week.

7.1 Benchmark Models

7.1.1 Current Method

Before continuing, a short notice must be made. At my last day at KLM, I noticed that the current system PTRA was updated. It appeared that at the Air France site, a team had been working on improving their model. This Air France team had ample time and resources at their disposal. For fair comparisons I used their new improved model results for evaluation. Due the lack of time to investigate, it is not clear how the new improved PTRA system works. Therefore, the old system will be explained. With the current model used at KLM (deployed by means of a software named PTRA), a passenger forecast is made from 180 days before departure. Actually, the forecast is not the number of passengers but the number of bookings. It produces a specifc outcome for transfer/local and economy/business.

PTRA uses the following features.

1. Bookings that are already available for future flights
2. Future flight information: location and timestamps
3. Historic bookings of the past 3 years

With this information a bookings curve (BC) is made based on bookings that are now available and historical bookings. On both sets, graph curves are created of the booking patterns indicating the speed at which the flight is filled with bookings. Now, the curve of the historical flights that is most similar to the curve of the target flight is chosen. The notation therefore is:

$$BestBC = (min_{\forall X \in E}) \, \|BC - X\|$$

The remaining bookings are then calculated by:

$$REM_{bookings}(t) = REM_{bookings}(\sum_{i} \frac{BestBC_i}{n})$$

Here i is denoted as the BC and n the days before departure. An example is given in the figure below.

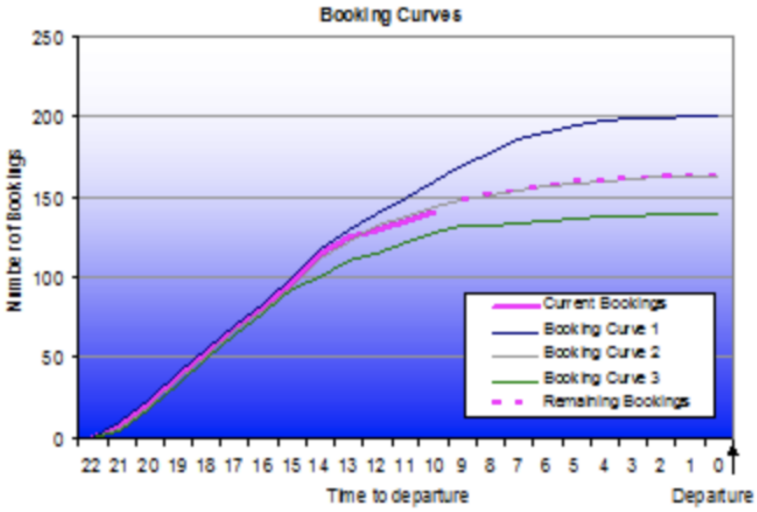


Figure 7.1: Example Current KLM PTRA Method

In the figure the purple line denotes the current bookings made and its expected curve path represented by dots. The three most similar BC found are plotted as well. In this case the gray line is most similar to the current BC, therefore, the purple dotes follow the gray line.

7.1.2 Baseline Model

A baseline model is made to better understand how difficult it is to predict the number of passengers. This baseline model is relative simple and should be easy to understand and simple to apply. The chosen method, is a moving average (MA) of the 3 previous flights taken the query moment into account. For example, if one wants to predict a specific upcoming flight with flight number KL 0001 and a query moment of 150 days before departure. Then one has to look 150 days back and take the average of the last 3 flights with flight number KL 0001.

7.1.3 Best in Class

As the best in class model, gradient boosting decision trees (GBDT) is chosen. First a brief motivation will be given why this is considered as best in class. Subsequently, it will be explained how this model works. Furthermore, there are different ways to implement it in terms of libraries, hyperparameter tuning and how to build multiple models. This will be discussed in the last section.

Motivation

GBDT is a very popular model used in many different settings. In the book XGBoost with Python, 13 algorithms are compared on 165 datasets. Although there is no silver bullet for what is the best algorithm to use, GBDT outperforms any other algorithm in most cases. In addition, it is a model used widely in Kaggle competitions and has won many times [1]. The main advantages of GBDT mentioned by (Hastie and Friedman, 2009), (Kuhn and Johnson, 2013), (Murphy, 2012) and (Strobl and Augustin, 2006) are:

1. Captures non-linear relationships in the data
2. Captures high-order interactions between inputs

[1] This website gives an overview where GBDT was applied and has won a podium place in Kaggle competitions: https://github.com/microsoft/LightGBM/blob/master/examples/README.md

3. Performs implicit variable selection
4. Deals with categorical and numerical data
5. Deals with different scaled input variables
6. Handles missing values
7. Robust to outliers
8. Fast predictions

The disadvantages mentioned are:

1. Requires datasets with a big sample size
2. Can be difficult to interpret
3. Can over-fit the data (predictive performance)
4. Tends to select predictors with a high number of distinct values
5. Training (optimizing hyper-parameters) is time consuming

Although there are some disadvantages, the advantages and evidence for its outstanding performance dominates. Therefore, this is the first advanced model applied, considered as best in class and model to beat with this thesis research work. The evaluation and comparisons of this model will be discussed in chapter 9.

How it works

Gradient boosting is an ensemble technique, hence predictions are executed by an ensemble of simple estimators. In case of GBDT, this estimators are the decision trees, however, in theory, any other prediction model can be used as estimator. Taken this into account, the objective of the gradient boosting is to train an ensemble of estimators, given that that it is known how to train a single estimator. Building this ensemble is called boosting and with boosting it is expected to improve performance in comparison to a single estimator. In addition, there are parameters to optimize for this algorithm. Because training and tuning is very time consuming not all parameters are considered, the one that where included in the optimization for this thesis work are:

1. *n_estimors*: number of boosting iterations i.e., number of trees in the ensemble.
2. *max_depth*: the depth of one tree, smaller values gives weaker learners.
3. *num_leaves*: number of leaves in one tree, this controls the complexity of the model.
4. *colsample_bytree*: fraction of features to use for an iteration.

Implementation

For implementation light-GBM developed by Microsoft is applied. The alternative would be XGBoost but in a comparison, (Guolin Ke, 2017), debates the following advantages of Light-GBM:

1. Faster training speed and higher efficiency: Light-GBM uses an histogram based algorithm i.e it buckets continuous feature values into discrete bins which fastens the training procedure.
2. Lower memory usage: Replaces continuous values to discrete bins which result in lower memory usage.

3. Better accuracy than any other boosting algorithm: It produces many more complex trees by following leaf wise split approach rather than a level-wise approach which is the main factor in achieving higher accuracy. However, it can sometimes lead to overfitting, which however can be avoided by setting the max depth parameter.

4. Compatibility with Large Datasets: It is capable of performing equally good on large datasets with a significant reduction in training time as compared to XGBOOST.

For optimization the hyper parameters three possibilities are considered i.e. Bayesian optimization, grid search and random search. For this case random search is used, because it is less biased compared to grid search. With grid search, one has to select the parameters to choose out, which creates bias. With Random search a range of possible options is given, from which a random selection takes place. Bayesian optimization could also be a choice, but it is not proven that random search or Bayesian is better to one or another. The trade-off depends on the type of problem and resources available. In this case random search is chosen for its simplicity. In practise, 500 different combinations with random search are used, with a cross-fold validation of 3. However, in future research it might be interesting to investigate Bayesian optimization as well.

The results of the best found parameter setting are:

```
MultiOutputRegressor(estimator=LGBMRegressor(boosting_type='gbdt',
                                             class_weight=None,
                                             colsample_bytree=1.0,
                                             importance_type='split',
                                             learning_rate=0.1, max_depth=-1,
                                             min_child_samples=20,
                                             min_child_weight=0.001,
                                             min_split_gain=0.0,
                                             n_estimators=500, n_jobs=-1,
                                             num_leaves=31, objective=None,
                                             random_state=None, reg_alpha=0.0,
                                             reg_lambda=0.0, silent=True,
                                             subsample=1.0,
                                             subsample_for_bin=200000,
                                             subsample_freq=0),
                     n_jobs=None)
```

Figure 7.2: Results Random Search Hyper-Parameters Light-GBM

Furthermore, since there are multiple outputs required, a model per target varible must be made. This is done by using the Multiple Output Regressor from Sklearn library.

7.2 Results

7.2.1 Comparison on Query Moments

From figure 7.3 till figure 7.8, the results of MAE are plotted over the query moments. There are 4 lines belonging to 4 different models. As discussed there are 3 baseline models. The simple model, denoted as the Moving Average (MA). The current system of KLM, denoted as PTRA. And the best in class model, denoted as Light-GBM. The first thing to notice is that the MA performs a lot worse. This implies that it is not so easy to predict. Second, it stands out that PTRA predicts on shorter time horizon, thus less query moments (maximum 180 days before departure). Perhaps for Air France this is not a requirement, but for KLM it is necessary to have predictions longer than 180 days before departure. Third, the MTL and Light-GBM perform better than PTRA on almost all targets. Only on the economy class passengers, PTRA sometimes performs better. It is difficult to reason why this is, because it unclear how the new PTRA model works. But assuming the previous model relies heavily on the current status of bookings, it can be seen that closer to departure PTRA comes closer in MAE. Perhaps, it utilizes this feature in a better way by interpolating the bookings curve. In addition, the PTRA predictions are more unstable and vary over the query moments. This was also the experience of the business, where the predictions of PTRA appeared to be unreliable. The ML models show a stable improvement in MAE when

the time to departure gets closer. Furthermore, the Light-GBM and DNN are very close to each other. Overall on average DNN performs better on all targets compared to Light-GBM. Only with the departure local passenger, Light-GBM performs slightly better on query moment 360 till 220 days before departure. Besides that DNN performs better than light-GBM, there are also practical benefits. The DNN is one model and Light-GBM consists of six models. One model is less effort to put in production, cost less maintenance and requires less time with monitoring. In addition, for auto-retraining, only one model needs to be tuned instead of six models separately. In conclusion, based on these charts and practicalities, it can be stated that the proposed DNN model outperforms all benchmark models.

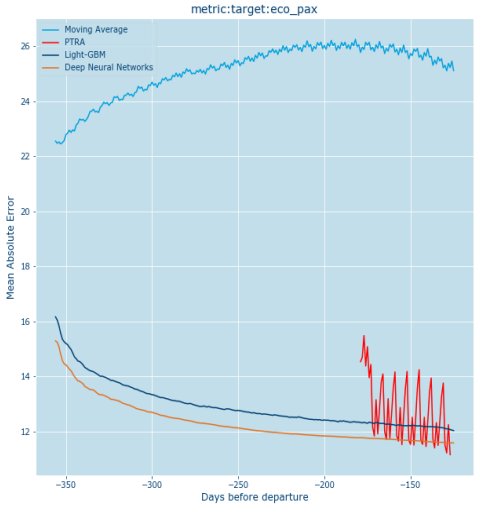


Figure 7.3: Query moments, economy class

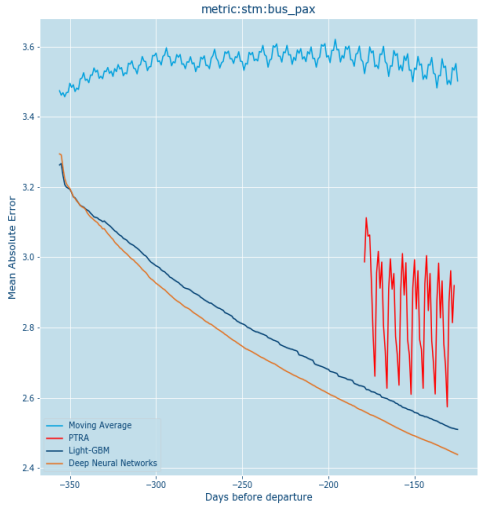


Figure 7.4: Query moments, business class

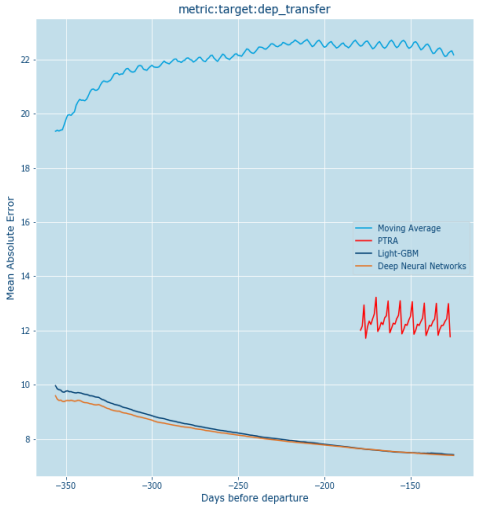


Figure 7.5: Query moments, departure transfer

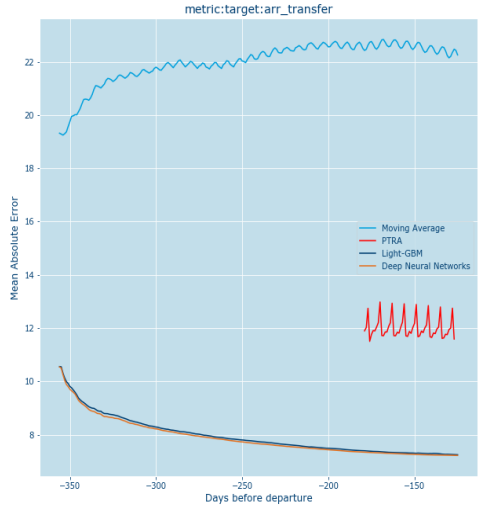


Figure 7.6: Query moments, arrival transfer

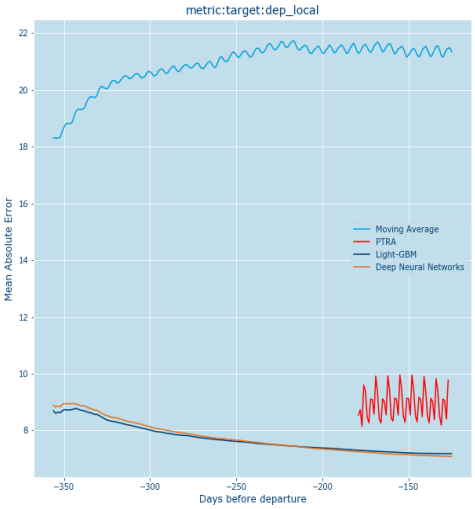


Figure 7.7: Query moments, departure local

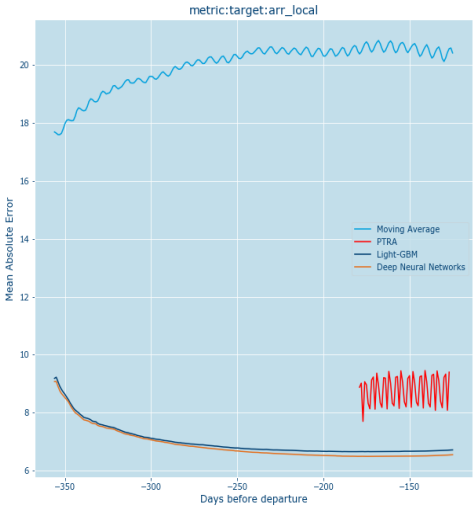


Figure 7.8: Query moments, arrival local

7.2.2 Comparison over the Year

In figure 7.9 till figure 7.14, the average MAE performance of the four models can be seen over the year on the test set. Where month 1 is January and month 12 is December. Based on the previous section, as expected, the MA performs a lot worse. This again indicates, it is not so simple to predict different type of airline passengers. In case of the local and transfer passengers targets, the ML models outperform PTRA on every aspect. Especially, on the transfer passenger, where the gap is relatively bigger. For the cabin class passengers, PTRA performs only better on economy class in March than the proposed DNN. This is in the beginning of the year, where the number of passengers is overall on its lowest. Interestingly, in the most busiest months; June, July and August, the gap between PTRA and ML models becomes larger. This are the most important months, because the demand is then the highest and thus, more work must be done. The more work there is, the more costs can be reduced and/or the more customers can be serviced, all with better planning.

Comparing DNN with light-GBM, it can be stated that the behaviour is very similar. The patterns over the month follow more or less the same movements and the MAE are very close to each other. Almost in all cases DNN matches or outperforms light-GBM. Only for departure local passenger in May, light-GBM clearly performs better. Especially for the cabin classes DNN shows better performances during the year. This strengthens the argument that the proposed DNN is a better choice for predicting different type of airline passengers compared to the benchmark models.

Figure 7.9: Over the year, economy class

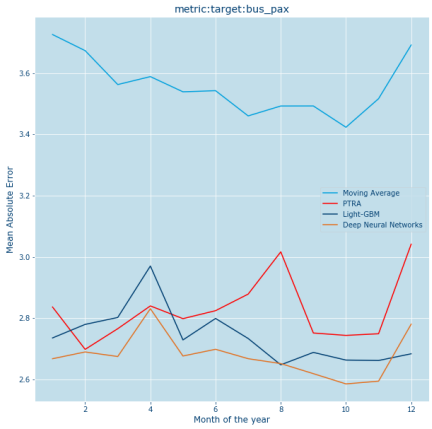


Figure 7.10: Over the year, business class

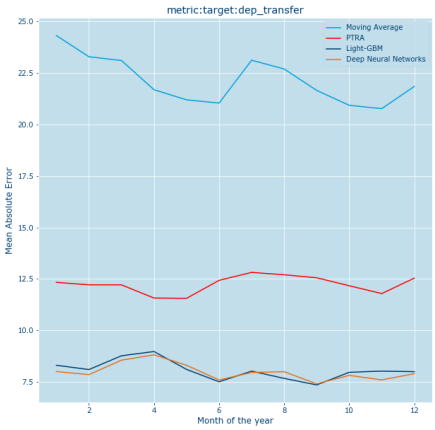


Figure 7.11: Over the year, departure transfer

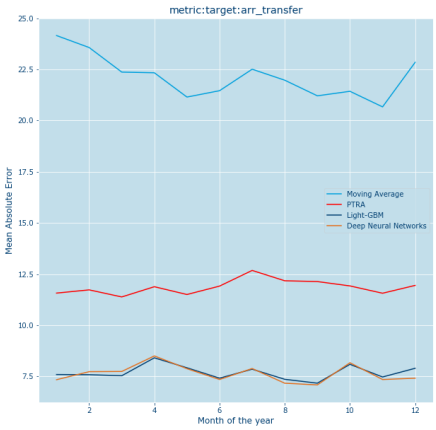


Figure 7.12: Over the year, arrival transfer

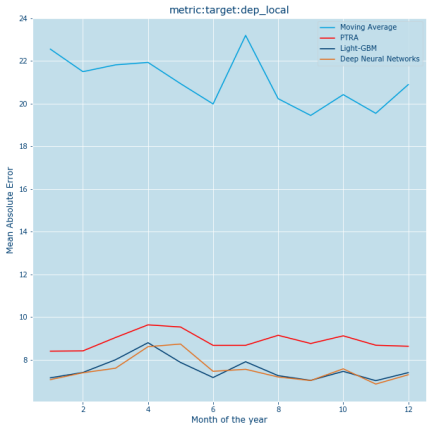


Figure 7.13: Over the year departure local

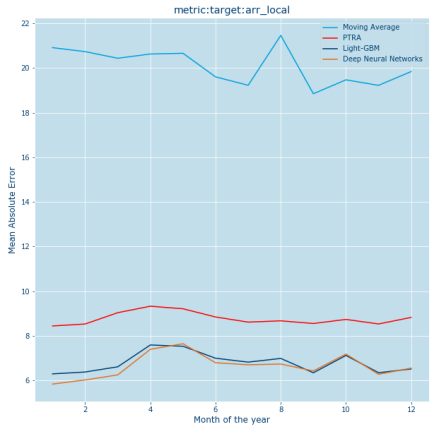


Figure 7.14: Over the year, arrival local

7.2.3 Comparison over Day of the Week

In figure 7.15 till figure 7.20, the average MAE performance over day of the week of the test set is shown. Where 0 is a Sunday and 6 is a Saturday. Again, as expected, the MA performs a lot worse than the other models. On almost every aspect the ML models perform better than PTRA. Only for the business class passengers, on Monday, PTRA works better. Light-GBM and DNN are again close to each other in terms of performance and patterns. Overall, DNN performs better than Light-GBM, especially on for the cabin classes.

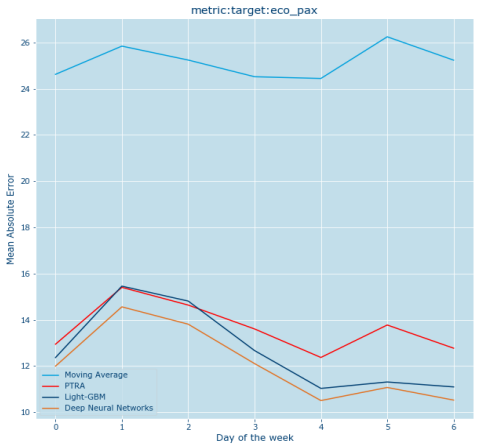


Figure 7.15: Day of the week, economy class

Figure 7.16: Day of the week, business class

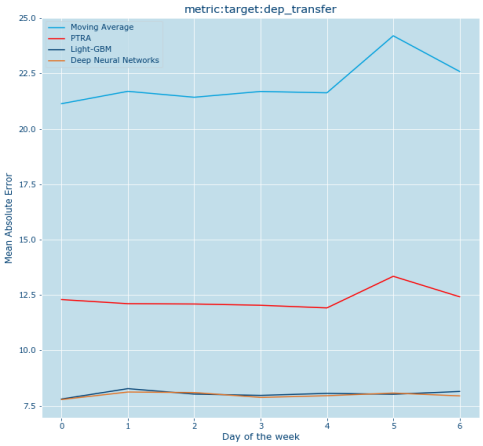


Figure 7.17: Day of the week, departure transfer

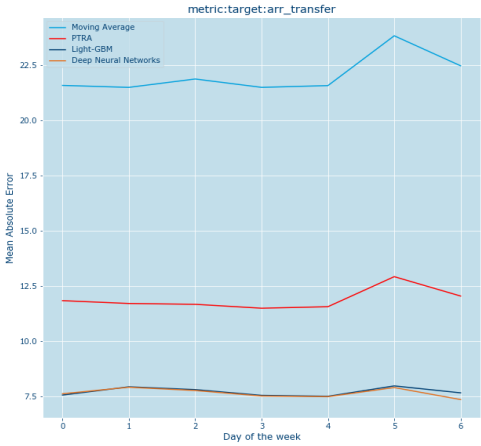


Figure 7.18: Day of the week, arrival transfer

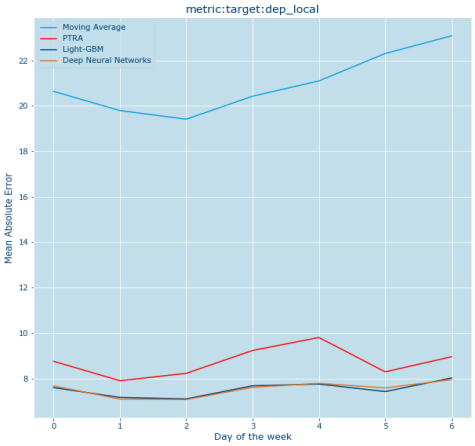


Figure 7.19: Day of the week, departure local

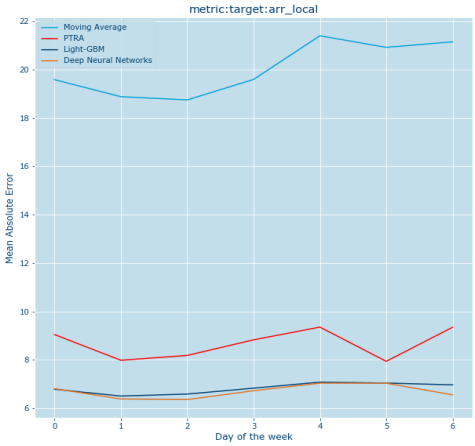


Figure 7.20: Day of the week, arrival local

7.3 Conclusions

In this chapter a comparison was made between the proposed DNN and the benchmark models. In general, the simple MA model showed that is not that easy to predict the different type of airline passengers. The gap between this MA and the more advanced methods is in all cases relative big. Furthermore, it can be concluded that the ML models outperform the current PTRA system used at KLM. The ML models predict on more query moments and have shown a better performance in terms of MAE. Lastly, the proposed DNN performs slightly better than the best in class ML model. The expectation was that this best in class model would be difficult to beat, however based on the results and practical benefits, it can be concluded that DNN is a better choice for this problem. To repeat these practical benefits, the DNN is one model and Light-GBM consists of six models. One model is less effort to put in production, cost less maintenance and requires less time with monitoring. In addition, for auto-retraining, one model needs to be tuned instead of six models separately. To summarize, table 7.1 presents an overview of the results. In this table the predictions are grouped by query moments and then the average MAE and standard deviations are calculated.

Table 7.1: Results Deep Neural Network compared to Benchmark Models

Target Category		**Moving Average**	**PTRA**	**Light-GBM**	**Deep Neural Network**
Economy Class	MAE	25.135	12.763	12.760	**12.405**
	st.dev	**0.951**	1.159	1.308	1.204
Business Class	MAE	3.552	2.854	2.811	**2.765**
	st. dev	**0.032**	0.137	0.202	0.221
Departure Transfer	MAE	21.994	12.321	8.363	**8.240**
	st.dev	0.762	**0.375**	0.832	0.790
Arrival Transfer	MAE	21.992	12.012	7.982	**7.934**
	st.dev	0.767	**0.397**	0.891	0.842
Departure Local	MAE	20.892	8.950	**7.696**	7.756
	st.dev	0.789	**0.545**	0.661	0.628
Arrival Local	MAE	19.977	8.789	7.045	**6.941**
	st.dev	0.770	**0.525**	0.877	0.543
Total	MAE	113.542	57.694	46.854	**46.041**
	st.dev	1.813	**1.494**	2.376	1.905

Looking at the total scores of the losses, it can be concluded that the proposed DNN performs almost 60% better than the simple MA. Although, the variation of the MA is more stable, it does not compete with the other benchmark models. Furthermore, one of the objectives of this master thesis project was to develop a new airline passenger forecasting system that performs better than current system used at KLM. Despite Air France themselves all of a sudden came with an improved system during the last day of this research at KLM, DNN still performs over 20% better. Comparing the variation is not fair taken the fewer number of query moments PTRA predicts on into account. Lastly, one of the hardest challenges was to outperform the light-GBM. The differences are small, but the proposed DNN performs in total 2% better. To compare the light-GBM with the proposed model a Wilcoxon signed-rank test is applied. On all target variables on MAE it shows a significant difference. On each target variable the P-value is 0.000. In addition, it shows more stable predictions evidenced by mostly a lower standard deviation.

Chapter 8

Discussion

This chapter will discuss the results relative to the stated research questions for this master thesis. In addition possible threats in terms of validity of this research will be discussed in combination with possible directions for follow-up research work.

8.1 Discussion of the Results

In chapter 1 section 2, the main research question of this master thesis was introduced. This was formulated as follows:

How to build a forecasting model based on historical airline data to predict the number of passengers on flights departing from a specific airport in specific flight categories? Specifically, local and transfer passengers as well as different cabin classes with a time horizon of 125-361 days prior to departure.

To answer this main question, it is split up in sub-research questions, which are answered throughout the chapters of this master thesis. In this section the answers found will be summarized and discussed.

What methods do already exist that concerns passenger forecasting on flight level? Passenger forecasting has been researched in different manners. Basically, there are two approaches in the literature, an econometric time series approach and a ML approach. Some forecast on time series levels such as daily, monthly or yearly time horizons. Where others use ML to predict on flight level the number of passengers or load factors. For the specific problem of KLM, where they want to predict multiple type of passengers for upcoming specific flights, there is no equivalent case in the literature. Furthermore, an econometric time series approach seems not the best option. In case of KLM, this would imply that around 1200 different models need to be made, one for each flight number. ML has shown relative good performance on regression problems. It can capture all flights in one model, learn non-linear relationships and can handle large amount of data.

Which features can be selected and, if required, be constructed or transformed, to build the prediction model?
For the prediction model, only inputs can be used that are available at the required moments. Moments for predictions, are also referred to as query moments. Data that is available on the required query moments are:

- Current state of bookings for the cabin classes
- Locations: arrival and departure station
- Aircraft type

- Aircraft capacity for the cabin classes
- Scheduled date-time information for departure and arrival
- Flight number

All, except the bookings data, should be available for the airline and are logged in the Official Aviation Guide (OAG). This guide provides a track record for over 900 airlines. Taken into account that each airline knows it own state of bookings, it can be assumed that all airlines have these input data available. Therefore, the proposed model can be generalized to other airlines as well.

There are a lot of possibilities for transforming or constructing features. However, for this thesis it is kept simple because the focus is on the modelling part and not on finding new features. In this study, only the timestamp features are used to extract new features.

What baseline can be used to compare the model performance with?

For this master thesis four baseline models are considered and compared with the new proposed model. First a simple MA model is applied that is easy to understand, fast to implement and demonstrates how difficult it is to predict the target variables. The MA applied is based on the 3 previous flights, taken the query moments into account. For example, if one wants to predict a specific upcoming flight with flight number KL 0001 and a query moment of 150 days before departure. Then one looks 150 days back and takes the average of the last 3 flights with flight number KL 0001.

The second baseline model is current airline passenger forecasting system used at KLM. One of the objectives of this thesis project was to outperform this system, therefore, this is considered to be a benchmark model.

Third, a so-called best in class model is considered. For this type Gradient Boosting Decision Trees (GBDT) are chosen. GBDT is a very popular model used in many different settings. In the book XGBoost with Python, 13 algorithms are compared on 165 datasets. Although there is no silver bullet for what is the best algorithm to use, GBDT outperforms in most cases any other algorithm. In addition, it is a model used widely in Kaggle competitions and has won many times.

Fourth, the MTL model is compared with an independent single-task neural network model. For this model the exact same architecture is used: only instead of having six outputs there is one output. In this way a comparison can give insights between the differences of STL and MTL.

Which machine learning techniques should be used to build a prediction model that produces the best results?

For this project the aim is to predict multiple splits in outputs at the same time, hence multiple type of passengers at the same time. These type of passengers share many of the same features and therefore MTL suits this problem by leveraging shared learning between the different types of passengers. The technique used to apply MTL is DNN. It was shown that for 4 of the 6 target variables the performance improved by leveraging shared learning with MTL. Only business class and departure transfer passenger predictions do not benefit from shared learning. Furthermore, looking at the total averages of the target variables MAE, by grouping the performance on query moments, the results have shown that MTL with DNN performed 60% better than the MA model, 20% better than current system used at KLM and 2% better than the best in class model. In this respect, MTL with DNN has proven to be a well suited technique to produce the best results for predicting the different types of airline passengers.

How can the prediction model be evaluated and tested?

The data available cover the years 2016 to 2019. In the airline section there is a clear seasonality. Therefore, a split in training and test data should comprise all seasons in order to capture the different patterns. For this reason it is chosen to take the last year, the most representative one, as the test data. The remaining data will be part of the training set. In total the data contains about 144,195,921 records. Therefore, according to (Raschka, 2018), a 2-way holdout method (train/test split) should be sufficient for performance estimation for the models. On the test set,

the different models are evaluated based on MAE. With MAE, there is no preference for over-under estimation and it is not necessary to penalize bigger errors. And this is what should be avoided, as by penalizing bigger errors, flights with larger capacity will probably get favoured since their absolute difference will be larger. This performance metric is evaluated for all models on query moments, performance of the year, day of the week and on residuals and variations.

8.2 Practical Implication

This section will discuss how KLM can use the improved forecasting system for their business processes. As discussed in chapter 4, the main users are the tactical planners of KLM. A summary of the planning process can be mapped as follow:

Figure 8.1: Planning process end user airline passenger forecasting system

The predictions of the different types of passengers, together with the flight schedule, serve as input to create a long-term planning. This long-term planning is used for resource planning and feasibility analysis. Later on, more detailed plannings are prepared where decisions are made about which job to assign to which employee etc. In terms of practical implications, it is recommended to do a shadow run on the passenger predictions. Hence, the current system PTRA is still used for the planning process, only the proposed MTL model is deployed and monitored in parallel for comparison. If the MTL works well and performs better, KLM should consider to replace the system PTRA by the proposed model. This limits the risks of practical errors. If for example problems are encountered in the deployment phase or the model does not generalize well, this could be tackled without hurting the planning process. Furthermore, the proposed model predicts the same type of passengers as PTRA, thus in practise the planners do not have to change the process. In practise they should not notice any differences if the model is replaced, hopefully, only that the predictions are better than before.

8.3 Threats to validity and directions of future work

In terms validity a few assumptions were made. First of all, the models were tested on the data of 2019. Ideally, a more robust evaluation procedure should be applied. To be more confident about the results, a rolling window method could be considered. Thus for example, first use 2016 as training data and then 2017 as test data. Then use 2016 and 2017 as training data and 2018 as test etc. Then for all test cases, the average can be taken to get a more robust result. Because of the huge amount of data used in this thesis, this procedure was unfortunately out of scope as it would have resulted in very long training and testing times. And, due to the time constraints and shortage of resources for this project, this would have limited the number of experiments to try.

Furthermore, it was assumed that GBDT is the best in class. However, this could not be the case for this problem since there is currently no superior prediction model that always outperforms each other model. To be more confident about which model is best in class, a study could be done that evaluates many more prediction models for this problem. In terms of directions for future work of the proposed MTL DNN model, it is still an open question about what to share in the architecture. In current MTL related literature, there is not a uniform definition about which tasks are related, hence which tasks should share which parameters. This thesis limited the number of possibilities in what to share and this could be extended in many different ways. For now only the output layers where considered in terms of sharing but all the layers before the output layer could also be tested and evaluated. It could also be examined which inputs to share, since not all inputs might be relevant for each target variable.

Besides what to share in the network, there is also room for improvement by hyper parameter optimization. For example, the learning rate is one of the most important parameters but is not optimized in this thesis. Other parameters such as batch size, number of nodes for the layers, activation functions, drop out rates, number of layers etc. are not fully explored and may offer possibilities of improvement.

Furthermore, in this case all target variables were considered equally important in defining the objective loss function. It could be that by optimizing the weight distribution among these target variables, the overall performance will improve. Different techniques might be possible here to find the optimal weights distribution, like a grid search, random search and other methods.

Lastly, this thesis did not focus much on finding new features for the prediction model. To improve the prediction models, it could be considered to explore new features by feature engineering or by adding complete new features. These new features could be marco-economic information or perhaps meta-search data on arrival and destination combinations. It could also be interesting to include information of other airlines such as difference in ticket prices or number of flights going to the same destination.

All in all, predicting airline passengers is a very interesting and challenging topic and is studied for many years now. With this thesis I hope to have augmented this research topic by demonstrating that predicting different types of passengers more accurately, can add more business value. I also showed MTL adds a new perspective to approach this problem and comprises many more opportunities to further improve.

Bibliography

Andreoni, A. and Postorino, M. N. (2006). A Multivariate Arima Model To Forecast Air Transport Demand. *European Transport Conference*, (January 2006):14. 6

Arik, S., Chrzanowski, M., Coates, A., Diamos, G., Gibiansky, A., Kang, Y., Li, X., Miller, J., Ng, A., Raiman, J., Sengupta, S., and Shoeybi, M. (2017). Deep voice: Real-time neural text-to-speech. *34th International Conference on Machine Learning, ICML 2017*, 1(February 2017):264–273. 10

BaFail, A. O. (2004). Applying data mining techniques to forecast number of airline passengers in saudi arabia (domestic and international travels). *Journal of Air Transportation*, 9(1):100–115. 7

Baker, B, M. (2018). Airlines Turn to AI as They Up IT Spending. *Business Travel News*, 1(1):1–21. 5

Baxter, J. (2000). A model of inductive bias learning. *ournal of Artificial Intelligence Research*. 10

Bilen, H. and Vedaldi, A. (2016). Integrated perception with recurrent multi-task neural networks. *Advances in Neural Information Processing Systems*, (Nips):235–243. 10

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics 31 (3), 307–327*. 6

Box, G. and Jenkins, G. (1970). Time series analysis, forecasting and control. *Holden-Day*. 6

Caruana, R. (1993). Multitask learning: A knowledge-based source of inductive bias. *Proceedings of the Tenth International Conference on Machine Learning*. 9

Caruana, R. (1997). Multitask Learning. 75:41–75. 8, 10

Dai, J., He, K., and Sun, J. (2016). Instance-Aware Semantic Segmentation via Multi-task Network Cascades. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:3150–3158. 10

Delta, A. L. (2017). Investor Day, 2017. *Delta Airlines Investor Day*. 5

Duong, L., Cohn, T., Bird, S., and Cook, P. (2015). Low Resource Dependency Parsing : Cross-lingual Parameter Sharing in a Neural Network Parser. pages 845–850. 10

Dvornik, N., Shmelkov, K., Mairal, J., and Schmid, C. (2017). BlitzNet: A Real-Time Deep Network for Scene Understanding. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-October:4174–4182. 10

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica 50 (4), 987–1007*. 6

Fedyk, A. (2016). How to Tell if Machine Learning Can Solve Your Business Problem. *Harvard Business Review*. 5

Fildes, R., Wei, Y., and Ismail, S. (2011). Evaluating the forecasting performance of econometric models of air passenger traffic flows using multiple error measures. *International Journal of Forecasting 27(3), 902-922*. 6

Gao, G., Bao, Z., Cao, J., Qin, A. K., Sellis, T., Fellow, IEEE, and Wu, Z. (2019). Location-Centered House Price Prediction: A Multi-Task Learning Approach. pages 1–14. 7

Godfrey, L. B. and Gashler, M. S. (2018). Neural decomposition of time-series data for effective generalization. *IEEE Transactions on Neural Networks and Learning Systems*, 29(7):2973–2985. 7

Guolin Ke, Qi Meng, T. F. T. W. W. C. W. M. Q. Y. T.-Y. L. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*. 51

Hastie, T., T. R. and Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction, second edition. *Springer Series in Statistics*. 50

He, K. (2015). Delving Deep into Rectifiers : Surpassing Human-Level Performance on ImageNet Classification. 36

Kendall, Alex, R., Gal, Y., and Cipolla (2018). Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 7482–7491. 10

KLM (2018). KLM Annual Report 2018. pages 1–216. 1, 25

Kokkinos, I. (2017). UberNet: Training a universal convolutional neural network for Low-, Mid-, and high-level vision using diverse datasets and limited memory. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-January:5454–5463. 10

Kuhn, M. and Johnson, K. (2013). Applied predictive modeling. *Springer New York*. 50

Laik, M. N., Choy, M., and Sen, P. (2014). Predicting airline passenger load: A case study. *Proceedings - 16th IEEE Conference on Business Informatics, CBI 2014*, 1:33–38. 7

Lawhead. R.J, G. A. (2019). A bounded actor–critic reinforcement learning algorithm applied to airline revenue management. *Engineering Applications of Artificial Intelligence*, pages 252–262. 6

Leeuwen, H. V., Zhang, Y., Zervanou, K., Mullick, S., Kaymak, U., and Ruijter, T. D. (2020). Lost and Found : Predicting Airline Baggage At-risk of Being Mishandled. 5

Liu, X., Gao, J., He, X., Deng, L., Duh, K., and Wang, Y. Y. (2015). Representation learning using multi-task deep neural networks for semantic classification and information retrieval. *NAACL HLT 2015 - 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, (January):912–921. 10

Lufthansa, G. (2019). Lufthansa Group Invests in Artificial Intelligence Partnership with Hopper. *Newsroom, Lufthansa Group*. 5

Lundberg, S. M. and Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 2017-December(December):4766–4775. 39

Mohie El-Din, M. M., Farag, M. S., and Abouzeid, A. A. (2017). Airline Passenger Forecasting in EGYPT (Domestic and International). *International Journal of Computer Applications*, 165(6):975–8887. 7

Mostafaeipour, A., Goli, A., and Qolipour, M. (2018). Prediction of air travel demand using a hybrid artificial neural network (ANN) with Bat and Firefly algorithms: a case study. *Journal of Supercomputing*, 74(10):5461–5484. 7

Murphy, K. P. (2012). A probabilistic perspective. *The MIT Press*. 50

Nekrasov, V., Dharmasiri, T., Spek, A., Drummond, T., Shen, C., and Reid, I. (2019). Real-time joint semantic segmentation and depth estimation using asymmetric annotations. *Proceedings - IEEE International Conference on Robotics and Automation*, 2019-May:7101–7107. 7, 10

Peffers, K. Tuunanen, T. (2007). A design science research methodology for information systems research. *Journal of management information systems*. 12

Pentina, A. and Lampert, C. H. (2017). Multi-task learning with labeled and unlabeled tasks. *34th International Conference on Machine Learning, ICML 2017*, 6:4292–4309. 10

Raschka, S. (2018). Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. 15, 33, 60

Ruder, S. (2017). An Overview of Multi-Task Learning in Deep Neural Networks arXiv : 1706 . 05098v1 [cs . LG] 15 Jun 2017. (May). 7, 8, 11

Shihab, S. A. M., Logemann, C., Thomas, D.-G., and Wei, P. (2019). Autonomous Airline Revenue Management: A Deep Reinforcement Learning Approach to Seat Inventory Control and Overbooking. *arXiv*. 6

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through propagating activation differences. *34th International Conference on Machine Learning, ICML 2017*, 7:4844–4866. 39

Sinno J., Pan, Q. Y. (2010). A survey on transfer learning. *Transactions on knowledge and data engineering*. 8

Song, S. and Xiao, J. (2014). Sliding Shapes for 3D Object Detection in Depth Images (Supplemental material). *Eccv*, (August):1–8. 10

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958. 37

Strobl, C., l. B. A. and Augustin, T. (2006). Unbiased split selection for classification trees based on the gini index. *Technical report*. 50

Tessler, C., Givony, S., Zahavy, T., Mankowitz, D. J., and Mannor, S. (2017). A deep hierarchical approach to lifelong learning in minecraft. *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pages 1553–1561. 10

Tsui, W., Balli, H., Gilbey, A., and Gow, H. (2014). Forecasting of hong kong airport's passenger throughput. *Tourism Management 42, (62-76*. 6, 7

Van Aken, J. (2005). Management research as a design science: Articulating the research products of mode 2 knowledge production in management. *British journal of management*. 12

Wotawa, F., Friedrich, G., Koitz-hristov, I. P. R., Eds, M. A., and Goebel, R. (2019). *Advances and Trends in Artificial Intelligence From Theory to Practice Series Editors*, volume 2. Springer International Publishing. 5

Xue, Y., L. X. C. L. (2007). Multi-task learning for classification with dirichlet process priors. *ournal of Machine Learning Research*. 10

Yang, Y. and Hospedales, T. M. (2017). T RACE N ORM R EGULARISED D EEP M ULTI -T ASK. (2014):2015–2018. 9

Zhang, Y. and Yang, Q. (2017). A Survey on Multi-Task Learning. pages 1–20. 8