NOVA IMS Information Management School

MAA

Mestrado em Métodos Analíticos Avançados
Master Program in Advanced Analytics

Forecasting sales and transactions of fast-food stores

A Proof of Concept

Cristina Isabel Palma Mousinho

Internship report presented as partial requirement for obtaining the Master's degree in Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

FORECASTING SALES AND TRANSACTIONS OF FAST-FOOD STORES: A PROOF OF CONCEPT

by

Cristina Isabel Palma Mousinho

Internship report presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics

Advisor: Mauro Castelli

External Advisor*:* Pedro Freitas Lopes

August 2021

DEDICATION

To all the teachers and professors that led me to become who I am today.

ACKNOWLEDGEMENTS

This report would not be possible if two people did not believe in me: my external supervisor, Pedro Freitas Lopes, and my internal supervisor, Mauro Castelli.

Thank you, Pedro, for taking your chance on me. For listening to my ideas as you did to any other teammate. Together with everyone else on our team, you taught me there is no weakest link. Each one of us has strengths and downfalls. That is why we are a team: to give what we have, and to receive what we need. I am finishing my internship feeling like a part of the Noesis family, especially a part of our little family. Thank you again, for taking me in, and for putting me as a person above me as an employee.

And none of this would be possible without Mauro. I must thank you for your excitement and availability, never making me wait. Thank you for your hard work, for accepting me as an advisee, and for all you taught me, both inside and outside of classes.

I must also express my deepest gratitude to those who, in addition to Pedro Lopes, worked by my side during the time I was an intern at Noesis: André Castro, Fábio Paixão, Carolina Carvalho, Ana Silva, Diogo Reis, Pedro Santos, and Ana Carolina Carrilho.

And finally, to all those who listen to me complain, but never complain about listening to me: Mom, Dad, Guilherme, Jack, Rita, Milene, Abdallah, Ravi, Campos, Escária, Ricardo, Paulo and Paula, a big, big thank you.

ABSTRACT

As time goes on, more and more clients look for solutions to their data-related problems. During a 9-month internship at the Portuguese consulting company Noesis, a request was presented by a customer that wished to improve the forecasting capabilities of their fast-food chain, on sales and transactions, for four different distribution channels, and globally. Following a data analytics approach, hundreds of time series were examined, external variables were added, and two algorithms were used - ARIMA and Facebook's Prophet. Both models were evaluated, and as each of them performed better in different segments, a hybrid system was implemented, successfully completing the task at hand. Based on the results, future improvements and recommendations were also identified.

KEYWORDS

Machine Learning; Forecasting demand; Time series; ARIMA; Facebook Prophet.

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ACF	Auto Correlation Function
AR	Autoregressive
ARIMA	Autoregressive Integrated Moving Average
COVID-19	Coronavirus disease of 2019
DAAI	Data Analytics and Artificial Intelligence
EMA	Exponential Moving Average
GRP	Gross Rating Point
I	Integrated
ID	Identification
MA	Moving Average
MAD	Mean Absolute Deviation
MAPE	Mean Absolute Percentage Error
MSE	Mean Squared Error
PACF	Partial Auto Correlation Function
PoC	Proof of Concept
SES	Simple Exponential Smoothing
SMA	Simple Moving Average
sMAPE	Symmetric Mean Absolute Percentage Error
TadGAN	Time Series Anomaly Detection using Generative Adversarial Networks
WMA	Weighted Moving Average

1. INTRODUCTION

1.1. NOESIS AS A COMPANY

Noesis began as a Portuguese tech consulting company, in December of 1995. 17 years later, in 2012, it became international, by opening its first office outside the country, in Brazil. Today, Noesis works alongside well-known Portuguese companies, such as Sonae MC, Fidelidade, Vodafone Portugal, and many others.

To ensure its clients see their needs satisfied, Noesis counts on its over 2000 employees. Although they all work together, different employees belong to different teams (that can be seen in Figure 1.1), according to their specialties.

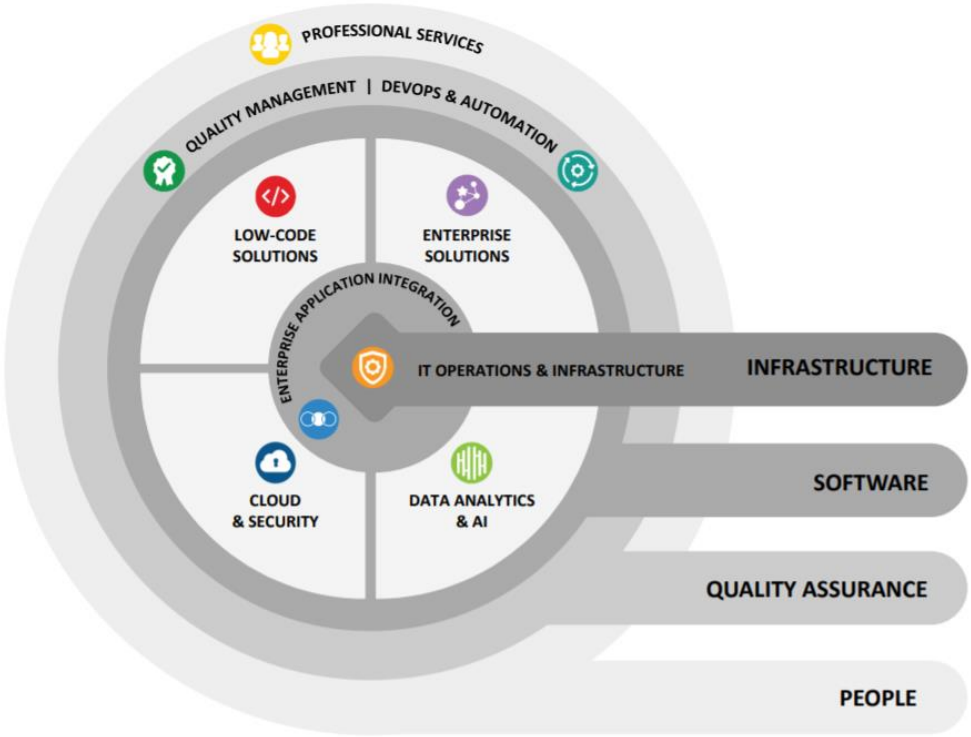


Figure 1.1 – Noesis' teams

The Data Analytics and Artificial Intelligence team, which integrated the internship this report discusses in 2020, was first created in 2010. Today, the team can be thought of as a combination of two components: Business Intelligence and Data Science. The first works with Business Intelligence related tools, such as Qlik and Power BI. The second, that accommodated this 9-month internship, is known for working with Data Science and Engineering associated tools, such as R, Python, Microsoft Azure, StreamSets, etc.

1.1.1. Internship overview

The internship lasted 9 months, from December of 2020 until September of 2021. It began with a presentation about the company and a transition period for learning how the team was organized.

New software was introduced, mostly business-related, like JIRA and GitLab (where all code is stored and organized). The team introduced previous projects, and it became clear from the beginning how much companies intend to develop their forecasting systems.

The position included the opportunity to learn more about Microsoft Azure, by studying for the DP-900: Microsoft Azure Data Fundamentals exam, which was completed successfully on its first try.

After enough knowledge was obtained, the internship began integrating different projects. The first being a meal forecasting Proof of Concept. Though similar to the one this report focuses on, it was for a different company. Unfortunately, the data provided by the client was insufficient to build a successful algorithm. Afterwards, another PoC was proposed - the one that is explained in detail in this report. After finishing that PoC, new steps were taken together with the client, and forecasts continued to be made and fed into a Qlik Sense dashboard for the company's use. At the same time, a third PoC began, this time related to credit-risk analysis and customer classification.

Today, Noesis continues to work on both projects, side-by-side with each of the correspondent companies.

1.2. THE PROJECT

In March of 2021, Noesis was challenged by a worldwide fast-food chain to develop an algorithm capable of predicting the number of transactions in a Portuguese store. This franchise was already a client of DAAI, but for Business Intelligence related reasons. Because of that, Noesis had easy access to the data, and knowledge within its company.

The interaction started with a focus on a particular Portuguese store that was considered to have interesting consumption behavior. Because sufficient results were obtained, the client agreed to move on to a proof of concept, with a goal of expanding the project to all stores in Lisbon and Greater Lisbon and expecting Noesis to build forecasting models to predict sales (in euros) and transactions (in units).

2. LITERATURE REVIEW

2.1. INTRODUCTION

Nowadays, more companies are looking to forecast their demand, to adapt their management decisions to the expected future demand. Forecasting techniques can be divided into three big categories: (1) judgment, (2) causal, and (3) time series methods (Krajewski et al., 2015). In this report study case, the company had been using mainly judgment methods. The employee assigned to the task of predicting sales and transactions had gained their knowledge through experience, knowing which factors influenced customer demand. For example, it was previously known that June 1st, 'Children's Day' in Portugal, was a holiday that made sales spike in almost every store. Though these estimates were fairly successful, the franchising has more than 100 different stores. This makes the task a lot harder, as every store will have its behavior, having to be analysed separately. The client requested that Noesis developed a forecasting technique using time series methods, which can be expanded to all stores without the need of looking at their data one by one. As always, the client had to become aware that sometimes the effort of improving a model might be too big in comparison to the expected improvement. It was important to make them aware of such a trade-off between effort and precision. The team of developers attempted to increase the precision with minimum effort, trying to reach good results in the least amount of time.

This literature section begins with a summary of the literature study that had to be made on time series forecasting in general. It also mentions the used models: ARIMA and Facebook's Prophet. And finally, it infers about previous works about fast-food/meal demand forecasting.

2.2. TIME SERIES FORECASTING

In "Time Series Analysis: Forecasting and Control", Box et al. (2015), describe a time series as "a sequence of observations taken sequentially in time" (p. 19).

A time series can be either continuous or discrete. When observations are logged continuously over some time interval, one is looking at a continuous time series. When they are logged at fixed time intervals, observations will make up a discrete time series (Brockwell & Davis, 2002).

A discrete time series can be originated in three different ways:

- By retrieving data from a continuous time series, only in fixed time intervals (for example, a store might keep track of the number of clients inside it at every second, but one might retrieve the number only every hour).
- By aggregating data over a period (which is the case at study, where all sales and transactions over a day were aggregated).
- By keeping track of inherently discrete data (for example, the number of employees working in a particular store every day).

An example of a discrete time series is presented in Figure 2.1.

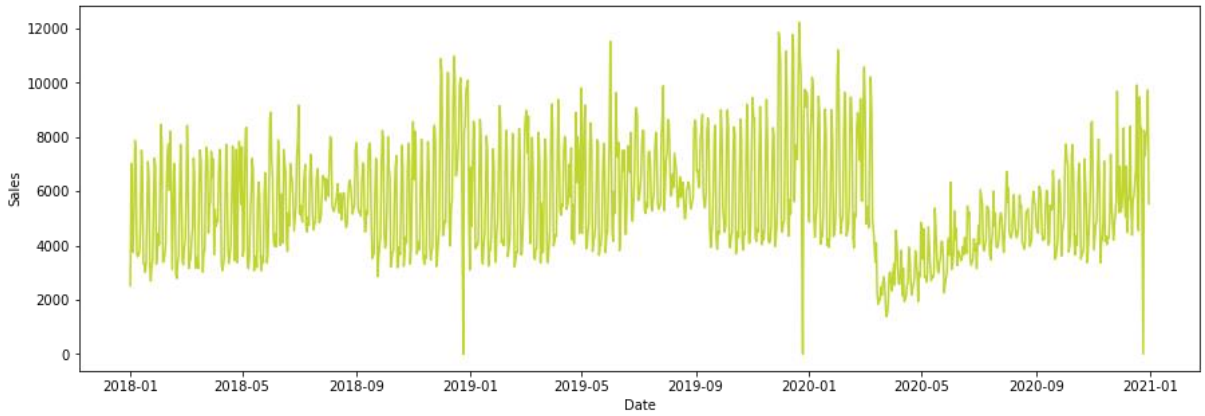


Figure 2.1 – Time series of total sales (in euros) from 2018 to 2020, in store 6

According to Montgomery et al. (2015), "a forecast is a prediction of some future event or events" (p. 1). When forecasting a time series, using the so-called "time series methods", one must consider that the future values will be related to the past, assuming the historic data follows a mathematical function that can be used to predict the future.

Supposing a collection of available observations that are discrete and spaced apart at equal distances, and that one pretends to forecast sales, they can think of z_t as being the sales at the current day t, making z_{t-1} yesterday's sales, z_{t-2} the number of sales from the day prior to yesterday, etc. Using those past observations, forecasts for the sales of the next day l, l = 1, 2, 3, … can be generated. Given the time series patterns on average, trends, seasonality, and cycle, the model assumes those patterns will be repeated in the future. If $\hat{z}_t(l)$ represents the forecast made at origin t for the sales of a lead day l, z_{t+l}, one can say the function $\hat{z}_t(l)$ provides forecasts for all future lead times, by using all n available observations, both current and past, z_{t-1}, z_{t-2}, z_{t-3}, …, z_{t-n}. Such function can be called the forecast function at origin t (Box et al., 2015; Krajewski et al., 2015). Over the years, scientists have studied different forecasting functions. An example can be given by a simple regression model,

$$y_t = x_t\beta + \omega_t, \tag{2.1}$$

with $x_t = \{x_t\}$ being the collection of observed sales, and ω_t being an unobservable white-noise sequence of independently and identically distributed random variables (Pollock, 1999).

The general temporal regression model, a more complex one, which looks to establish a relationship between any number of consecutive elements of x_t, y_t, and ε_t, is given by:

$$y_t = \sum_{i=1}^{p} -\alpha_i\, y_{t-i} + \sum_{i=0}^{k} \beta x_{t-i} + \sum_{i=0}^{q} \mu_i \omega_{t-i}, \tag{2.2}$$

where it is assumed that $\alpha_0 = 1$. The sums of the function can be infinite, but for the model to be viable, the sequences of coefficients $\{\alpha_i\}, \{\beta_i\}$, and $\{\mu_i\}$ must depend only on a limited number of parameters (Pollock, 1999).

As one can compare in Figure 2.2 and Figure 2.3, two years of sales are clearly similar, and 2019 sales can probably be predicted based on the 2018 ones.

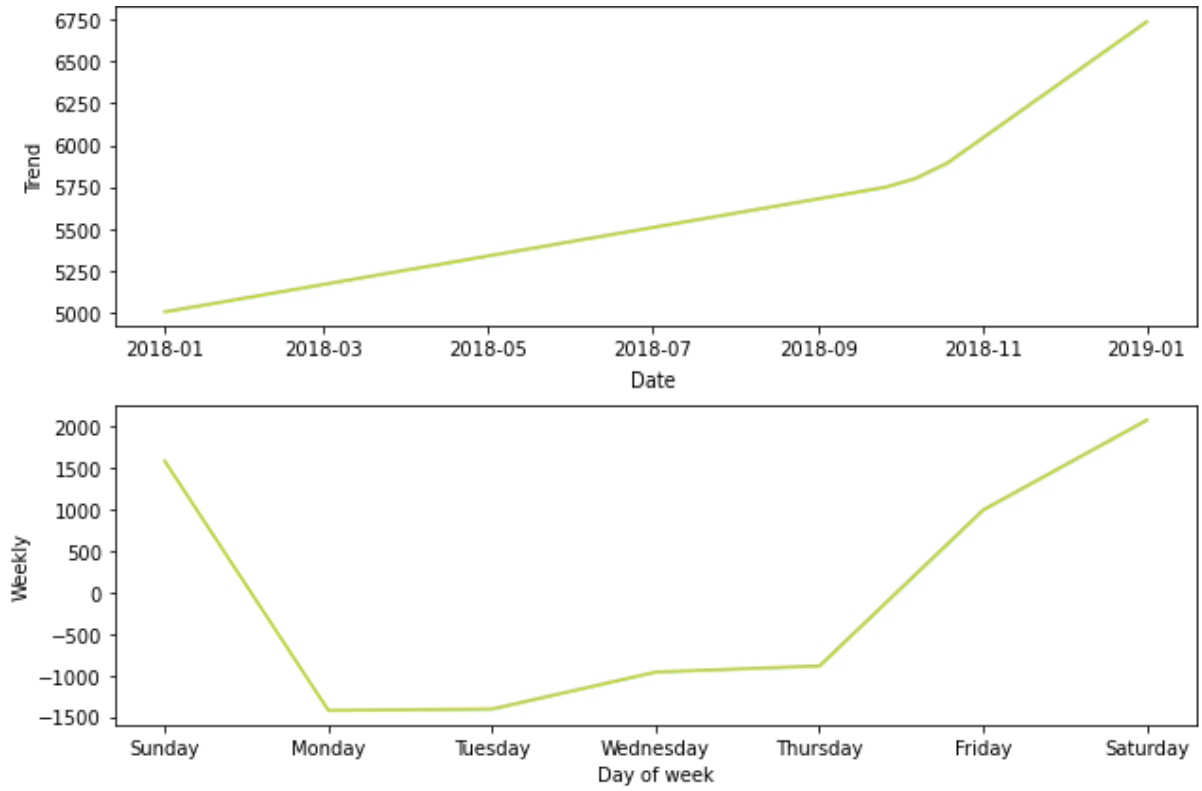


Figure 2.2 – Decomposition of the total sales time series during 2018, in store 6

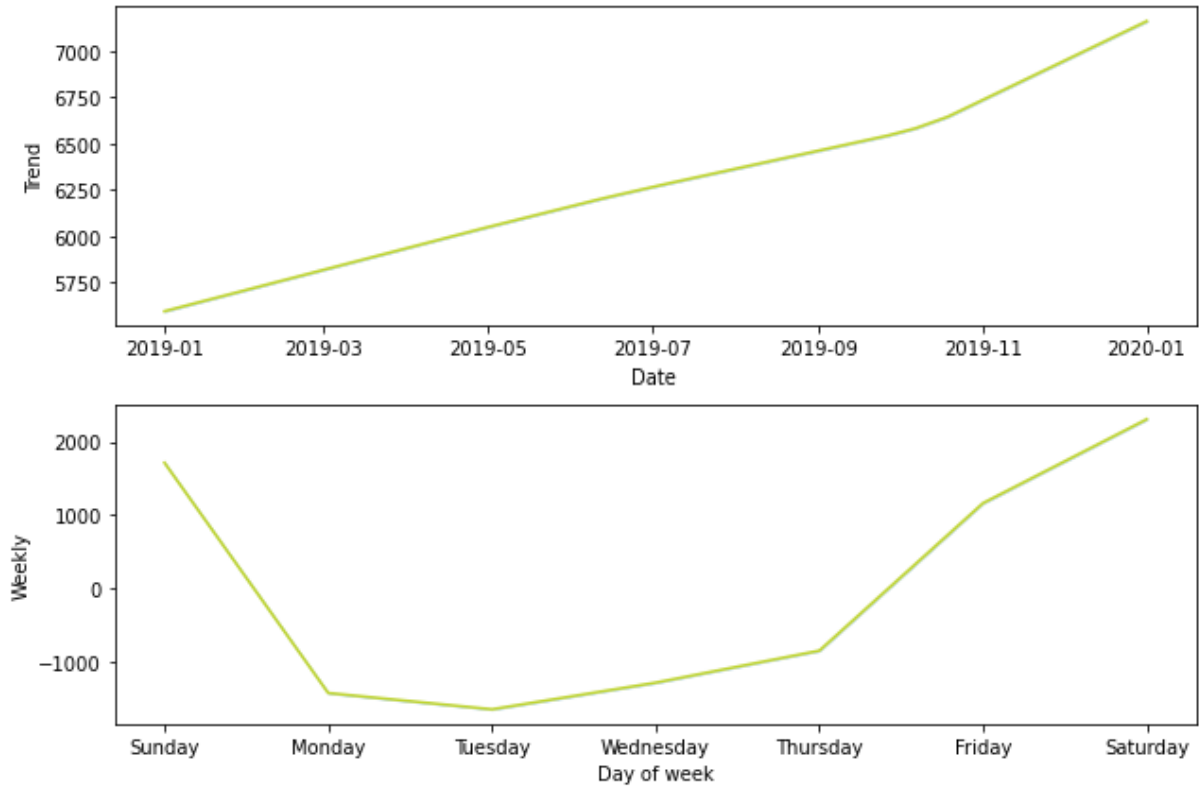


Figure 2.3 – Decomposition of the total sales time series during 2019, in store 6

2.2.1. Measuring forecast error

In 1978, George E. P. Box states “All models are wrong, but some are useful.”. The short version of this famous statement is still used in statistics to this day, “All models are wrong”. This means that though a scientific model can be valuable, it is almost (if not) impossible to make them 100% accurate. That is exactly how forecasting performance is measured, by calculating its error. As stated in “Operations Management: Processes and Value Chains”, a book by Lee J. Krajewski, Manoj K. Malhotra, and Larry P. Ritzman, one can measure forecast error using five different techniques.

2.2.1.1. Mean Absolute Deviation

Using the MAD technique, one will have an idea about the variance present in a dataset, by averaging the distance between each data point and the prediction, which translates to the following equation:

$$MAD = \frac{1}{T}\sum_{t=1}^{T}\left|X_t - \hat{X}_t\right|, \tag{2.3}$$

where T is the number of observations, X_t is the observed value at time t, and $\hat{X}_t$ is the forecasted value for time t.

The mean absolute deviation is commonly used. However, it can lead to errors in interpretation. Sometimes the MAD might be very small, but only because the demand is low. If the algorithm picks up on those null demands, the forecasts will start being lower and lower, leading to a lower MAD. As the algorithm is re-assured of these lower errors, it might start predicting in a biased way, foreseeing values lower than the real demand (Wallström & Segerstedt, 2010).

2.2.1.2. Mean Squared Error

The MSE measurement is similar to the MAD one. Instead of averaging the distance between data points and the mean, one must average the squared difference between the forecasted and actual values:

$$MSE = \frac{1}{T}\sum_{t=1}^{T}\left(X_t - \hat{X}_t\right)^2, \tag{2.4}$$

Since the MSE is associated with the standard variation of forecast errors, Silver (et al., 1998) recommends its use. Nevertheless, one should keep in mind that this measurement is sensitive to outliers and may produce smaller errors simply due to its squared function (Wallström & Segerstedt, 2010).

2.2.1.3. Mean Absolute Percentage Error

The MAPE is calculated by averaging the absolute percentage between the forecasted and real values:

$$MAPE = \frac{100}{T}\sum_{t=1}^{T}\frac{|X_t - \hat{X}_t|}{X_t}, \tag{2.5}$$

Since the error is provided in terms of percentages, it becomes easier to understand, and that makes it very commonly used. Also, because the percentages are absolute, situations in which negative errors would cancel out the positives are avoided.

It is because of its easy interpretability that MAPE was used to measure the prediction error of the Proof of Concept presented in this report.

However, MAPE cannot be used when the real value is null. The Proof of Concept did not have that problem, as will be explained later in this report. Still, if one wishes to use the MAPE but is faced with such a problem, the use of the Symmetric Mean Absolute Percentage Error might be considered. The sMAPE can vary between -200% and 200%, and is given by the following equation:

$$sMAPE = \frac{100}{T}\sum_{t=1}^{T}\frac{|X_t - \hat{X}_t|}{(X_t + \hat{X}_t)/_2} \tag{2.6}$$

Besides avoiding the zero-demand problem, another reason to use sMAPE is the fact that the error becomes symmetric. That way, it will not matter if the forecast is larger or smaller than the demand, as the error will be the same (Makridakis & Hibon, 2000).

2.2.1.4. Cumulative sum of forecast errors

The three previous methods do not account for bias in the data. To do that, one can use the cumulative sum of forecast errors, using the equation:

$$CEF = \sum_{t=1}^{T} X_t - \hat{X}_t \tag{2.7}$$

If they wish to find the CEF at a given period, they must sum the errors up through that period. A good use for this measure is to evaluate planning efforts. If the forecast is regularly smaller than the real demand, the value of the CEF will grow higher and higher over time, indicating a systematic deficiency in the model (Krajewski et al., 2015).

2.2.1.5. Standard Deviation

Finally, the standard deviation, σ, measures the likeliness that a forecast is accurate. This measure is often turned into a confidence score, and is granted by the following equation:

$$\sigma = \sqrt{\frac{\sum_{t=1}^{T}\left((X_t - \hat{X}_t) - \overline{(X - \hat{X})}\right)^2}{T-1}} \tag{2.8}$$

2.3. ARIMA MODEL

During the beginning of the 20th century, a Russian statistician and economist, named Eugen Slutsky, and a British statistician, called George Yule, suggested the Autoregressive and Moving-Average processes. However, these practices would only reveal their power in the 1970s, when the Box-Jenkins method was introduced by George E. P. Box and by Gwilym Jenkins.

An ARIMA model has three parameters: p (the number of auto-regressive terms), d (the number of times one must differentiate a series for it to become stationary), and q (the number of moving average terms) (Gujarati, 2003). To understand them further, various concepts must be introduced.

2.3.1. Autoregressive model (AR)

The autoregressive model (AR) is a specific type of regression model where the dependent variable (in the PoC, sales, or transactions) depends on its past values. This implies that the forecast is necessarily related to previously observed values. This model is based on the concept of partial autocorrelation. An AR model of order p is given by the following equation:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \omega_t \tag{2.9}$$

Just like before, ω_t stands for an unobservable white-noise sequence of independently and identically distributed random variables.

2.3.1.1. Autocorrelation and partial autocorrelation

Autocorrelation is used to measure the linear relationship between values of the same variable at different times. The interval that separates two points in time is called the lag (Hyndman & Athanasopoulos, 2018; Montgomery et al., 2015).

The autocorrelation coefficient at lag k, i.e., the coefficient of the correlation between the sales at time t, y_t, and the sales at another time period, y_{t+k}, can be written as

$$\rho_k = \frac{E[(y_t - \mu)(y_{t+k} - \mu)]}{\sqrt{E[(y_t - \mu)^2]E[(y_{t+k} - \mu)^2]}} \tag{2.10}$$

where μ is the mean of the data. The autocorrelation function is the collection of values of ρ_k, for k = 0, 1, 2, …. A correlation between a variable and itself is 1, so when k=0, it is assumed that the autocorrelation is 1. To get such a function, one must multiply both sides of the AR(p) function mentioned before by X_{t-k}, which leads to

$$\begin{gathered} X_{t-k}X_t = \phi_1 X_{t-1}X_{t-k} + \phi_2 X_{t-2}X_{t-k} + \cdots + \phi_p X_{t-p}X_{t-k} + \omega_t X_{t-k} \Leftrightarrow \\ \Leftrightarrow p_k = \phi_1 \rho_{k-1} + \phi_2 \rho_{k-2} + \cdots + \phi_p \rho_{k-p} \Leftrightarrow \\ \Leftrightarrow \phi(B)\rho_k = 0, \\ \phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p \end{gathered} \tag{2.11}$$

The set of equations one gets after replacing k with values 1, 2, … are called ad the Yule-Walker equations. Since the ACF does not depend on the scale of measurement of the time series, that makes it a dimensionless quantity. Besides this, the function is symmetric around 0, meaning $\rho_k = \rho_{-k}$.

To better understand this concept, a particular example is shown. Below, one can see the plot showing the correlation between the total sales of store 6 in relation to its past values. Such plot is known as the Auto Correlation Function Plot (ACF Plot).

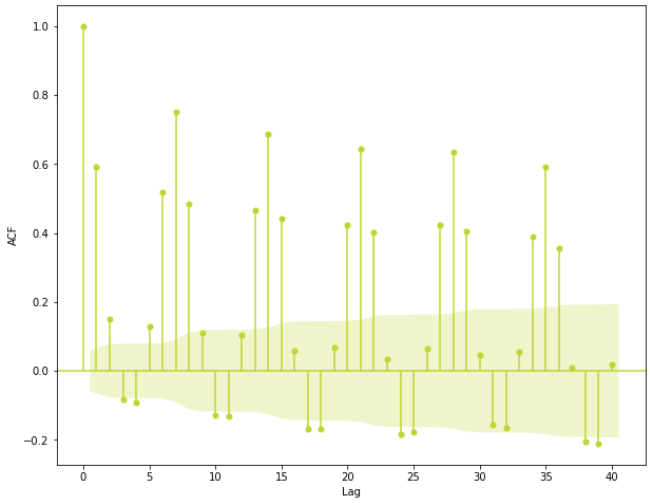


Figure 2.4 – ACF Plot for the total sales time series (2018 to 2020), in store 6

As one can see, today's sales have a big correlation with the number of sales from yesterday, from 6 days ago, from 7 and 8 days ago, and so on. To simplify, one can focus only on the first 4 higher correlations (lags 1, 6, 7, and 8). Assuming today is Friday, the past data that is relevant for today's forecast is data from yesterday (Thursday), and data from last week's Saturday, Friday, and Thursday. But yesterday's values had been related to the previous day (Wednesday), and the week prior. This means that when one makes a prediction, that forecast will depend on all previous ones. A simple schema to help the understanding of the concept is shown in Figure 2.5.

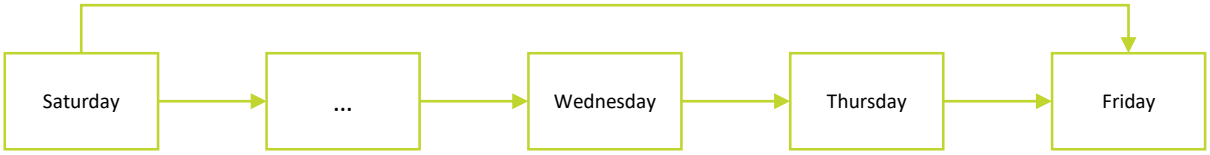


Figure 2.5 – Autocorrelation: daily influence in a forecast

In the case of partial autocorrelation, it is only taken into consideration the days that affect directly today's forecast. Implying that today's value will depend on Thursday's, Saturday's, and Sunday's. But, as one can see in Figure 2.6, the model will not look at the past values that influence each one of those. A simple correspondent schema is shown in Figure 2.7.

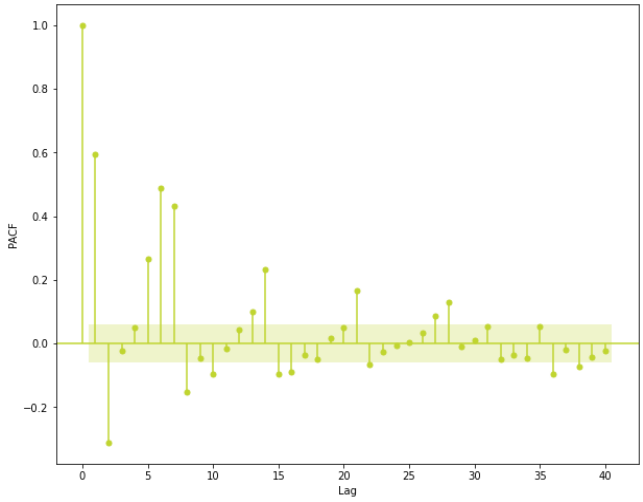


Figure 2.6 – PACF Plot for the total sales time series (2018 to 2020), in store 6

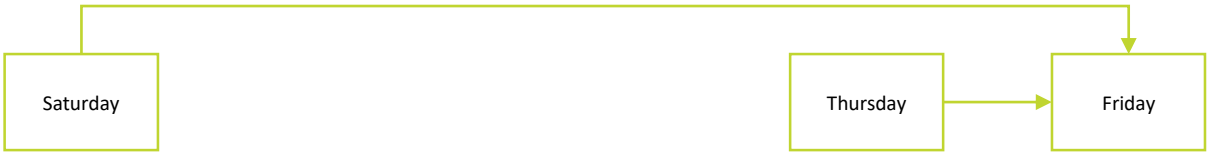


Figure 2.7 – Partial autocorrelation: daily influence in a forecast

Because of this, the PACF between y_t and y_{t+k} becomes the autocorrelation amongst y_t and y_{t+k} after adjusting for y_{t-1}, y_{t-2}, …, y_{t-k+1}:

$$\rho_j = \frac{E[(y_t-\mu)(y_{t+k}-\mu) \mid y_{t-1},\dots,y_{t-k+1}]}{\sqrt{E[(y_t-\mu)^2 \mid y_{t-1},\dots,y_{t-k+1}]E[(y_{t+k}-\mu)^2 \mid y_{t-1},\dots,y_{t-k+1}]}} \quad (2.12)$$

The autoregressive part of the ARIMA will then give different weights to past days, according to its correlations with the forecast. To find $P_{kk}, k = 1, \dots, p$ one can apply Cramer's rule. If $p < k$, P_{kk} will equal 0, and the PACF of an AR(p) must cut down to zero after lag $k = p$, where p is the order of the AR model. p can be thought of as the number of past days the model has to take into consideration.

2.3.2. Moving-average model (MA)

The concept behind the moving-average model is like the one behind the autoregressive model. The MA part of the ARIMA model will consider the error resulting from previous forecasts. Again, assuming today is Friday. Yesterday, Saturday's values were used to make today's predictions. Unfortunately, those forecasts were below the real demand. That error will be inputted into today's estimates. Just like in an autoregressive model, different weights are given to different errors, according to the correlations between the days. This time, the number of days from which errors should be considered is introduced by using the parameter q, known as the order of the MA model. For a general MA(q) process, the ACF must cut down to zero after the lag $k = q$.

The three most popular types of moving averages are:

- The Simple Moving Average (SMA),

$$SMA = \frac{A_1 + A_2 + \cdots + A_n}{n}, \tag{2.13}$$

  with A_n being the average at the time period n, and n being the number of time periods.

- The Weighted Moving Average (WMA),

$$WMA = \frac{A_1 \times n + A_2 \times (n-1) + \cdots + A_n}{n \times (n+1)/2}, \tag{2.14}$$

- The Exponential Moving Average (EMA),

$$EMA_t = \left[V_t \times \left(\frac{s}{1+n}\right)\right] + EMA_{t-1} \times \left[1 - \left(\frac{s}{1+n}\right)\right], \tag{2.15}$$

where EMA_t is the EMA for the time period t, V_t is the value for that time period, EMA_{t-1} is the EMA for the time period $t-1$, s is the smoothing factor, and n is the number of data points. Usually, the smoothing factor follows the formula $2/(selected\ time\ period + 1)$ (Alexander, 2008).

2.3.3. Integration (I)

To use AR and MA models, it is necessary to have a stationary time series. This means that the average and the standard deviation of the time series points must be constant through time and that the series does not show the existence of seasonality. In the PoC, just in total transactions, 40 out of 48 stores did not have a stationary time series. Without transforming them into stationary ones, it would not be possible to generate predictions using the ARIMA model. Usually, to turn a non-stationary series into a stationary one, one uses the difference between each day and the day before, a calculation that is given by the equation $Z_t = X_{t+1} - X_t$. Sometimes, doing that one time is not enough. The model must know the number of differentiations it has to do so the series becomes stationary. That is exactly why the number of differentiations the model has to do so the series becomes stationary must be inserted into the model. That information is passed through the parameter d.

2.3.4. ARIMA pros and cons

The ARIMA model makes very few assumptions and is very versatile. Furthermore, the model is iteratively selected based on the underlying data structure, having no need to arbitrarily assume a model prior to analysing the data (Ho & Xie, 1998).

It is also found that when a time series is non-stationary, one of the most effective approaches to make demand forecasts is the ARIMA model (Box et al., 2015).

Though ARIMA can be considered complicated in terms of theoretical concepts, it is such a well-known model that many user-friendly software packages are now available to help data scientists build models. (Hot et al., 1998).

Unfortunately, the model comes with more disadvantages other than its hard-to-understand theoretical basis, requiring a lot of data, and being computationally expensive. ARIMA models also struggle to predict changes in trend behavior.

Still, they are known to perform well in short-run forecasts, most times, even better than other sophisticated structural models (O'Donovan, 1983).

2.4. FACEBOOK'S PROPHET ALGORITHM

Besides using ARIMA to generate forecasts in the PoC, a model that is much more recent, Facebook's Prophet, was also used. This algorithm was originally built to make predictions on internal projects. The code would later become open source (accessible to the general public), making it available to anyone who wishes to use it. (Taylor & Letham, 2018).

Prophet is based on an additive regression model. Unlike ARIMA, which can be thought of in 3 different parts, this algorithm can be decomposed into 4 four parts.

2.4.1. Additive regression

In a so-called simple regression (Equation 2.1), a statistician attempts to model the relationship between two variables, one of them being dependent (the one wished to be forecasted), and another being independent, often called explanatory. In the case of multiple regression, there are multiple explanatory variables (Gujarati & Porter., 2009). In multiple regression, it is assumed that the dependent variable is related to all the explanatory ones, but that those are not related to each other. The model is formulated as:

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n + \varepsilon, \tag{2.16}$$

with y being the dependent variable, $x_i, i = 1, 2, \dots, n$ being the explanatory variables, $\beta_i, i = 1, 2, \dots, n$ being the regression coefficients, and ε being the error.

In the real world, it is almost impossible to guarantee that the explanatory variables do not influence each other and that their individual influence on the dependent variable is linear. The additive

regression model drops the linearity assumption of the multiple regression one. As a result, a new model is created, in which the dependent variable can be explained by each of the explanatory ones via an individual functional form (Schimek & Turlach, 1998). To account for possible relationships between explanatory variables, it is also added a marginal effect to each of them. All of that makes the additive regression a more flexible model than the standard multiple regression one. Prophet uses a decomposable time series model (Harvey & Peters, 1990), with three components: trend, seasonality, and holidays which can be characterized by the following function:

$$y_t = g_t + s_t + h_t + \varepsilon_t, \tag{2.17}$$

where g_t represents the non-periodic changes in the time series (Trend), s_t the periodic changes (.

Annual and weekly seasonality), h_t the effect of the holidays in the forecast (Holidays), and ε_t the changes not accommodated by all the previous functions. The model assumes that such errors are normally distributed (Taylor & Letham, 2018).

2.4.2. Trend

Each time series' trend is modeled using one (or more) linear curves. One, if the time series follows a linear trend in terms of increase or decrease. In the PoC, this situation is pretty much nonexistent. There are periods where the sales/transactions begin to rise, only to fall again. Considering that the period includes the COVID-19 pandemic, it is assumed that the demand suffers, not only due to the rise/fall in the number of virus cases but also due to the various lockdown measures that took place. This means the series trend cannot be modeled with only one curve. In this case, the series is divided into parts, and each of them is modeled individually. Prophet can use two different trend models: a saturating growth model and a piecewise linear model. In the specific case of the PoC, one can focus only on the piecewise linear model, as the saturating growth does not apply to any of the cases.

Supposing there are S change points at time $s_j, j = 0, 1, 2, \dots, S$ in a time series, one can define the vector of rate adjustments as $\delta \in \mathbb{R}^2$, with δ_j is the change in rate that occurs at time s_j. Considering that k is the growth rate, the rate at time t will be given by: $k + \sum_{j:\, t>s_j} \delta_j$ (i.e., is it given by the base rate k plus all the rate adjustments up to that point). Considering a vector $a(t)$ such that, $a(t) \in \{0, 1\}^S$, one can re-write the previous formula as

$$k + a_t^T \delta, \text{ where } a_j(t) = \begin{cases} 1, & if\ t \geq j, \\ 0, & otherwise. \end{cases} \tag{2.18}$$

As one adjusts the rate k, they must also adjust the offset parameter, m, in order to connect the endpoints of the segments. The correct adjustment at change point j is given by

$$\gamma_j = (s_j - m - \textstyle\sum_{l<j} \gamma_l)\left(1 - \frac{k + \sum_{l<j} \delta_l}{k + \sum_{l \leq j} \delta_l}\right) \tag{2.19}$$

Then, knowing all of this, if the trend is modelled by the piecewise linear model, that means it is being modelled by the following equation:

$$g_t = (k + a_t^T \delta)t + (m + a_t^T \gamma). \quad (2.20)$$

To make the function continuous, it is considered that $\gamma_j = -s_j \delta_j$ (Taylor & Letham, 2018).

Now, when implementing the algorithm, one can provide information for the selection of the changing points. On the other side, the algorithm can automatically detect turning points in a series trend, making it more flexible than the ARIMA model. This is done by putting a sparse prior on δ, in Equation 2.20.

2.4.3. Annual and weekly seasonality

When working with business-related time series, an analyst can have a multi-period seasonality. This happens because humans can act differently according to the point in time they are in (Taylor & Letham, 2018). In the PoC, if one focuses on a Lisbon store, it is most likely that it will have a higher number of transactions during the week. And if one focuses on a Greater Lisbon store, it is most likely that the transaction number rises during the weekend. This happens because most people who work in Lisbon do not live in Lisbon. So, if they wish to get a meal during the workweek, they are more likely to do it in a Lisbon store, but if they wish to get one during the weekend, they will probably choose a store that is closer to their home. Examples of this behavior are shown below, where Lisbon's store 22 shows a drop in sales during the weekend, and Greater Lisbon's store 36 shows the opposite.

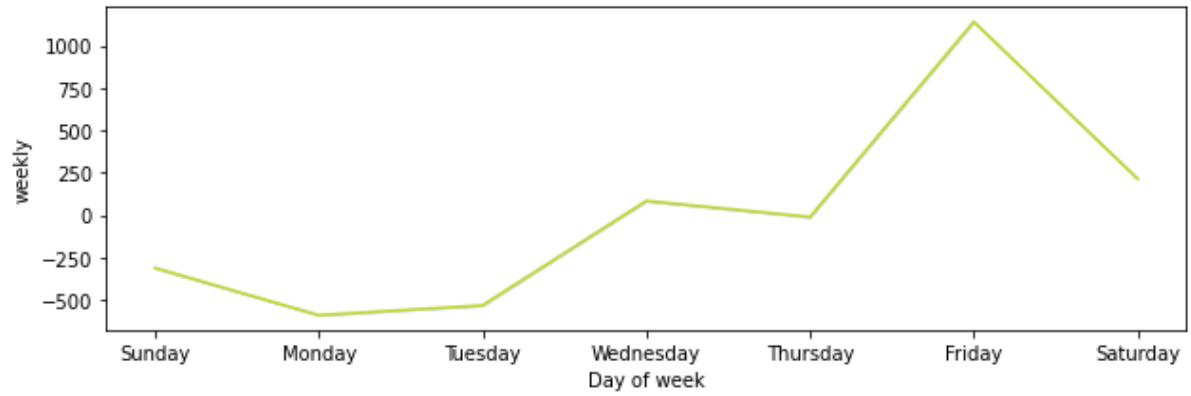
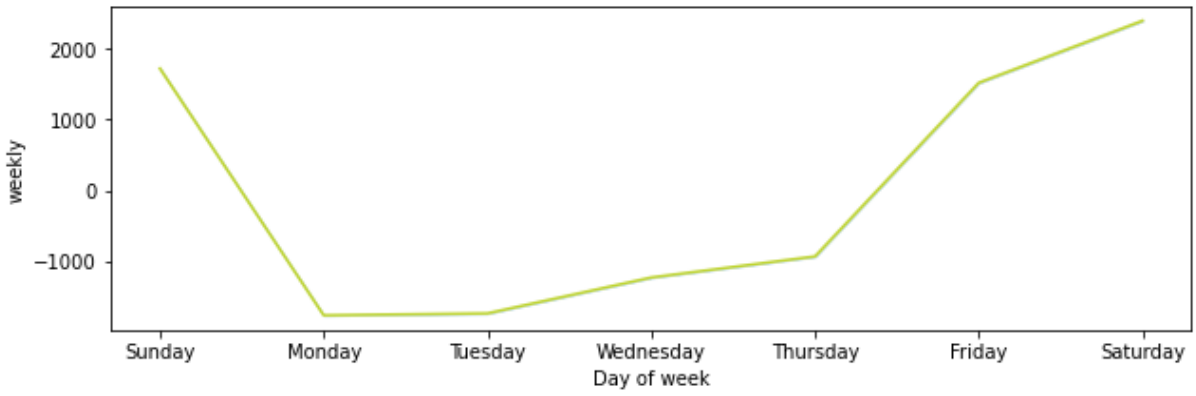


Figure 2.8 – Weekly seasonality of the total sales time series (2018 to 2020), in store 22 (from Lisbon)



To model annual and weekly seasonality, Prophet uses the Fourier series. In mathematics, these types of series are used to decompose functions that are time-dependent into functions that are period-dependent (Harvey & Shephard, 1993). The underlying idea is that any arbitrary signal can be approximated with enough sin curves (Hyndman & Athanasopoulos, 2018). A standard Fourier series is given by:

$$s_t = \sum_{n=1}^{N}\left(a_n \cos\left({}^{2\pi nt}/_{P}\right) + b_n \sin\left({}^{2\pi nt}/_{P}\right)\right). \quad (2.21)$$

If a series has yearly seasonality, then $P = 365.25$. N can be automatically chosen, but Taylor and Letham (2018) have found that $N = 3$ or $N = 10$ are generally a good choice. The higher the N, the quicker changes in seasonal patterns are detected, but one must be careful, as the risk of overfitting increases too.

The use of the Fourier series allows for higher flexibility when modelling seasonality, hence their usage in the algorithm.

2.4.4. Holidays

Holidays and events can cause a predictable effect on customer demand. As said before, children's day in Portugal (June 1st) is a particular day where sales spike at the PoC client's stores. However, these days do not follow a periodic pattern, so it becomes hard to model them using a smooth cycle (Taylor & Letham, 2018). June 1st can be any day of the week, so one cannot declare it programmatically. But, as one can see in the figure below, the impact of that particular day on the time series is similar year after year.

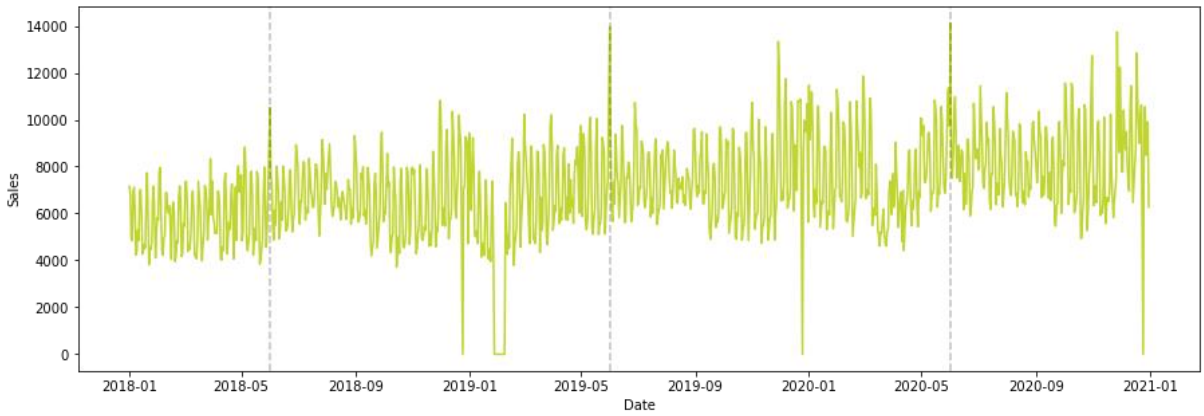


Figure 2.10 – Time series of total sales (in euros) from 2018 to 2020, in store 37, marked by a rise in demand every June 1st

Prophet allows the user to incorporate holidays into the algorithm and provides a list of holidays for each specific country. If the user previously knows a holiday that can impact the model but does not come in the country's list, they can add it to that list. The algorithm can then learn to increase or decrease the forecast according to the day it is predicting, bypassing the instability that is often caused by special days.

If t is during the holiday i, and κ_i is the corresponding change in the forecast, then such effect can be added to the model by the means of

$$h_t = Z_t \kappa, \tag{2.22}$$

where $Z_t = [1(t \in D_1), \dots, 1(t \in D_L)]$, $D_j, j = 1, 2 \dots, L$ is the collection of holiday dates, and $\kappa \cap Normal(0, v^2)$.

2.4.5. Prophet pros and cons

Unlike ARIMA, Prophet is "designed to have intuitive parameters that can be adjusted without knowing the details of the underlying model" (Taylor & Letham, 2018, p. 38). As mentioned before, one of the most powerful parts of Prophet is its ability to automatically detect turning points. Nevertheless, this feature comes with a catch: the automatization can lead the model to over or under fit. On the positive side, the user can tune the number of changing points or the way the model detects them.

Another big advantage of Prophet is the fact it still works when values of the series are missing, and it is not as sensitive to outliers. Besides all of this, it is much faster to fit the model to a dataset than ARIMA (Taylor & Letham, 2018).

One cannot forget, however, that the ARIMA model has been proven to be more powerful than these newer, more sophisticated models. Later in this report, a comparison between the performances of both algorithms in each of the PoC time series will be presented.

2.5. PREVIOUS WORKS ON FORECASTING MEAL-RELATED DEMAND

Demand forecasting can be considered essential to a restaurant's procedures strategy. Good forecasting means efficiency and effective foodservice operations. Analyzing the demand of the customers helps a business when doing item production decisions, when preventing wastage, and when allocating resources, such as staff or stock. The more on target a long-term forecasting is, the more accurate of a budget is generated, which could result in money available to other activities. Knowing how a store's demand will change can impact the marketing decisions, and eventually increase consumption (The Restaurant Times, 2020; Egan et al., 2020).

In 1990, researchers Judy Miller, Cynthia McCahon, and Brenda Bloss looked to forecast the customer count for a daily period in a university dining hall. At the time, not many forecasting systems had been implemented, and in this particular case, it was believed that mathematical models could improve production planning, which would lead to less waste and lower operational costs.

Three modeling techniques were used: a naïve (manual) method, a simple moving average model, and a simple exponential smoothing model. The naïve model assumed each day of the week would have the same demand as it had the previous week (i.e., using last Monday's demand as the prediction for the next Monday's demand). The simple moving average model predicts demand by using the mean of the meals served on a particular day of the week from the previous 3 weeks. And finally, the simple exponential smoothing (SES) model predicts the demand for a day by taking the latest SES forecast, and by taking into consideration how off the last forecast was from the actual real

value. This error is added in proportion, which is imputed into the model through a parameter named alpha. This translates into the function

$$\hat{y}_{t+1|t} = \alpha y_t + \alpha(1-\alpha)y_{t-1} + a(1-\alpha)^2 y_{t-2} + \cdots. \quad (2.23)$$

For this specific case, the researchers went with alpha equal to 0.3.

After applying all the models, the authors found that the naïve model was outperformed 64% of the time by the time series models. Between the moving average and the exponential smoothing model, it was found that the first outdid the second. Because of this result, they came to the conclusion that using mathematical models can improve the predictions for dining hall meal consumption.

In 2004, Lee Blecher did something similar, this time comparing 5 techniques to predict meal demand. In his research, the author attempted to forecast the number of meals served in seven different congregate meal sites in Southern California. Without a forecasting system, managers relied on the individual reservations of those who wish to have lunch on a particular day. However, those reservations can be done up to the day before the day of the meal, making organization much harder, as one would only have a day to purchase ingredients, to plan the process, to assign people, etc. With such a little amount of time, those activities ended up being done prior to the day before, without an idea of how many people could still sign up. Having to guess the number of people that would still sign up, the process would usually lead to higher amounts of waste and operational costs.

To generate his predictions, the researcher used a naïve model (the same as the one used in 1990), and a Simple Exponential Smoothing model. He also used three different versions of the simple moving average: the first, like the one mentioned before, took into account the demand for the past 3 weeks. The second version did the same but using the data from the previous 5 weeks. And the third version used the average of the past 5 weeks, but only after removing two values: the highest and the lowest. In the end, this third version was actually a mean of 3 weeks, but not necessarily consecutive.

After applying the five models, the author compared their performances and found that in all seven congregate meal sites, the “time series forecasting techniques provided better predictions of meal demand when compared with the naïve method” (p. 1282). In conclusion, the author re-assures the previous idea - that simple forecasting techniques can provide better results than a naïve technique.

In 2016, researchers Agnieszka Lasek, Nick Cercone, and Jim Saunders performed a literature survey and a categorization of methods used for forecasting restaurant sales and customer demand. Though in 1990 there were already attempts to model such behaviors, this group of investigators found that there was no “review of forecasting methods for the restaurant industry” (p. 480). They used that opportunity to survey and classify forecasting techniques that had been published in the past 20 years. It was found that certain features are valuable to generate predictions, namely variables related to time, weather, holidays, promotions, events, historical data, macroeconomic indicators (in case the target of the forecast is monthly or yearly), competitive issues, web, location type and demographics of the location.

In terms of algorithms, Lasek, Cercone, and Saunders focused on multiple regression, Poisson regression, Box-Jenkins models (AR, MA, and ARIMA), exponential smoothing, and Holt-Winters models, artificial neural networks, Bayesian network models, hybrid models, and association rules.

In the end, the researchers came to the conclusion that each unique situation will require its own algorithm. In most cases, various models are implemented and the one that has the best performance is selected. In their opinion, the techniques that can take into account external variables like the ones mentioned before are the best. Adding to this, they mention other authors have created hybrid systems that seem to perform better than separate algorithms.

2.5.1. Previous works on forecasting fast-food related demand

In 2001, Lon-Mu Liu, Siddhartha Bhattacharyya, Stanley L. Sclove, Rong Chen, and William J. Lattyak published a paper in which they detailed their approach when data mining and predicting demand on a fast-food restaurant franchise. The forecast was made using the ARIMA model, and the authors found that identifying the different model parameters was "the most complicated and difficult task" (p. 463). Because of the amount of time series at hand, the researchers found that by using an automated system, "the need for visual examination of statistics in intermediate analysis can be greatly reduced or eliminated" (p. 463). Besides applying the ARIMA model on the raw dataset, an outlier detection technique was also applied, leading to a lower estimated residual standard error than the one obtained without it. In the 365 observations, 8 outliers had been detected. According to the authors, outliers in a time series can help detect significant events or exceptions, allowing the managers of the stores to take advantage of that information. However, one must be careful with outlying data, as they might be related to random special events but still impact the algorithm predictions (an example is given, of a school bus stopping at a restaurant to eat after a field trip). Moreover, it was noted that the dataset was missing data from some days, which was then "replaced by appropriately estimated values" (p. 470). In the end, it was found that though the automatization of figuring out the model parameters made the process a lot easier, the researchers still agree that at least in the early stages of data analysis, a more manual approach should be used, to detect abnormal situations. The outlier detection technique improved results, but one must be careful when applying it.

Later, in 2018, Mateus Meneghini, Michel Anzanello, Alessandro Kahmann, and Guilherme Luz published a paper on their effort to forecast meat demand in a fast-food restaurant. Instead of using a quantitative method like the ones seen previously, these researchers added expert judgment, in order to adjust their predictions. To start, the authors attempted to forecast the demand for two specific hamburgers, using four different forecast models – additive and multiplicative exponential smoothing of Holt-Winters and moving average (using the previous 2 and 3 days). The Holt-Winters model is similar to the one behind Prophet's, as seen in (2.17). This method takes into account level, l_t, trend, b_t, and seasonality, s_t. Knowing m is the frequency of the seasonality, that $a, \beta^*$and γ are, respectively, the smoothing parameters for level, trend, and seasonality, and that k is the integer part of $\frac{(h-1)}{m}$, the additive exponential smoothing of Holt-Winters is given by

$$\begin{aligned} \hat{y}_{t+h|t} &= l_t + hb_t + s_{t+h-m(k+1)}, \\ l_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1}) \\ b_t &= \beta^*(l_t - l_{t-1}) + (1-\beta^*)b_{t-1} \end{aligned} \tag{2.24}$$

$$s_t = \gamma(y_t - l_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}.$$

Whereas, the multiplicative exponential smoothing of Holt-Winters is given by

$$\hat{y}_{t+h|t} = (l_t + hb_t)s_{t+h-m(k+1)}, \tag{2.25}$$

$$l_t = \alpha\left({}^{y_t}/_{s_{t-m}}\right) + (1-\alpha)(l_{t-1} + b_{t-1})$$

$$b_t = \beta^*(l_t - l_{t-1}) + (1-\beta^*)b_{t-1}$$

$$s_t = \gamma\left({}^{y_t}/_{(l_{t-1} + b_{t-1})}\right) + (1-\gamma)s_{t-m}$$

(Hyndman & Athanasopoulos, 2018)

The first, additive exponential smoothing of Holt-Winters, brought the best results when predicting demand for both hamburgers. The authors then adjusted their predictions using expert knowledge. Expert judgment came from three employees, who evaluated their colleagues in terms of experience. Between the three, a promotional factor, a store-reform factor, a special day factor (Children's Day), and a climate factor were identified as possible influences on the restaurant's demand. Independently, each employee gave their optimistic and pessimistic guess on how much they thought the factors influenced the meat request (on a percentage scale). Weights to their opinions were given, based on the expert's experience, schooling, and company time, leaving the most qualified with the higher weight to their opinion. After using this qualitative measure to adjust the results obtained with the mathematical model, the previous error of 38% lowered to 10%.

3. METHODOLOGY

3.1. PROJECT INTRODUCTION

As stated in 1.2, the client decided to explore (almost) all stores in Lisbon and Greater Lisbon and expected Noesis to build forecasting models to predict sales (in euros) and transactions (in units). Overall, 10 algorithms were to be developed, each looking to forecast:

- Total sales and transactions.
- Delivery sales and transactions (meals prepared and delivered by services like Uber Eats or Glovo).
- Drive sales and transactions (meals prepared and served in the drive-through).
- Eat-in sales and transactions (meals prepared and expected to be eaten inside the store).
- Take-away sales and transactions (meals prepared and bagged to be eaten outside the store).

Two existing algorithms were used: ARIMA and Facebook Prophet.

3.1.1. Stores

In total, after removing and joining stores together with the client, 48 stores (46 real and 2 mocks created from existing ones) from Lisbon and Greater Lisbon were analysed. These restaurants can be divided into three smaller groups: Free Stands, Malls, and Storefronts. Such distribution is shown below:

Region	Store Type	Store ID [1]
Lisbon	Free Stand	25
		38
		39
		46
	Mall	2
		3
		19
		22
		26
		32
		45
	Storefront	4
		9
		13

[1] Encoded from the original store IDs.

		20
		21
		29
		31
		34
		40
		43
		100
Greater Lisbon	Free Stand	10
		11
		15
		16
		23
		27
		28
		30
		33
		35
		36
		37
		42
	Mall	5
		6
		7
		8
		12
		14
		17
		18
		41
		101
	Storefront	1
		24
		44

Table 1 – Distribution of stores per region and store type

3.1.2. Variables

The variables provided by the company were (via the Noesis DAAI BI experts):

- Date.
- Store IDs (encoded in this report).
- Store names (occulted from this report).
- Store types (Free Stand, Mall, or Storefront).
- Sale type (Delivery, Drive, Eat-In, or Take-Out).
- Sales (in euros) for each day and sale type.
- Transactions (in units) for each day and sale type.
- The number of active campaigns (marketing campaigns, coupons, etc.) for each day and store.

Besides these variables, the client also gave access to data related to media campaigns (investment in GRPs and awareness in percentage). Using intuition and some business knowledge, weather data was also added.

3.1.3. Development tools

All of the code developed to produce forecasts was done using Python. The raw data was stored in Azure Blob Storage, transferred to Azure Databricks, where PySpark was used for reading. The produced results were then written back to Azure Blob Storage. After, the predictions were downloaded and sent to Noesis' BI specialists, to be fed into a Qlik Sense dashboard, that allowed the client to view the forecasts in a more intuitively and interactively way.

3.1.4. Success metrics

In order to test the developed algorithms, sales and transactions were predicted for the months of June, July, August, September, and October of 2020. These months were not chosen at random: since COVID-19 cases in Portugal were slowing down, most lockdown measures had been lifted at the time. The client understood that lockdown months were atypical, and more than likely, completely unpredictable.

The option of removing June and October was kept open, as the first month was marked by lockdown measures being slowly taken down and the second by a visible increase in COVID-19 infections.

Each algorithm was trained with the last X days. It was then used to predict the following 7 days. The forecast for the next day was added to the train set. Afterwards, X+1 days were used for training and used to predict the following 7. In the end, each date in the test set had 7 forecasts.

To evaluate the model performance, a known metric, Mean Absolute Percentage Error was utilized, which allowed to form an idea of how far off the predicted were from the real values. Though MAPE is not always advised as a way to measure performance, the metric was kept, as it was much easier to explain to the client.

Given that each day has 7 forecasts, a day's final MAPE is considered as being the average of the 7 forecasts. To evaluate the algorithm performance as a whole, those means were averaged out.

3.2. ROADMAP

Together with the client, a 36-day roadmap was defined, predicting a project kick-off on April 5th, 2021, and a final presentation on May 25th.

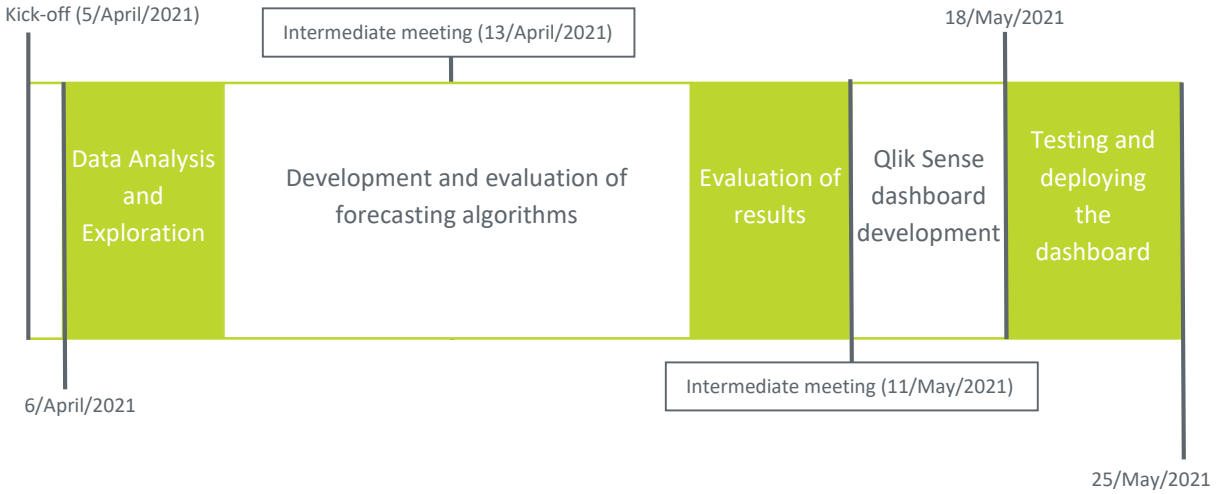


Figure 3.1 – Proof of Concept roadmap

Most of the above roadmap was followed. The development and evaluation of forecasting algorithms and the evaluation of the results ended up being merged into just one stage. All the results and documentation were made available to the client on June 2nd, along with the Qlik Sense dashboard developed by the BI consultants.

3.3. DATA VALIDATION AND FUNCTIONAL CONSIDERATIONS

To kick-off the Proof of Concept, a data validation stage had to take place, and some considerations had to be gathered, to solve some problems found during it:

- The days/sale types with negative sales and/or transactions were considered to have sales and/or null transactions.
- The days/sale types with sales but with no transactions were considered to have null sales.
- The days/sale types with transactions but with no sales were considered to have null transactions.

Note that the above rules were only put in place after it was verified that those problems represented less than 1% of the total dataset.

As mentioned before, two of the restaurants are mocks, made by joining data from two other stores:

- Store 100 is a combination of two others. Different IDs represented the same store, but before and after remodeling. Because the store was renamed at the end of 2019, it had to be removed from the dataset. Since the testing period was in 2020, the second ID, for the purposes of this report, numbered 9999, was likewise deleted, as it would only have a short amount of data to train the model. A visualisation of store 9999's total transactions can be found in 8.1.
- Store 101 is also a combination of two stores, one of them being store 18. Store 101 was a store in Greater Lisbon that was re-located in 2018. With the new building, the store ID was renamed to 18. Just like before, the first store was removed, has it did not have data for the testing period. However, store 18 was kept. This way, a comparison could be made between the two stores, to find out which had better performance:
    - Store 101, with data from January 2018 to March 2020, or
    - Store 18, with data from January 2019 to March 2020.

Finally, one can interpret special days as being made up by:

- National holidays.
- St. Anthony's Day (because it is a holiday in Lisbon).
- Valentine's Day, Mother's Day, Father's Day, and Children's Day.

It is important to denote that days with zero demand were not considered when calculating prediction errors. Together with the client, it was concluded that it was pretty much impossible to have unexpected days/sale types with no purchases. In all those cases, the client previously knows that a store will be closed. In such a big city like Lisbon, it is very improbable that the demand is indeed null merely due to customer behavior.

Besides the days with negative sales and/or transactions, some restaurants had to be removed from the dataset, either because of their unique non-predictable behavior or due to lack of data. No other problems were found.

3.4. Data Analysis and Exploration

Before beginning the development of algorithms, an analysis of the hundreds of time series used to feed the models was conducted.

All the plots mentioned in the visual analysis can be found in 8 - Appendix. Because so many time series are visually similar in terms of trends and patterns, most of them have been omitted, being represented by the displayed ones. The majority of the time series plots have two distinct lines on them, representing the dates of March 17th, 2020, and June 3rd, 2020. Their addition is due to their importance in the evolution of the COVID-19 pandemic in Portugal. The first is when all non-essential services (including fast-food stores) were mandatorily closed. The second is when all non-essential services were free to open in total to customers again.

3.4.1. Time series visualisation

To produce a visual analysis of the data, a Jupyter Notebook was created. The daily data was aggregated by sale type. Total sales and transactions were obtained by summing all the other sale type values.

As expected, all the time series visualisations of eat-in and take-out sales and transactions (seen in appendixes 8.2, 8.3, 8.4, and 8.5) show null values between March and June of 2020. One can see there are many resemblances between free stands, malls, and storefronts. Though some show different patterns in terms of rises and falls in sales and transactions, none of them regain their previous eat-in behaviors after the first Portuguese lockdown. All stores went through a small period where sales and transactions slowly began to increase, only to start falling again closer to the end of the year. It is interesting to see that stores 4, 21, and 46, unlike all the others, have a drop in eat-in and take-out sales and transactions during the summertime. The common factor between all three stores is their close whereabouts to universities, showing that location can influence demand.

When it comes to delivery, during the lockdown period, most stores have a huge rise in sales and transactions. Not only services like Uber Eats and Glovo are becoming more and more available, in some parts of the city, ordering through delivery was the only way to get a taste of fast food during the lockdown. Note, however, that as one can see in Figure 8.15 and Figure 8.17, some restaurants (8, 26, and 32) do not have any sales and transactions during the March to May lockdown. This null period comes from a business strategy. The company realized some stores were too close to others, and all deliveries could be sent out from one restaurant, instead of two. To reduce costs, some stores were completely shut down.

For unknown reasons, as can be seen in Figure 8.14 and Figure 8.18, store 39 has a period between August and December of 2019 where sales and transactions rose to never-before-seen values. After December, however, the sales and transactions went back to their previous demand. Such an unexpected change in demand can be detrimental to the algorithms.

As for the decrease in sales and transactions in store 21 (seen in Figure 8.16 and Figure 8.19), it is assumed that it happened for the same reason as the decrease in eat-in and take-out sales and transactions – the restaurant sits close to a university. Unlike stores 4 and 46, which are far from student residential locations, this surrounding area is mainly resided by the students themselves. As the university shut down, most students went back to their hometowns, causing the demand to decline.

Concerning drive sales and transactions, all 19 stores that have it (the 17 free stands, mall 12, and storefront 34) showed an increase in sales during the lockdown. As can be seen in 8.8 and 8.9, all restaurants reach levels of demand higher than ever in their past. Though storefront 34 slowly regains its previous patterns, all the other restaurants go back to their previous behavior rather quickly. As one will be able to see later in this report, this return of known behavior will lead to much better prediction results than any of the other sale types.

And finally, when it comes to total sales, as one can assume given the analysis above, all stores had a decrease in their sales and transactions during the year 2020. Stores like mall 3, which do not have a drive-through, and that were closed for delivery, have a null March to May period, making no

revenue during that time. With such an unstable period, the algorithms struggled to predict total values for most stores.

3.4.2. Outlier detection and analysis

As mentioned in 2.5.1, detecting and treating outliers can improve a model's predictions. Though Facebook's Prophet is not as sensitive to outliers, the ARIMA model can be influenced by them, so it was decided that outliers should be detected and analysed, as they might impact predictions. One must keep in mind that outliers can provide useful information and should not always be removed. It was considered that outliers found to match special days should be kept so that the models received information on their demand effect. To identify such abnormal values Python's Orion library was used. This package is very user-friendly and contains different detection techniques. For this particular analysis, the TadGAN (Time Series Anomaly Detection using Generative Adversarial Networks) model was chosen. This model was developed by Geiger et al. (2020) and combines deep learning approaches with GAN ones.

Below, two tables are presented – each of them presents the average percentage of outliers in each time series for each sale type. The first relates to sales and the second to transactions.

Sale Type	Average percentage of outliers (after removing special days)
Total	0.49%
Delivery	0.65%
Drive	0.72%
Eat-In	0.43%
Take-Out	0.63%

Table 2 – Average percentage of outliers (after removing special days) for sales time series, for each sale type

Sale Type	Average percentage of outliers (after removing special days)
Total	0.45%
Delivery	0.52%
Drive	0.57%
Eat-In	0.29%
Take-Out	0.65%

Table 3 – Average percentage of outliers (after removing special days) for transactions time series, for each sale type

Given that each store has a very low percentage of outliers that are not special days, and that most of those match the beginning of the March to May 2020 lockdown, no special treatment was applied, and all detected outliers were kept in the dataset.

3.5. Development and evaluation of forecasting algorithms

After the gathering of the functional considerations stated in 3.3 and an in-depth analysis of all time series, the next step became the development of forecasting algorithms. Due to research, and experience in previous similar use cases, it was decided two models should be tested and compared in terms of prediction performance. Before inserting any data into the algorithms, a division was done between train, validation, and test sets. The test dates had previously been defined by the client, as mentioned in 3.1.4. The developed pipeline should be tested by comparing forecasts and actual values for the months of June, July, August, September, and October of 2020. Eventually, the client hopes to invest in a system capable of generating daily predictions for the 7 following days. To mimic such a tool, 7 days of the test dates were removed. The historical data previous to those was used for training and validation. The first set was used to test different parametrizations, and the second to evaluate those. The best parameters (and most important external features, in Prophet's case) were then employed to predict the 7 test days. The corresponding MAPE was stored, and the first of the 7 dates became part of the validation set. The first date of the validation set became part of the training set. The algorithm then predicted the following 7 days, and MAPE values were again stored. This process was repeated every day for the 5 testing months, and MAPE values were averaged out to evaluate performances. A diagram of this process is shown in Figure 3.2.

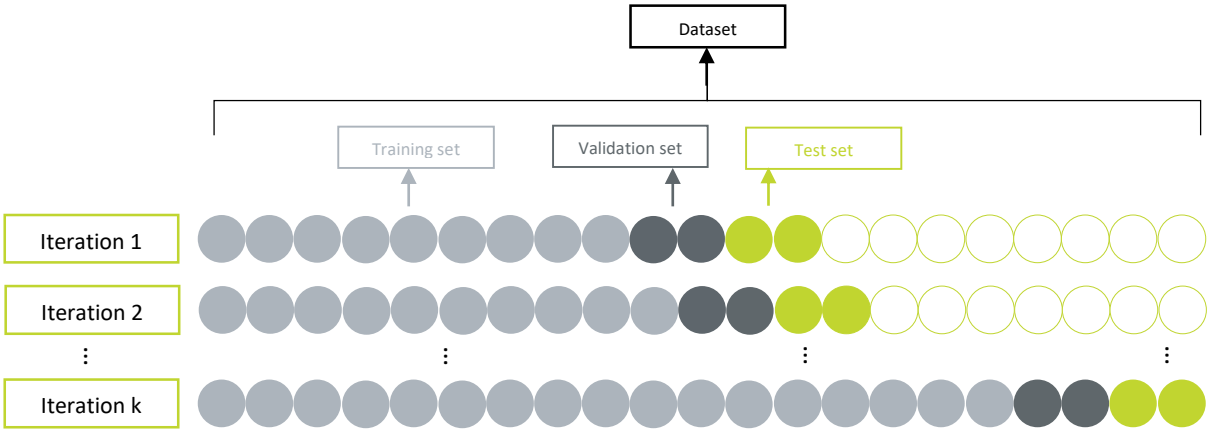


Figure 3.2 – System validation diagram

3.5.1. ARIMA

As explained in 2.3, the ARIMA model needs three parameters (p, d, q) to predict future demand. Besides the fact that these parameters may change over time, it is impossible to analyse over 400 time series manually. Because of that, the model was implemented using Python's pmdarima package. This library applies an Auto ARIMA model, automatically finding the optimal p, d, and q values. Since the PoC timeline was limited, it was decided to randomly choose 3 stores (a free stand, a mall, and a storefront), a prediction segment, and a test month, to examine how often the p, d, and q values must be reviewed. The chosen stores were numbers 26, 34, and 39; the segment was total transactions, and the test month was October. Below, the results of various tests can be consulted.

Store	Daily		Every 7 days		Every 10 days	
	MAPE	*Time*	*MAPE*	*Time*	*MAPE*	*Time*
26	**13.77%**	8.3 h	13.87%	1.6 h	14.32%	30 min
34	12.08%	5.6 h	12.06%	36 min	12.14%	14 min
39	14.24%	6.2 h	8.6%	52 min	9.29%	36 min
Store	**Every 15 days**		**Every 20 days**		**Every 30 days**	
	MAPE	*Time*	*MAPE*	*Time*	*MAPE*	*Time*
26	14.03%	31 min	13.95%	21 min	14.03%	13 min
34	**11.70%**	25 min	12.28%	16 min	12.29%	7 min
39	**8.45%**	32 min	8.96%	15 min	9.29%	16 min

Table 4 – Running time and average MAPE when reviewing ARIMA's parameters every X days

Though the mall (26) had better performance when replacing the p, d, and q values daily, the free stand (39) and the storefront (34) had better performance when doing it every 15 days. Note that the malls suffered a bigger impact on their trend than the other 2 store types due to the Portuguese lockdown measures, making their parametrization more complex. Given the limited time, it was decided to review parameters every 15 days for every store, no matter which store type it belonged to.

Knowing the model's parameters must be reviewed every 15 days, it was also important to test different lengths for the validation period. Usually, the validation period is the one immediately before the testing one. However, the team has found that in forecasting projects, validating parameters in the dates of the previous year that correspond to the test set dates (i.e., validating parameters in the 1st to 15th of October of 2019 to test the 1st to the 15th of October of 2020) sometimes brings better results. Given the change in behavior from 2019 to 2020 in almost all time series, and with no time to review them all, the second option was discarded. Therefore, there was an attempt to use a 10-fold method, where every 15 days, the parameters were tested in 10 different periods, and the combination that led to the lowest average error was chosen. Yet, the predictions were taking too long to generate, and the 10-fold had to be reduced to a 5-fold.

Finally, external variables were added to the model. The ones related to the weather did not improve the results. The others, like active campaigns, investments, and awareness, were impossible to add to the model. Python's ARIMA model is not capable of generating forecasts if the external variables present the same value throughout the testing period. For example, as one can see in Table 5, in the first week of august, store 39 has the same values for active campaigns.

Date	Store Number	Sale Type	Transactions	Sales	Active Campaigns	Awareness	Investment
8/1/2020	39	Drive	664	5725.347	3	59	646
8/2/2020	39	Drive	608	5318.283	3	59	646
8/3/2020	39	Drive	591	4474.282	3	60	486
8/4/2020	39	Drive	551	4275.856	3	60	486
8/5/2020	39	Drive	634	4758.08	3	60	486

8/6/2020	39	Drive	620	4800.96	3	60	486
8/7/2020	39	Drive	643	5354.278	3	60	486

Table 5 – Excerpt of the dataset, with store 39 having the same number of active campaigns throughout the first week of August 2020

These cases were incredibly common, and the effort of working around them was too big for the PoC duration. Therefore, a decision was made to not add any external variables to the ARIMA model.

3.5.2. Prophet

As mentioned in 2.4, Prophet allows the user to introduce specific parameters. Before starting parameter testing, the default automatization was kept, and the month of October was chosen to, just like with the ARIMA model, test how often parameters should be reviewed. This time, instead of choosing 3 restaurants and 1 forecasting segment, all of them were used for testing. Moreover, only 3 attempts were made: one with revision every 7 days, another every 15 days, and a third every 30 days. Though adjustment every 7 days showed better performance, it was decided to use tuning every 30 days, due to the trade-off between error and time.

Again, different validation set lengths were tested. In Prophet's case, just two approaches were taken: one with validation set length equaling 7 days, and another equaling 15 days. As expected, given the disruptive behavior shown by most time series during the lockdown, the best results came from a validation set length of 7 days.

After establishing that parameters should be tuned every 30 days and based on the past 7 days, it was time to check if Prophet's automatization indeed provided the lowest forecasting error. To test that, the following options were explored:

- Turn yearly seasonality on and off.

Input different values for the following parameters:

- changepoint_range (controls the proportion of the historical data in which the trend can change).
- changepoint_prior_scale (controls the flexibility of the trend).
- seasonality_prior_scale (controls the flexibility of the seasonality).
- features_default_prior_scale (controls the flexibility of the external variables).

The forecasts showed improvement, and so, when doing parameter adjustment every 30 days, the algorithm began choosing between different (inputted) values for each parameter, instead of automatically finding them.

When it came to adding external variables, the process was much easier than the ARIMA one. For each store, Prophet ran a model for each combination of the external variables, namely:

- The number of active campaigns in each store, in each day.
- Average daily temperature in Lisbon (ºC).
- Average daily humidity in Lisbon (%).

- Average daily wind speed in Lisbon (m/s).
- Weekly investment in communication channels (GRPs).
- Weekly customer awareness (%).

The combinations ran for both total transactions and total sales. Because time was limited, the external variables that led to the lowest error were stored for each restaurant, for transactions and sales, and were then used for the other sale types. Even with this "shortcut", the results improved, and the approach was kept. Below, one can see the combinations of parameters that led to the lowest forecasting errors for the total sales, and the number of stores for which they were chosen.

Combination of parameters	Number of stores	Percentage of stores
Active Campaigns	2	4.2%
Humidity	2	4.2%
Investment	2	4.2%
Temperature	9	18.8%
Active Campaigns, Humidity, Temperature, Wind	8	16.7%
Active Campaigns, Investment	19	39.6%
Humidity, Temperature, Wind	4	8.3%
Investment, Humidity, Temperature, Wind	2	4.2%

Table 6 – Number of times each combination of features generated the best forecasts for total sales

Both Prophet and ARIMA were set to run in parallel in the same Databrick's cluster. Their application, performance, and outcomes are presented in 4.

4. RESULTS AND DISCUSSION

After developing two algorithms, it was time to generate forecasts for the months of June to October 2020. The results were compared to the real demand values, and the MAPE was calculated for each store. At the beginning of the project, it was thought that out of the two algorithms, one would be chosen. When the evaluation took place, it was obvious that the best results came from a combination of the two. The presented errors came from choosing the better-performing algorithm for each time series. When implementing the system, there will not be actual values for comparison, and new testing will have to take place. Most likely, the predictions from each algorithm will be stored, and after the real values are fed into the system, errors will be calculated, and an algorithm will be chosen for the next batch of predictions. Both algorithms will again be used, and their results will be stored, but the results displayed for the client's consultation will be those that came from the chosen algorithm. Results for each segment and each store can be seen in Table 7 and Table 8.

Store Number	**Total**		**Delivery**		**Drive**	
	Sales	*Transactions*	*Sales*	*Transactions*	*Sales*	*Transactions*
1	10.12%	10.57%	12.25%	11.14%		
2	9.25%	8.49%	14.91%	13.89%		
3	14.97%	13.31%	21.86%	18.54%		
4	12.44%	11.67%	16.25%	16.78%		
5	14.48%	12.37%	17.29%	15.74%		
6	12.41%	11.60%	25.76%	22.30%		
7	16.28%	17.82%	69.04%	57.23%		
8	18.76%	15.94%	13.58%	11.88%		
9	11.48%	10.60%	14.95%	13.97%		
10	7.80%	7.12%	18.24%	15.96%	7.19%	5.60%
11	11.92%	11.81%	18.35%	16.75%	11.69%	10.04%
12	10.54%	10.65%	12.93%	10.62%	15.85%	12.17%
13	17.84%	18.30%	11.51%	10.60%		
14	10.07%	9.71%	16.63%	14.69%		
15	7.02%	6.89%	15.25%	13.11%	7.13%	5.45%
16	8.83%	7.04%	16.39%	15.11%	7.79%	5.87%
17	11.15%	9.06%	14.47%	13.56%		
18	11.28%	10.33%	16.59%	15.31%		
19	10.89%	9.69%	18.14%	16.73%		
20	10.98%	9.64%	10.71%	9.31%		
21	12.30%	11.16%	14.69%	12.28%		
22	10.39%	10.44%	12.95%	11.91%		

23	6.69%	6.01%	48.58%	41.28%	5.49%	4.63%
24	13.98%	12.34%	16.39%	13.18%		
25	7.85%	6.79%	13.70%	12.19%	8.29%	7.25%
26	19.01%	16.59%	13.71%	12.20%		
27	7.74%	6.73%	17.40%	15.59%	7.71%	5.95%
28	7.96%	7.42%	18.26%	17.61%	9.04%	6.02%
29	11.74%	9.53%	22.80%	20.40%		
30	9.27%	8.25%	13.90%	12.25%	9.27%	7.51%
31	17.52%	15.04%	12.98%	11.13%		
32	11.73%	11.93%	18.07%	16.68%		
33	9.46%	7.44%	15.79%	14.07%	9.96%	7.91%
34	7.99%	6.77%	14.90%	14.49%	8.41%	7.14%
35	8.78%	7.47%	16.68%	14.75%	7.17%	6.34%
36	13.23%	11.33%	24.30%	19.67%	8.56%	8.06%
37	8.88%	8.80%	15.26%	14.00%	8.14%	6.32%
38	9.57%	7.02%	16.29%	14.94%	8.52%	7.78%
39	8.09%	6.48%	14.54%	12.58%	6.91%	5.40%
40	10.20%	9.10%	16.54%	14.57%		
41	14.84%	13.37%	16.15%	14.48%		
42	9.77%	9.21%	17.02%	15.72%	8.52%	7.34%
43	10.77%	9.18%	20.90%	19.47%		
44	13.21%	11.20%	14.91%	13.48%		
45	13.81%	12.22%	16.63%	15.28%		
46	10.84%	10.96%	12.25%	11.14%	10.18%	6.55%
100	21.34%	14.33%	14.91%	13.89%		
101	11.42%	10.23%	21.86%	18.54%		

Table 7 – Final average MAPE for each store in total, delivery and drive sales and transactions

Store Number	**Eat-In**		**Take-Out**	
	Sales	*Transactions*	*Sales*	*Transactions*
1	15.87%	13.92%	16.76%	14.58%
2	16.05%	11.97%	15.07%	14.95%
3	14.85%	12.29%	35.36%	16.24%
4	17.54%	16.97%	13.69%	12.82%
5	63.93%	27.15%	16.82%	14.49%
6	63.81%	28.43%	16.82%	14.00%

7	19.68%	16.59%	21.80%	15.91%
8	19.84%	17.08%	22.84%	17.83%
9	16.49%	15.81%	17.83%	17.53%
10	15.43%	14.19%	22.46%	20.47%
11	16.47%	14.88%	28.23%	25.55%
12	15.16%	12.28%	20.08%	14.98%
13	22.92%	16.17%	33.63%	27.39%
14	24.35%	21.60%	15.59%	14.34%
15	13.46%	10.81%	22.74%	21.46%
16	16.02%	11.57%	24.56%	19.59%
17	14.00%	12.15%	13.16%	11.39%
18	60.46%	18.31%	20.30%	16.96%
19	19.98%	11.74%	15.69%	14.39%
20	25.54%	12.76%	14.13%	13.32%
21	49.95%	15.61%	14.24%	14.23%
22	15.17%	15.39%	18.52%	16.61%
23	25.22%	11.46%	21.06%	17.48%
24	16.94%	13.98%	20.19%	18.48%
25	14.15%	13.36%	25.89%	21.94%
26	60.02%	35.53%	28.60%	24.86%
27	13.00%	11.30%	22.08%	16.90%
28	15.30%	13.31%	22.86%	17.96%
29	19.89%	12.37%	13.72%	13.15%
30	14.12%	11.26%	24.25%	20.49%
31	21.32%	15.81%	27.80%	27.30%
32	33.03%	15.48%	20.57%	18.23%
33	10.97%	19.45%	22.27%	17.05%
34	12.78%	11.12%	19.55%	15.03%
35	13.95%	12.82%	22.62%	20.04%
36	21.17%	20.08%	32.43%	31.75%
37	14.50%	12.55%	21.00%	17.07%
38	11.44%	10.07%	25.95%	19.85%
39	21.61%	10.53%	21.17%	20.38%
40	45.32%	12.56%	15.90%	13.78%
41	37.04%	16.97%	19.96%	16.79%
42	13.08%	13.61%	22.83%	15.99%

43	28.83%	11.84%	16.15%	12.23%
44	17.40%	14.17%	15.72%	13.00%
45	16.13%	13.10%	20.76%	18.42%
46	15.17%	14.25%	34.36%	30.18%
100	24.13%	15.79%	53.97%	32.08%
101	37.17%	17.94%	19.16%	16.92%

Table 8 – Final average MAPE for each store in eat-in and take-out sales and transactions

The MAPE values for all 5 months are averaged in Table 9.

Sale Type	Sales	Transactions
Total	11.6%	10.4%
Delivery	18.1%	16.1%
Drive	8.7%	7.0%
Eat-In	23.6%	15.2%
Take-Out	21.9%	18.3%

Table 9 – Average MAPE for each sale type, for the months of June to October 2020

The results pretty much matched the team's expectations, with eat-in and take-out having the worst MAPE. Delivery had a spike in demand, so it was predictable that its errors would not be great either. The total sales and transactions performed better than anticipated, as well as drive, that ended up with great outcomes. As mentioned in 3.3, stores 18 and 101 were kept in representation of the same store. Store 101, the one with the larger amount of data, had better results than store 18. But that did not happen in all prediction segments – surprisingly, delivery predictions were closer to the actual demand when using less historical data.

To understand how the MAPE changed according to region and store type, one can check Table 10 and Table 11. There seems to be no region easier to predict. Lisbon shows better performance in delivery and drive, but Greater Lisbon performs better in the other sale types. When it comes to store types, and as expected, the malls show the worst performance in all sale types except take-out. Greater Lisbon's malls performed worse than the Lisbon ones, especially in terms of delivery sales and transactions.

		Total		**Delivery**		**Drive**	
Region	**Store Type**	*Sales*	*Transactions*	*Sales*	*Transactions*	*Sales*	*Transactions*
Lisbon	Free Stand	9.09%	7.81%	14.20%	12.71%	8.48%	6.75%
Lisbon	Mall	12.86%	11.81%	16.61%	15.03%		
Lisbon	Storefront	13.15%	11.39%	15.56%	14.26%	8.41%	7.14%
Greater Lisbon	Free Stand	9.03%	8.12%	19.65%	17.37%	8.28%	6.70%

Greater Lisbon	Mall	13.12%	12.11%	22.43%	19.44%	15.85%	12.17%
Greater Lisbon	Store front	12.44%	11.37%	14.52%	12.60%		
Lisbon	All	12.32%	10.87%	15.65%	14.23%	8.46%	6.82%
Greater Lisbon	All	10.88%	9.93%	20.28%	17.79%	8.82%	7.09%
All	Free Stand	9.04%	8.05%	18.36%	16.28%	8.33%	6.71%
All	Mall	13.02%	11.99%	20.03%	17.62%	15.85%	12.17%
All	Storefront	12.99%	11.39%	15.34%	13.91%	8.41%	7.14%

Table 10 – Average MAPE for each region and store type in total, delivery and drive sales and transactions

		Eat-In		**Take-Out**	
Region	**Store Type**	*Sales*	*Transactions*	*Sales*	*Transactions*
Lisbon	Free Stand	15.59%	12.05%	26.84%	23.09%
Lisbon	Mall	25.03%	16.50%	22.08%	17.67%
Lisbon	Storefront	25.88%	14.26%	21.87%	18.08%
Greater Lisbon	Free Stand	15.59%	13.64%	23.80%	20.14%
Greater Lisbon	Mall	35.54%	18.85%	18.65%	15.36%
Greater Lisbon	Storefront	16.74%	14.02%	17.56%	15.35%
Lisbon	All	23.74%	14.57%	22.84%	18.86%
Greater Lisbon	All	23.66%	15.76%	21.14%	17.72%
All	Free Stand	15.59%	13.26%	24.52%	20.83%
All	Mall	31.22%	17.88%	20.06%	16.31%
All	Storefront	23.92%	14.21%	20.95%	17.49%

Table 11 – Average MAPE for each region and store type in eat-in and take-out sales and transactions

To see the distribution of the MAPE by day of the week, the plot of Figure 4.1 was generated. As one can see, the hardest day of the week to predict is Monday. In all sale types, Monday performs the worst between all weekdays, with take-out sales showing the highest MAPE. Regarding the most predictable weekday, the answer is not as simple. Eat-in sales present lower forecast errors on

Tuesdays and delivery transactions on Thursdays. The rest show better performance during the weekend.

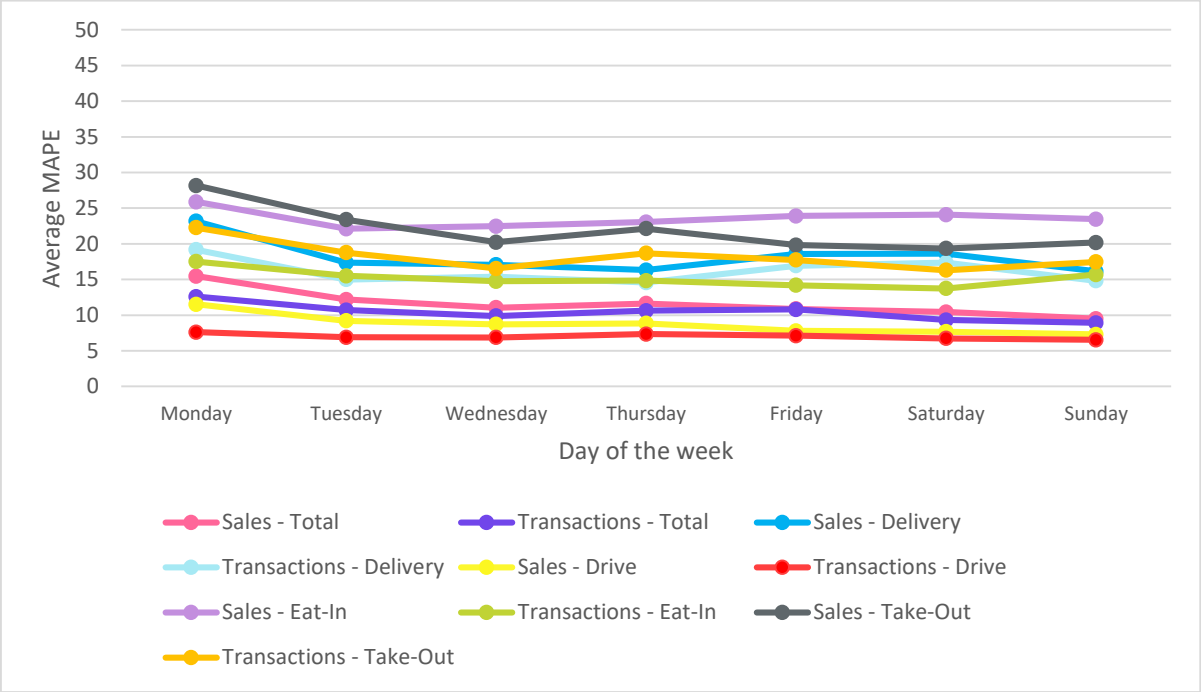


Figure 4.1 – MAPE distribution by day of the week

A short analysis on the MAPE distribution by month was also conducted, and, as can be comprehended by looking at Figure 4.2, June and October were the two worst-performing months in all prediction segments. As mentioned in 3.1.4, the client had opened the possibility of removing the months of June and October. To check their impact on the average errors, new calculations were made after their removal. The results can be seen in Table 12.

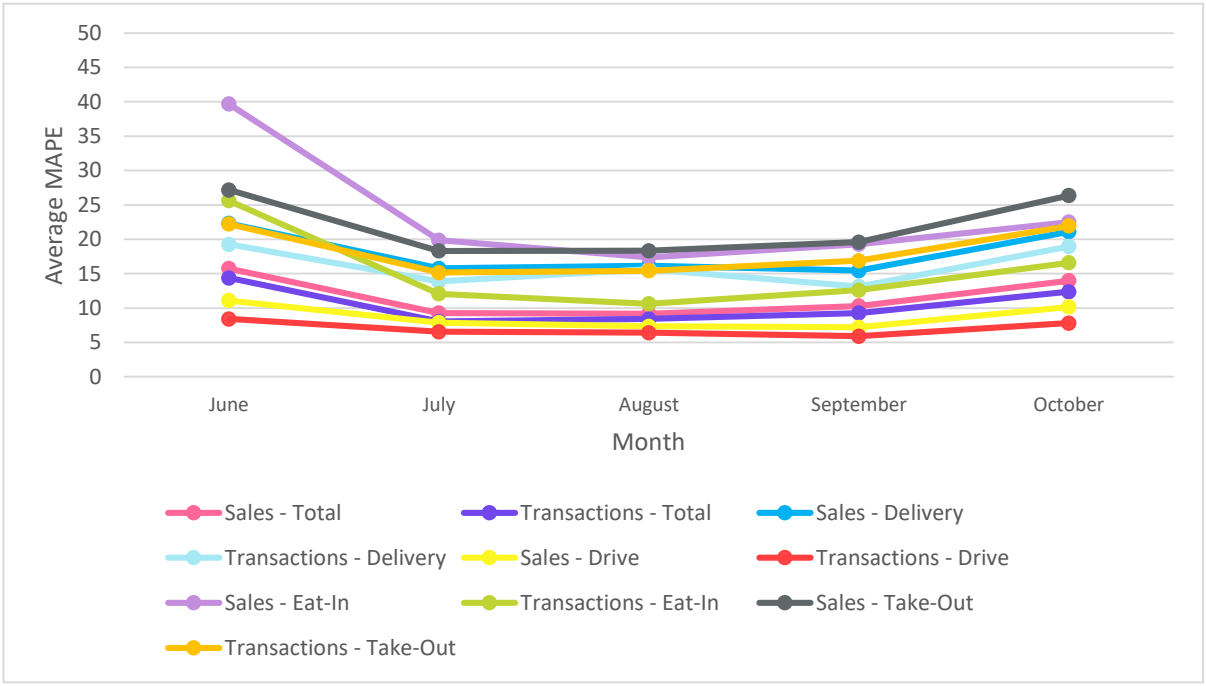


Figure 4.2 – MAPE distribution by month

Sale Type	Sales	Transactions
Total	9.6%	8.6%
Delivery	15.8%	14.2%
Drive	7.5%	6.3%
Eat-In	18.8%	11.7%
Take-Out	18.7%	15.8%

Table 12 – Average MAPE for each sale type, for the months of July to September 2020

Removing the months of June and October obviously improved all results, corroborating the theory that the COVID-19 measures and case numbers probably influence the demand in the stores.

Concerning algorithm distribution, there was not one that overshadowed the other. As one can see in Table 13, ARIMA seems to produce better results in more disruptive segments. Assuming demand will normalize in the next few years, Prophet might become more important in producing the franchise's forecasts.

Sale Type	Sales		Transactions	
	ARIMA	*Prophet*	*ARIMA*	*Prophet*
Total	26	13	38	11
Delivery	13	32	9	36
Drive	6	13	8	11
Eat-In	34	15	46	3
Take-Out	23	26	32	17

Table 13 – Number of times each algorithm showed better performance in each prediction segment

Given the fact that the algorithms' performances were highly affected by the COVID-19 pandemic, the client was very interested in seeing a present-day performance. It was agreed that the Proof of Concept would expand for a couple of more months, with the team generating predictions every week and feeding them to the Qlik Sense dashboard for the client's usage. Weekly average MAPE can be consulted in 8.12. The overall performance can be seen below, in Table 14. It is important to note that some days presented problems in their data. For example, store 14 reaches over 2000€ in eat-in sales on July 16th but makes less than 1€ in the two following days. Other restaurants had the same problem, leading to the removal of days 10, 11, 17, and 18 of July from the error calculations.

Sale Type	Sales	Transactions
Total	12.4%	10.7%
Delivery	19.1%	16.1%
Drive	11.5%	9.3%
Eat-In	21.5%	19.9%
Take-Out	25.7%	20.8%

Table 14 – Average MAPE for each sale type, from May 29th until July 16th

At the end of 2020, a new lockdown measure was put in place – most stores and restaurants should be closed before 1 pm every weekend. Previously, Saturday and Sunday used to show the lowest forecast errors in almost every segment. To see if the weekends continued their predictability, a new visual analysis was conducted.

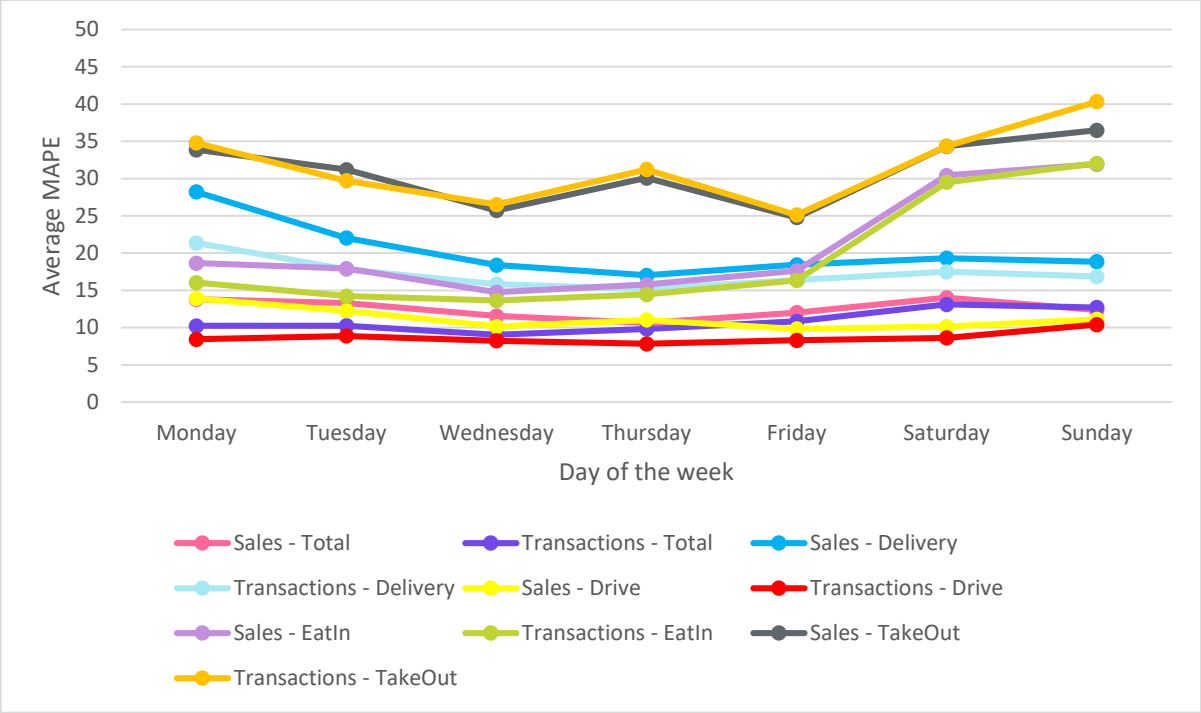


Figure 4.3 – MAPE distribution by day of the week

Unlike before, the plot of the MAPE distribution by day of the week (Figure 4.3), shows that weekends became the most un-predictable weekdays in the majority of the segments. To see how much those days impacted the outcomes, new MAPE values were calculated, after removing the weekends. Results can be seen in Table 15.

Sale Type	Sales	Transactions
Total	12.1%	9.9%
Delivery	18.7%	15.6%
Drive	10.9%	8.4%
Eat-In	17.2%	14.7%
Take-Out	24.8%	19.3%

Table 15 – Average MAPE for each sale type, from May 29th until July 16th (excluding weekends)

As expected, the removal of the weekend forecasts did improve the results. Again, it seems like the different lockdown measures put in place definitely impacted demand, both in 2020 and in 2021.

Even though the Qlik Sense dashboard development is out of the scope of the internship, it is considered an important part of the project, as it became the way for the client to visualise all the work done by the Data Science consultants. A snapshot of the dashboard, developed by the Business Intelligence specialists, can be seen in 8.13.

5. CONCLUSIONS

The forecasting project kicked off in April 2021 and began with a dive into the business strategies of the franchise. Various meetings with the client allowed Noesis to align expectations, and to understand their goals. It was defined from the beginning that the long-run objective was to have a system capable of making forecasts for the company's demand. To prove such a concept, 10 prediction segments were chosen (Total/Delivery/Drive/Eat-in/Take-Out x Sales/Transactions), and 48 stores from Lisbon and Greater Lisbon were selected. The process opened with a data analysis phase, which quickly led to the realization that the COVID-19 pandemic would more than likely complicate the forecasts. Lots of time series seemed disruptive, with long periods of nonexistent demand. Two algorithms were tested (ARIMA and Facebook's Prophet), and it was expected that one would be chosen to produce all future forecasts. Since both showed better results in different segments, a hybrid system was created. Predictions were then sent to the BI experts and incorporated into a dashboard. As the process went on, the client began feeling more curious about how the system would behave in the present time, so an "after-PoC" stage was conducted, with forecasts being generated each week, from the end of May until mid-July 2021.

The internship allowed the usage of many concepts learned in the first year of the Master's in Data Science and Advanced Analytics. The whole system was built in Databricks, a software that became familiar in the Big Data Analytics course. The forecasting algorithms, especially ARIMA, had been introduced in the Statistics course, and the programming side of the project joined knowledge from various other courses. New knowledge was acquired, especially when it came to the business operations, both of Noesis, as a consulting company, and the client's franchise, as a fast-food chain. New algorithms were explored along with different software. Overall, the experience fully related to the Master's concepts and opened the door to future opportunities.

6. LIMITATIONS AND RECOMMENDATIONS FOR FUTURE WORKS

Though the proof of concept ended on a successful note, there is still a lot of space for improvement. The biggest setback was time, which happened mainly due to the fact that all code was developed using solely Python. The algorithm packages that were used are yet to be adapted for PySpark data frames, making the processes much slower. In the future, the team should look to over-ride non-compatible methods by adapting the packages for PySpark data frames. This procedure will allow for much faster development, as Databricks is built on top of Spark, and can automatically parallelize code.

If the PoC moves on to a project to be deployed, an engineering solution will have to be implemented to make the complete process automatic. The client will be able to choose between an on-premises solution (shown in Figure 6.1) or a cloud solution (shown in Figure 6.2).

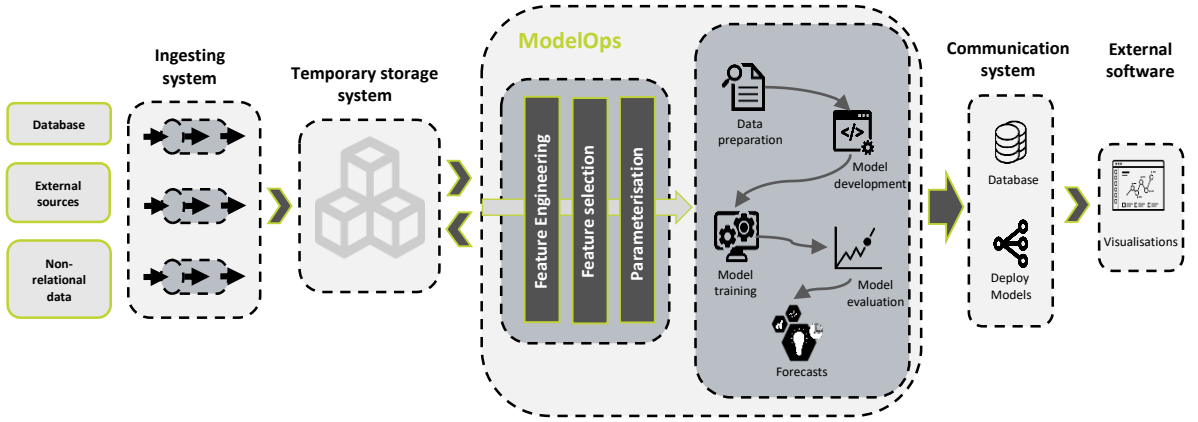


Figure 6.1 – Proposed on-premises architecture

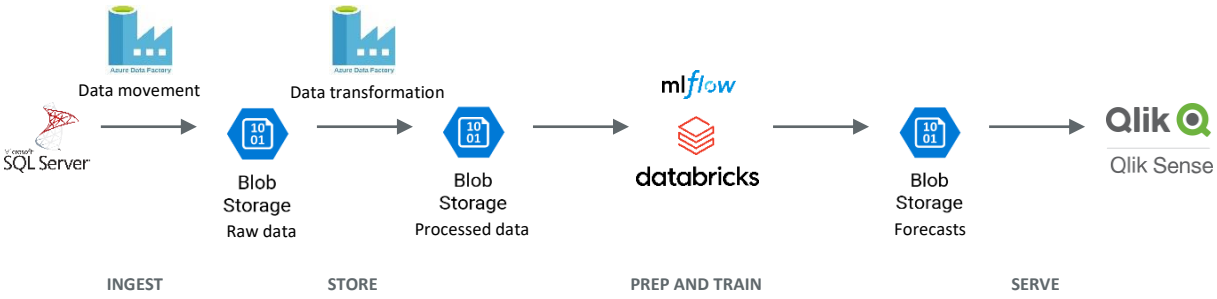


Figure 6.2 – Proposed cloud architecture

Moreover, it is advised that in future developments, the outlier detection technique be reviewed, and that outlier treatment should be considered. Normalization of numerical variables should also be evaluated.

Besides, more business knowledge is important - for instance, the work around external variables is considered incomplete, as many more variables can impact demand: school calendars, the distance between stores, football data (derbies might influence delivery demand, and even all sale types in stores close to stadiums). The hypothesis of adding them to the ARIMA model must also be re-considered.

So far, time series were fed into the algorithms one by one, not taking into account the other sale types sales, and transactions. The possibility of considering other time series when forecasting another should be investigated. New algorithms can be used, such as XGBoost Regressor. Until now, the two algorithms were generating forecasts for all prediction segments. It is possible that each algorithm works better for a specific type of store. Clustering restaurants might reduce the number of forecasts, as the models can possibly be assigned to each cluster. This process, however, must also be automatized, as stores will continue to open and close.

Finally, and most significantly, it is believed that the results will only improve when the pandemic situation has either stabilized, or the day-to-day of the Portuguese people goes back to what it was before. Until the routines of customers become firm, and lockdown measures stop being reviewed so often, it is not expected that the forecasting errors diminish significantly.

7. BIBLIOGRAPHY

Blecher, L. (2004). Using forecasting techniques to predict meal demand in Title IIIc congregate lunch programs. *RESEARCH: RESEARCH AND PROFESSIONAL BRIEFS, 104(8),* 1281-1283. https://doi.org/10.1016/j.jada.2004.05.209

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time Series Analysis: Forecasting and Control* (5th ed.). John Wiley and Sons Inc.

Brockwell, P. J., & Davis, R. A. (2002). *Introduction to Time Series and Forecasting* (2nd ed). Springer Nature. https://doi.org/10.1007/b97391

Alexander, C. (2008). *Moving average models for volatility and correlation, and covariance matrices*. Handbook of finance, 3. Wiley Press.

Geiger, A., Liu, D., Alnegheimish, S., Cuesta-Infante, A., & Veeramachaneni, K. (2020). TadGAN: Time Series Anomaly Detection Using Generative Adversarial Networks. BigData-2020. *In Proceedings of IEEE International Conference on Big Data*. https://arxiv.org/pdf/2009.07769.pdf

Gujarati, D. N., & Porter, D. C. (2009). *Basic Econometrics* (5th ed). McGraw-Hill.

Gujarati, D. N. (2003). *Basic Econometrics* (4th ed). McGraw-Hill.

Harvey, A. C., & Shephard, N. (1993). Structural time series model. *Handbook of Statistics, 11*, 261–302.

Harvey, A., & Peters, S. (1990). Estimation procedures for structural time series models. *Journal of Forecasting, 9*(2), 89-108. https://doi.org/10.1002/for.3980090203

Ho, S. L., & Xie, M. (1998). The Use of ARIMA Models for Reliability Forecasting and Analysis. *Computers & Industrial Engineering, 35*(1-2), 213-216. https://doi.org/10.1016/S0360-8352(98)00066-7

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice*. (2nd ed). OTexts: Melbourne, Australia. https://OTexts.com/fpp2

Krajewski, J., Ritzman, B., & Malhotra, M. (2015). *Operations Management: Processes and Value Chains* (11h ed). Pearson Prentice Hall.

Lasek, A., Cercone, N., & Saunders, J. (2016). Restaurant Sales and Customer Demand Forecasting: Literature Survey and Categorization of Methods. *Smart City 360° – First EAI International Summit: Revised Selected Papers, Bratislava, Slovakia and Toronto, Canada*, 479-491. https://doi.org/10.1007/978-3-319-33681-7_40

Liu, L., Bhattacharyya, S., Sclove, S. L., Chen, R., & Lattyak, W. J. (2001). Data mining on time series: an illustration using fast-food restaurant franchise data. *Computational Statistics & Data Analysis*, *37*(4), 455-476. https://doi.org/10.1016/S0167-9473(01)00014-7

Makridakis, S., & Hibon, M. (2000). The M3-competition: results, conclusions and implications. *International Journal of Forecasting, 16*(4), 451–476. https://doi.org/10.1016/S0169-2070(00)00057-1

Meneghini, M., Anzanello, M., Kahmann, A., & Tortorella, G. (2018). Quantitative demand forecasting adjustment based on qualitative factors: case study at a fast food restaurant. *Systems & Management, 13*, 68-80. https://doi.org/10.20985/1980-5160.2018.v13n1.1188

Miller, J. L., McCahon, C. S., & Bloss, B. K. (1990). Food Production Forecasting with Simple Time Series Models. *Journal of Hospitality & Tourism Research, 14*(3), 9–21. https://doi.org/10.1177/109634809101400303

Montgomery, C. D., Jennings, C. L., & Kulahci, M. (2015). *Introduction to Time Series Analysis and Forecasting* (2nd ed). John Wiley & Sons Inc.

Pollock, D. S. G. (1999). *A Handbook of Time Series Analysis, Signal Processing and Dynamics*. (1st ed). Academic Press. https://doi.org/10.1016/B978-0-12-560990-6.X5000-3

Schimek, M. G., & Turlach, B. A. (1998). Additive and generalized additive models: A survey. *SFB 373 Discussion Paper* (No 1998,97).

Silver, E. A., Pyke, D. F., & Peterson, R. (1998). *Inventory Management and Production Planning Scheduling*. (3rd ed). John Wiley & Sons Inc.

Taylor, S. J., & Letham, B. (2018). Forecasting at Scale. *The American Statistician, 72*(1), 37-45. https://doi.org/10.1080/00031305.2017.1380080

Wallström, P., & Segerstedt, A. (2010). Evaluation of forecasting error measurements and techniques for intermittent demand. *International Journal of Production Economics, 128*(2), 625-636. https://doi.org/10.1016/j.ijpe.2010.07.013

Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing, 50*(17), 159–175. https://doi.org/10.1016/S0925-2312(01)00702-0

8. APPENDIX

8.1. TIME SERIES OF TOTAL TRANSACTIONS FROM 2018 TO 2020, IN STORE 9999

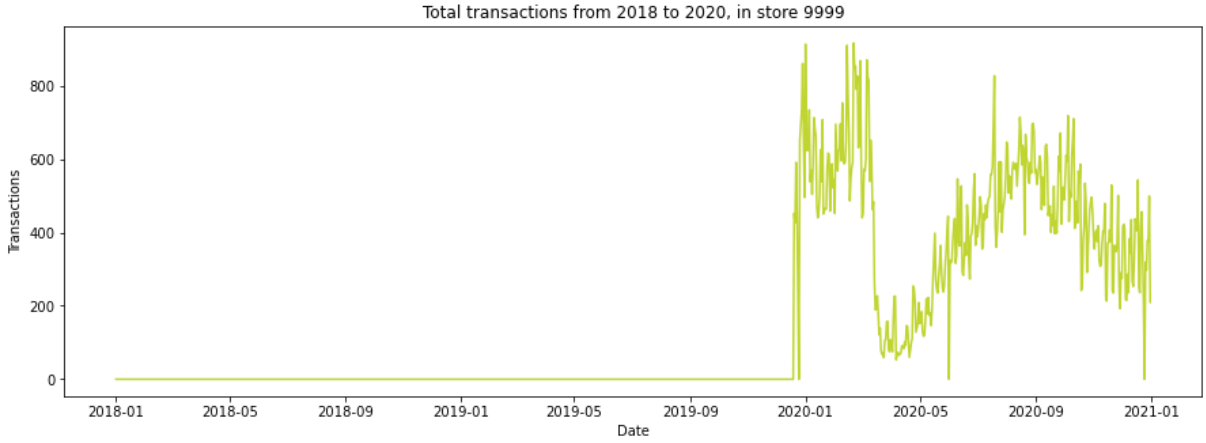


Figure 8.1 – Time series of total transactions from 2018 to 2020, in store 9999

8.2. TIME SERIES OF EAT-IN SALES FROM 2018 TO 2020

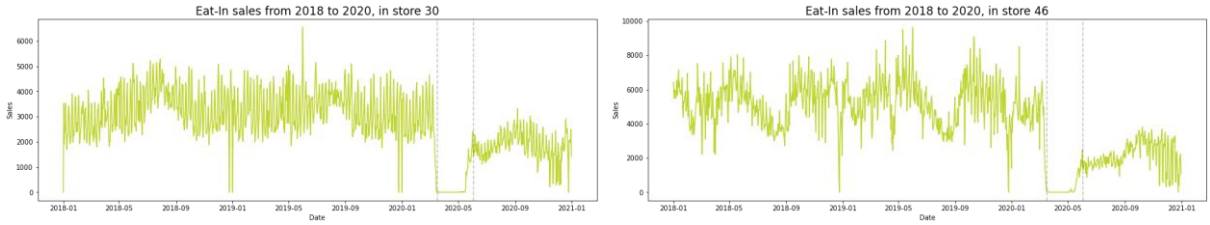


Figure 8.2 – Time series of eat-in sales from 2018 to 2020, in free stands 30 and 46

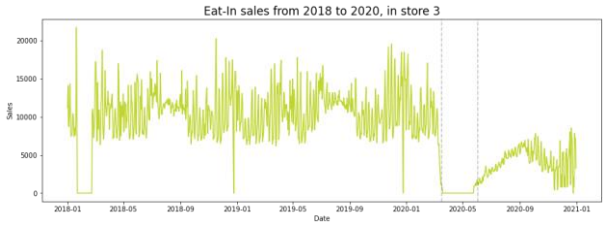


Figure 8.3 – Time series of eat-in sales from 2018 to 2020, in mall 3

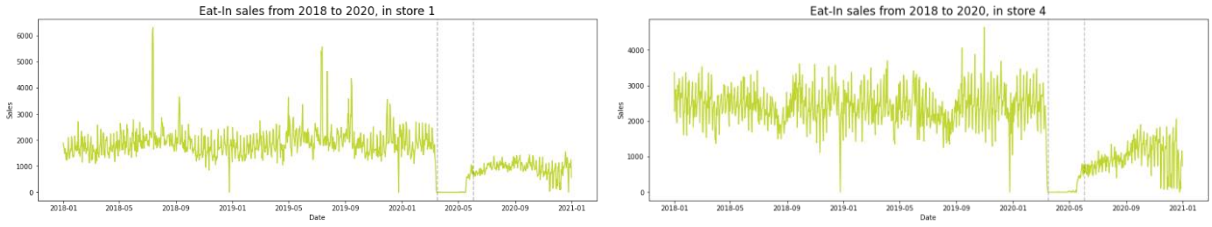


Figure 8.4 – Time series of eat-in sales from 2018 to 2020, storefronts 1 and 4

8.3. TIME SERIES OF EAT-IN TRANSACTIONS FROM 2018 TO 2020

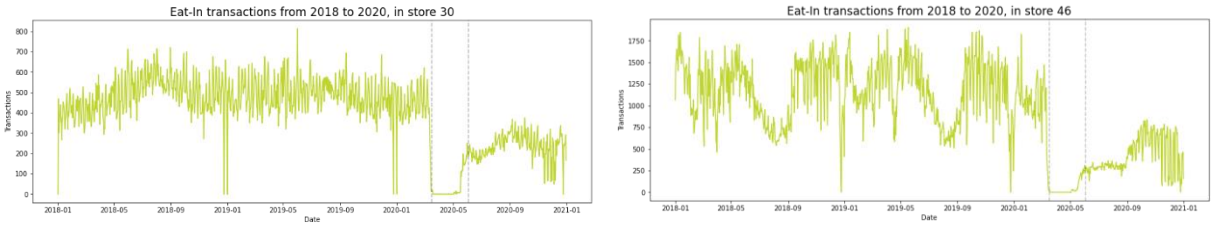


Figure 8.5 – Time series of eat-in transactions from 2018 to 2020, in free stands 30 and 46

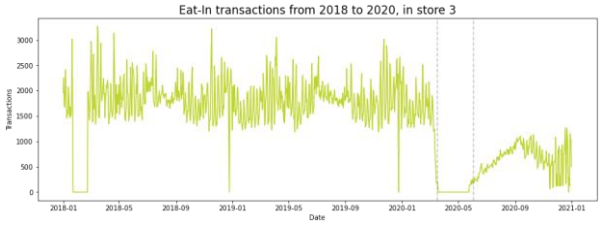


Figure 8.6 – Time series of eat-in transactions from 2018 to 2020, in mall 3

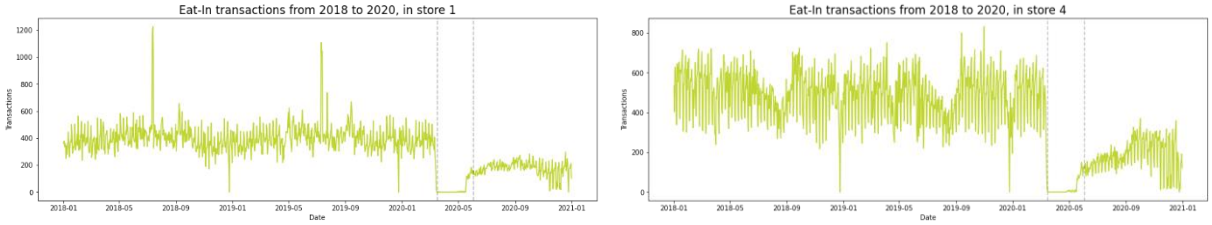


Figure 8.7 – Time series of eat-in transactions from 2018 to 2020, in storefronts 1 and 4

8.4. TIME SERIES OF TAKE-OUT SALES FROM 2018 TO 2020

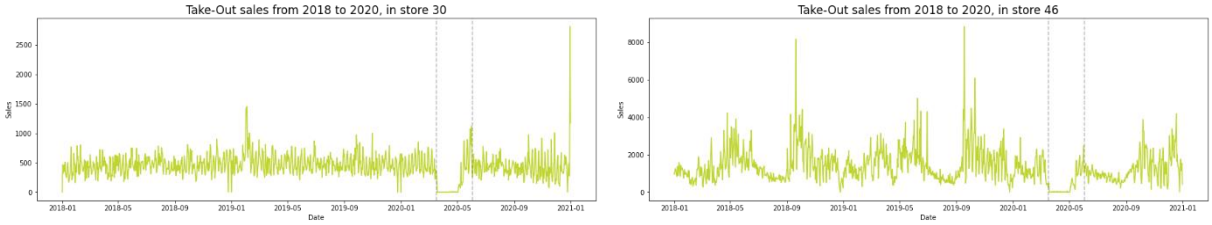


Figure 8.8 – Time series of take-out sales from 2018 to 2020, in free stands 30 and 46

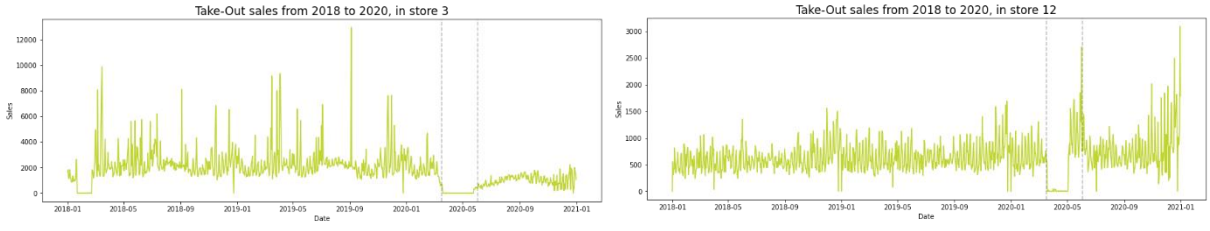


Figure 8.9 – Time series of take-out sales from 2018 to 2020, in malls 3 and 12

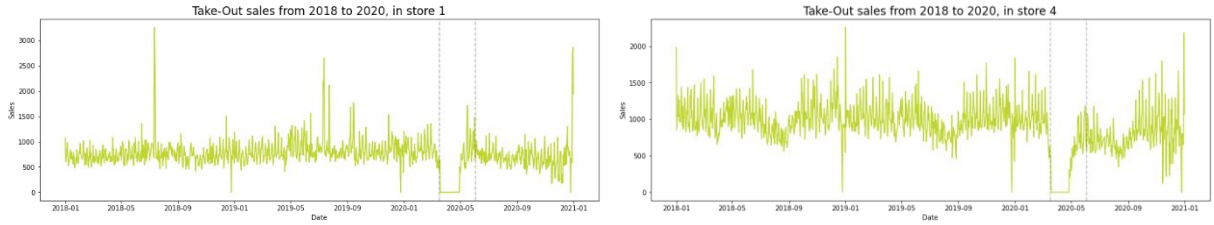


Figure 8.10 – Time series of take-out sales from 2018 to 2020, in storefronts 1 and 4

8.5. TIME SERIES OF TAKE-OUT TRANSACTIONS FROM 2018 TO 2020

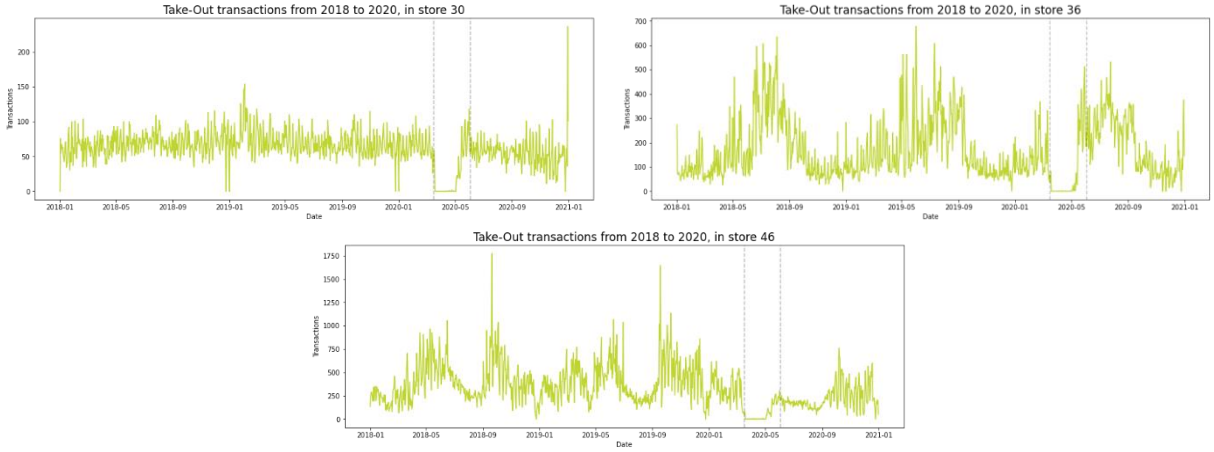


Figure 8.11 – Time series of take-out transactions from 2018 to 2020, in free stands 30, 36, and 46

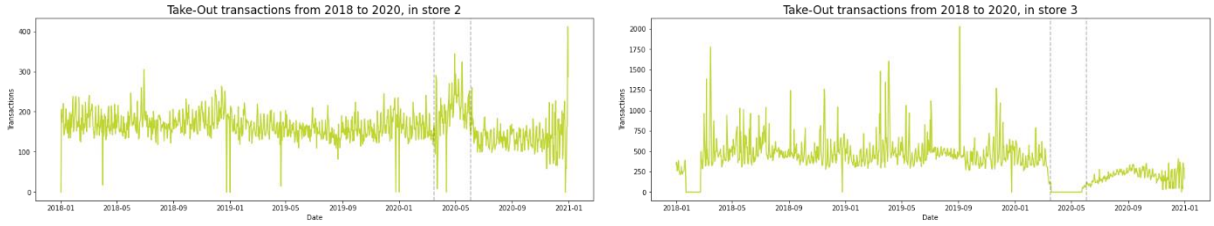


Figure 8.12 – Time series of take-out transactions from 2018 to 2020, in malls 2 and 3

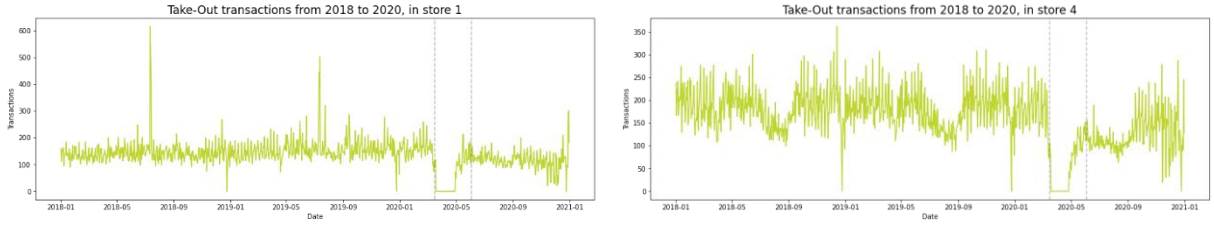


Figure 8.13 – Time series of take-out transactions from 2018 to 2020, in storefronts 1 and 4

8.6. TIME SERIES OF DELIVERY SALES FROM 2018 TO 2020

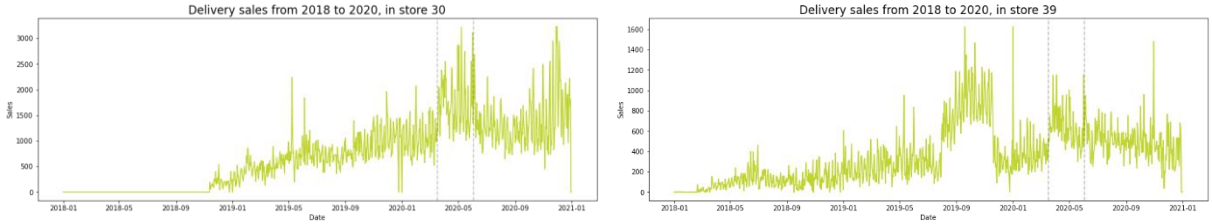


Figure 8.14 – Time series of delivery sales from 2018 to 2020, in free stands 30 and 39

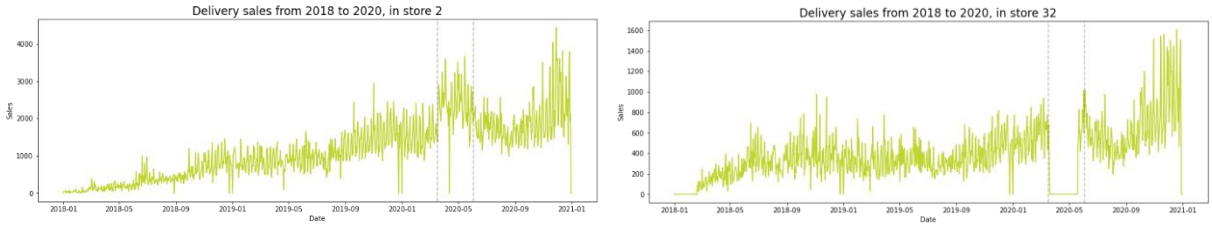


Figure 8.15 – Time series of delivery sales from 2018 to 2020, in malls 2 and 32

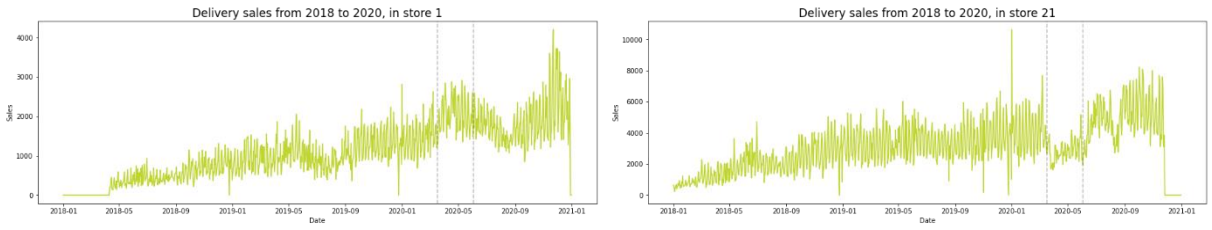


Figure 8.16 – Time series of delivery sales from 2018 to 2020, in storefronts 1 and 21

8.7. TIME SERIES OF DELIVERY TRANSACTIONS FROM 2018 TO 2020

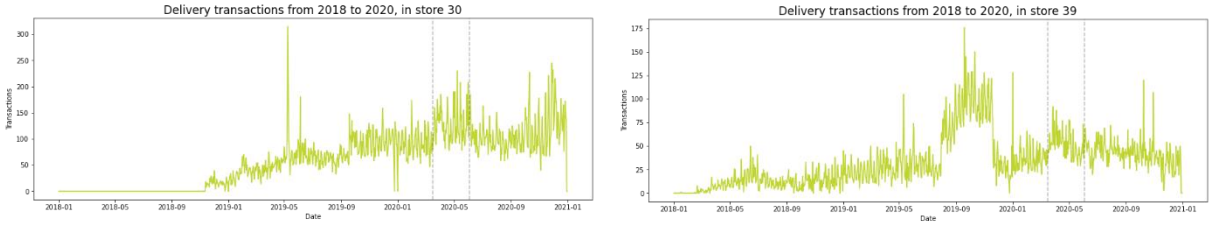


Figure 8.18 – Time series of delivery transactions from 2018 to 2020, in free stands 30 and 39

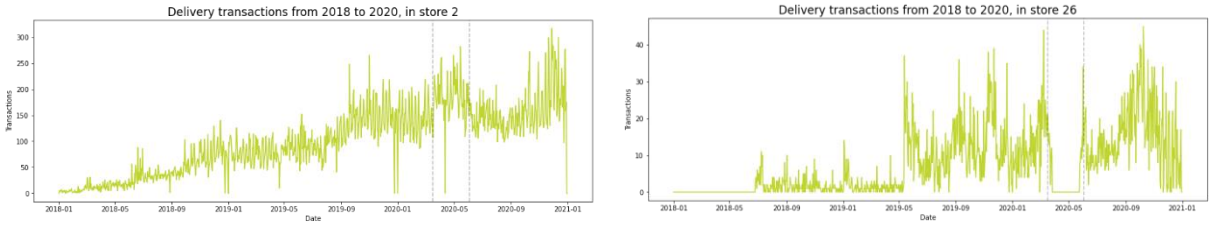


Figure 8.17 – Time series of delivery transactions from 2018 to 2020, in malls 2 and 26

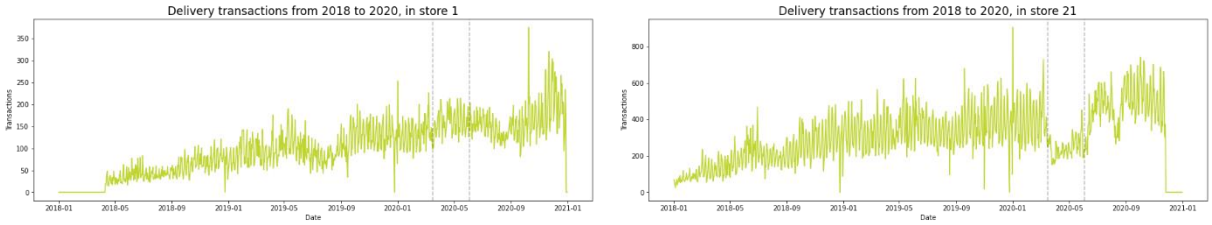


Figure 8.19 – Time series of delivery transactions from 2018 to 2020, in storefronts 1 and 21

8.8. TIME SERIES OF DRIVE SALES FROM 2018 TO 2020

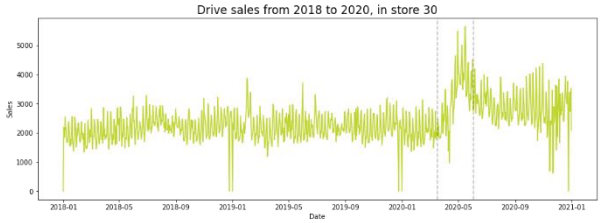


Figure 8.20 – Time series of drive sales from 2018 to 2020, in free stand 30

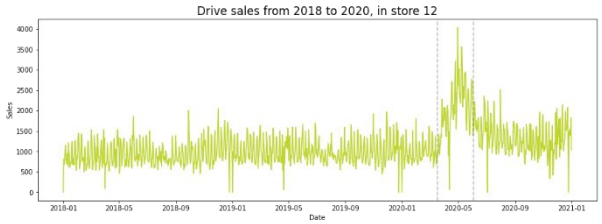


Figure 8.21 – Time series of drive sales from 2018 to 2020, in mall 12

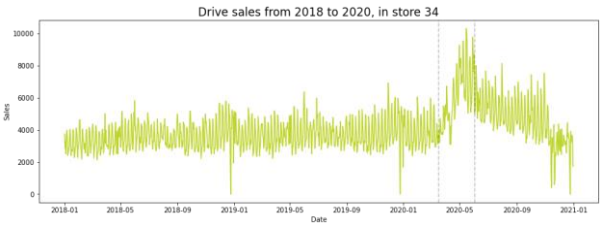


Figure 8.22 – Time series of drive sales from 2018 to 2020, in storefront 34

8.9. TIME SERIES OF DRIVE TRANSACTIONS FROM 2018 TO 2020

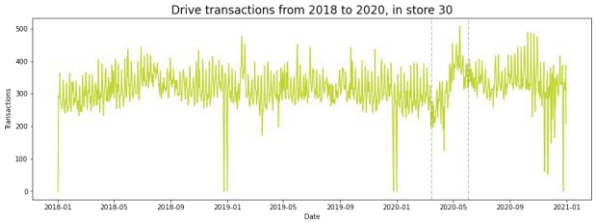


Figure 8.23 – Time series of drive transactions from 2018 to 2020, in free stand 30

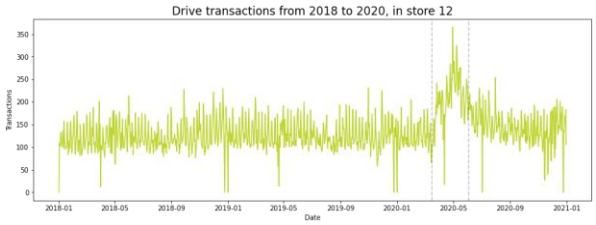


Figure 8.24 – Time series of drive transactions from 2018 to 2020, in mall 12

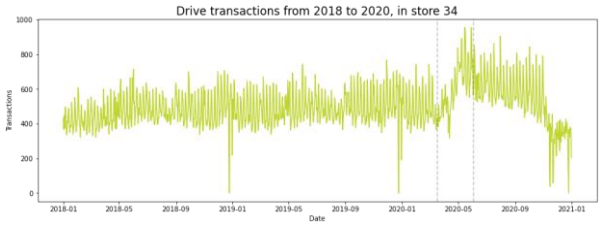


Figure 8.25 – Time series of drive transactions from 2018 to 2020, in storefront 34

8.10. TIME SERIES OF TOTAL SALES FROM 2018 TO 2020

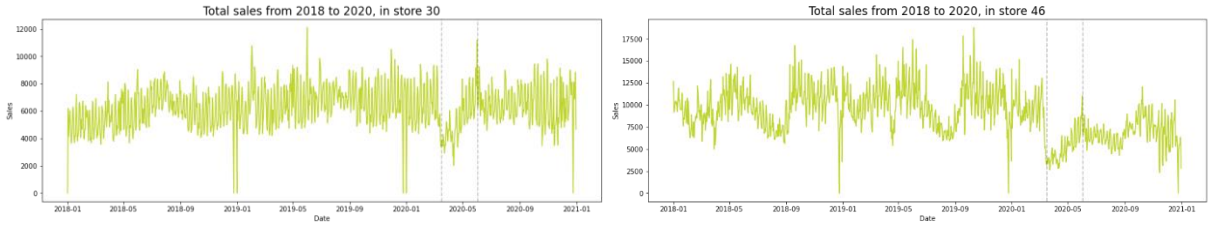


Figure 8.27 – Time series of total sales from 2018 to 2020, in free stands 30 and 46

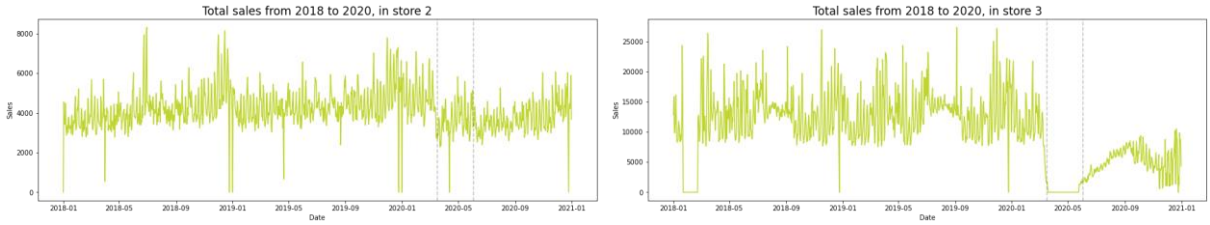


Figure 8.26 – Time series of total sales from 2018 to 2020, in malls 2 and 3

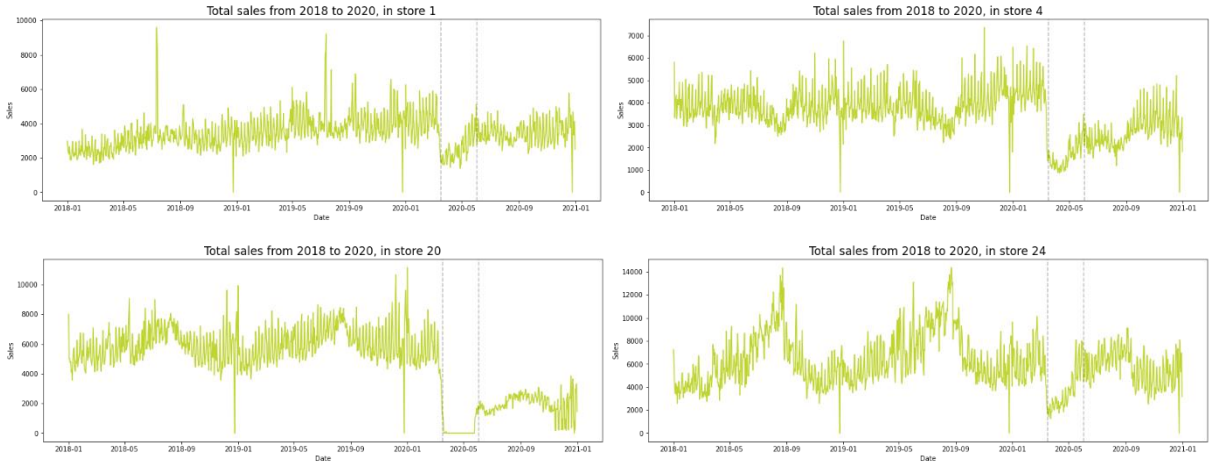


Figure 8.28 – Time series of total sales from 2018 to 2020, in storefronts 1, 4, 20, and 24

8.11. TIME SERIES OF TOTAL TRANSACTIONS FROM 2018 TO 2020

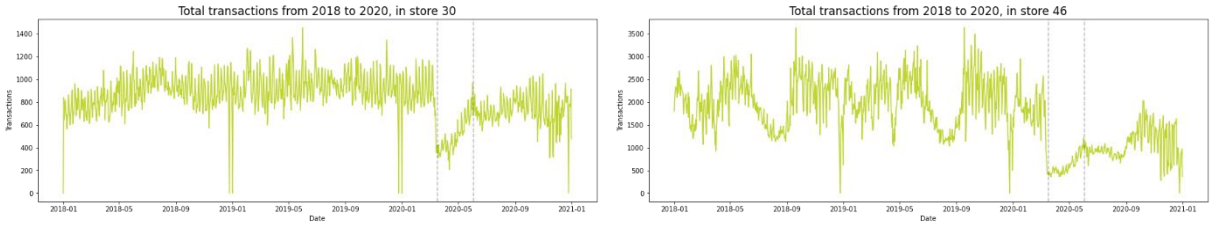


Figure 8.30 – Time series of total transactions from 2018 to 2020, in free stands 30 and 46

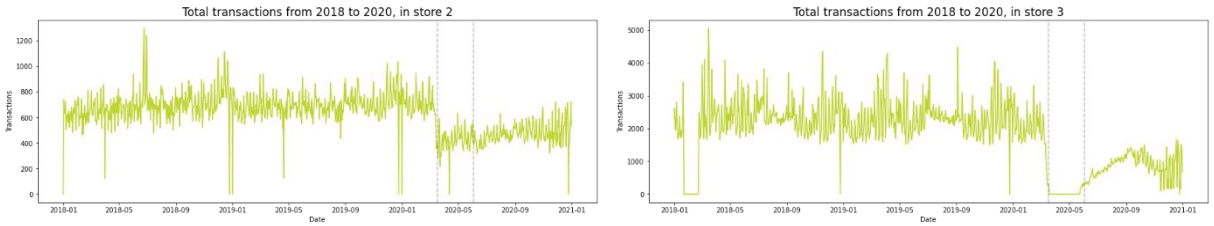


Figure 8.29 – Time series of total transactions from 2018 to 2020, in malls 2 and 3

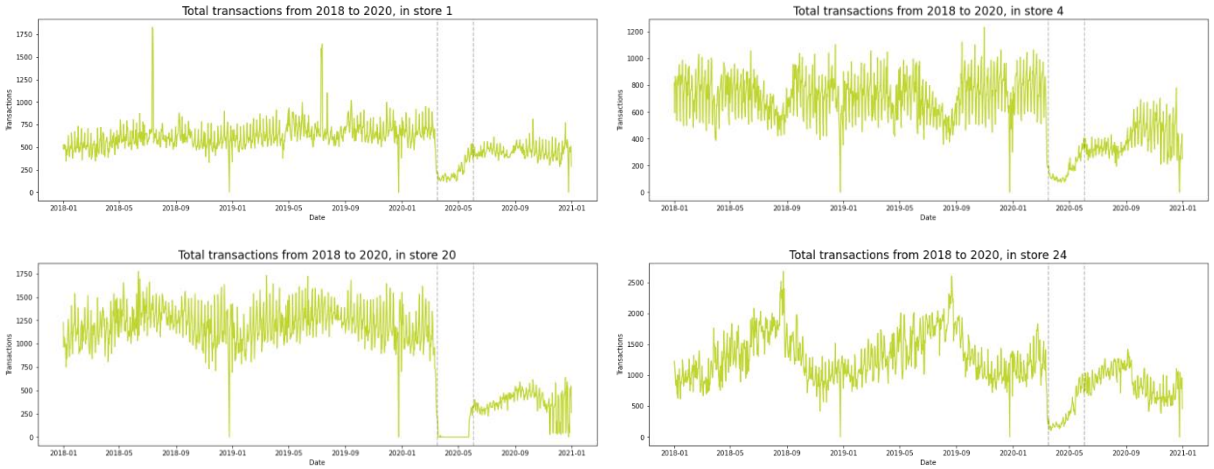


Figure 8.31 – Time series of total transactions from 2018 to 2020, in storefronts 1, 4, 20, 24

8.12. AFTER-POC WEEKLY RESULTS

Sale Type	Sales	Transactions
Total	16.0%	13.0%
Delivery	21.6%	18.4%
Drive	11.5%	8.3%
Eat-In	19.3%	15.9%
Take-Out	27.8%	21.0%

Table 16 – Average MAPE for each sale type, from May 29th to June 5th

Sale Type	Sales	Transactions
Total	16.2%	12.5%
Delivery	23.3%	18.4%
Drive	11.8%	8.5%
Eat-In	22.7%	14.7%
Take-Out	29.7%	22.2%

Table 17 – Average MAPE for each sale type, from June 6th to June 12th

Sale Type	Sales	Transactions
Total	12.9%	10.7%
Delivery	18.4%	16.0%
Drive	12.4%	10.1%
Eat-In	23.0%	19.0%
Take-Out	25.7%	19.8%

Table 18 – Average MAPE for each sale type, from June 13th to June 19th

Sale Type	Sales	Transactions
Total	11.0%	11.9%
Delivery	19.5%	16.4%
Drive	10.7%	11.0%
Eat-In	20.8%	22.3%
Take-Out	23.2%	20.7%

Table 19 – Average MAPE for each sale type, from June 20th to June 26th

Sale Type	Sales	Transactions
Total	11.5%	10.5%
Delivery	18.7%	16.7%
Drive	11.8%	8.5%

Eat-In	28.3%	30.1%
Take-Out	24.8%	21.7%

Table 20 – Average MAPE for each sale type, from June 27th to July 3rd

Sale Type	Sales	Transactions
Total	8.5%	8.2%
Delivery	15.3%	12.7%
Drive	12.3%	10.1%
Eat-In	19.5%	22.4%
Take-Out	22.7%	19.6%

Table 21 – Average MAPE for each sale type, from July 4th to July 9th

Sale Type	Sales	Transactions
Total	8.0%	6.4%
Delivery	14.1%	12.3%
Drive	9.4%	7.9%
Eat-In	14.8%	14.4%
Take-Out	25.1%	20.4%

Table 22 – Average MAPE for each sale type, from July 12th to July 16th

8.13. QLIK SENSE DASHBOARD[2]

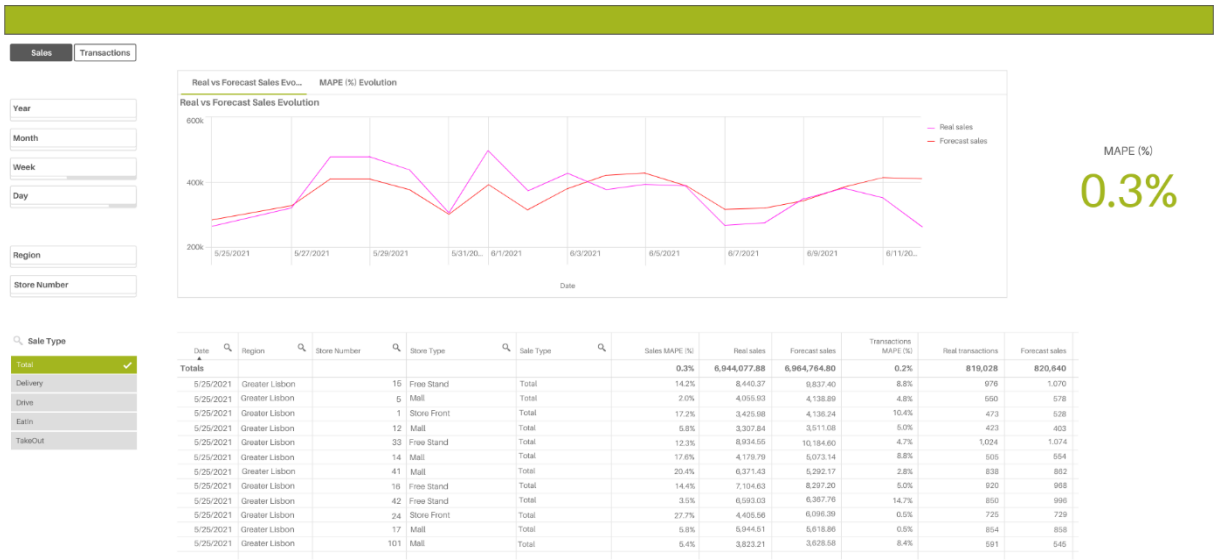


Figure 8.32 – Qlik Sense dashboard developed by Noesis' DAAI (BI) specialists

[2] Anything that could be used to identify the franchise (including the color scheme) has either been altered or completely omitted.