FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Including the effect of retail product characteristics on retail sales forecasting using Deep Learning

João Miguel

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Alexandra Oliveira

Second Supervisor: Luís Paulo Reis

March 9, 2022

Including the effect of retail product characteristics on retail sales forecasting using Deep Learning

João Miguel

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. Rui Maranhão
External Examiner: Prof. Joaquim Gonçalves
Supervisor: Prof. Alexandra Oliveira

March 9, 2022

Abstract

The retail market sells a wide range of products, from food to pharmaceuticals, some of them characterized by their short shelf-life. The high volatility in the demand for these products makes it difficult for the retailers to forecast their sales. A poor prediction of the products that need to be replenished and in what quantity results in the understocking or overstocking of a product. While the understocking of a product results in loss of sales, the overstocking of a product leads to the deterioration of products, waste of inventory, or need to reduce its price. Both errors cause the loss of revenue to the retailer and a weakening of its position in the market to its competitors. An accurate forecast of the future sales of these products is therefore a key aspect in the stock management and logistic operations of the retailers, in order to maximize their profits and establish a competitive advantage in the market.

Current research indicates that the characteristics of products, as well as characteristics of stores where they are sold, can influence the customers demand behavior. The characteristics of a product may include its appearance, size, weight, package size, package type, fresh perception, expiration date, etc. [21]. The characteristics of a store may include its location and the location of the competitors, the region's weather, the shell display of products, etc. [21]. Although forecast models were already developed in order to assess the effect of various external and internal factors on customer demand, there is still a need for models that explore the effects of specific product characteristics [6] more accurately.

Our goal is to implement and analyze methods, mainly based on Deep Learning techniques, that can learn global models from several time-series, thus including the effect of the attributes/characteristics of retail products and stores to help predicting costumers demand for different products. One of these models is DeepAR, which proved to be effective doing this task in problems involving hundreds or even millions of related time-series. These models can also create predictions for products with short sales history [24]. Despite these advantages, they lack some extra research. We also want to implement supervised models based on Decision Trees, such as Random Forest and Gradient Boost, with the necessary adjustments to learn a global model from all time-series, in order to access the importance of the features used as input and thus understand which products and stores' attributes have the greatest impact on product sales values - or if they have an impact at all. These models capable of learning a global model from several time-series are also compared to Prophet models, which follow a more classic and common approach in State-of-Art studies, tuning its hyper-parameters and fitting a new model for each time-series to be predicted.

For this purpose, we used a three-step methodology: the first step aims to search datasets with historical data on products' sales, as well as static variables with information on products and stores where they are sold; the second one targets to develop all models that will be used in the study - the DeepAR and Decision Trees able to use the previously static variables and sales

information to learn a global model -, as well as the Prophet model for comparison; in the third step, the models are used to predict time-series of sales from different levels of aggregation, with different behaviors - stable sales with strong seasonality and trends, or unstable and intermittent sales -, and different magnitudes. To make a good evaluation of the models in the different datasets, we have to choose the appropriate errors and apply a robust and effective strategy to validate our models.

When product sales were stable and with rare intermittency, the MASE and MAPE errors made a good assessment of the performance of the models in almost all time-series of the dataset, as they both don't depend on the magnitudes of sales of different time-series. When product sales became unstable, volatile, and intermittent, the MAPE metric became unreliable as it handles poorly the presence of null values in the time-series to be predicted, and the best metric to use was MASE. The validation strategy implemented - Walk-Forward Validation - allied to these metrics allowed for a good evaluation of the models at the different levels of aggregation, showing whether their results remained consistent as new sales values of the time-series become known and it becomes necessary to predict new horizons of sales.

All models were able to obtain accurate predictions for time-series where sales are stable and with strong seasonality or trend, easily capturing their behavior. In this case, Random Forest obtained the best results within the global models, only giving importance to lag features and not attributing any importance - or residual values of importance - to the remaining features. This shows that, for these conditions, the previous sales values of the time-series are enough for the models to make good predictions. When time-series sales become more unstable, volatile, and intermittent, Random Forest models begin to increase the importance of other features, in addition to lag values, including static variables with information on products and stores. This shows that, under these conditions, the models start needing the help of new features and attributes to make good predictions, in addition to the previous sales values of the time-series. Under these conditions, the models presented greater difficulty in capturing the sales behavior of the different time-series, and most models tended to obtain better results in time-series with higher sales magnitudes. In almost all conditions, Prophet models obtained more accurate results in their predictions, with the trade-off of needing much more time to tune their hyper-parameter and fit a new model for each predicted time series. Within the global models, the deeper models took about 8,5 minutes and 10 minutes, using the Negative Binomial and Poisson distributions, to learn to predict 70 time-series, while the Gradient Boost and Random Forest models took 14 and 22 minutes, respectively. On the other hand, predicting the same time series using Prophet models required more than 13,5 hours.

DeepAR models were able to learn to predict a much larger number of time-series in less time. As such, these models will ideally continue to be studied in the future, as they respond to an urgent need for retailers: learn a model capable of predicting several time series in a timely manner, given that the quantities of SKUs sold at different stores tend to grow with modern times and with the expansion of large retailers. An interesting idea would be to use powerful computational resources to implement parallelism with Prophet models so that different models are simultaneously fitted to different time series of sales. This would reduce the time needed to predict the sales of several products with this model, which is ideal for capturing the sales behavior of individual time series.

Keywords: Retail Forecasting, Deep Learning, Product Characteristics

ACM Areas:

CCS → Applied computing → Operations research → Forecasting,
CCS → Applied computing → Operations research → Consumer products,
CCS → Computing methodologies → Machine learning → Machine learning algorithms

Acknowledgements

First, I want to give special thanks to Supervisor Alexandra Oliveira for her dedication, help, advice and patience over the last year.

My thanks are extended to Professor Luís Paulo Reis, who also made possible the development of this thesis.

This work has been partially supported by the project “IBPS - Intelligent Batch Processing System” , with the reference POCI-01-0247-FEDER-069998, co-financed by the European Regional Development Fund (ERDF), through the Operational Programme for Competitiveness and Internationalization (COMPETE 2020), under the PORTUGAL 2020 Partnership Agreement.

João Miguel

"Until I began to learn to draw,
I was never much interested in looking at art."

Richard P. Feynman

Contents

List of Figures

List of Tables

Abbreviations

ARIMA	Auto-regressive Integrated Moving Average
SARIMA	Seasonal Auto-regressive Integrated Moving Average
SARIMAX	Seasonal Auto-regressive Integrated Moving Average with external variables
NN	Neural Networks
ANN	Artificial Neural Networks
RBF	Radial Basis Function
WNN	Wavelets Neural Networks
MLPNN	Multiple-Layer Perceptron Neural Network
SVR	Support Vector Regression
SVM	Support Vector Machines
DNN	Deep Neural Networks
RNN	Recurrent Neural Networks
LSTM	Long Short-Term Memory
ME	Mean Error
MAE	Mean Absolute Error
MdAE	Median Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
MPE	Mean Percentage Error
MAPE	Mean Absolute Percentage Error
MdAPE	Median Absolute Percentage Error
RMSPE	Root Mean Square Percentage Error
RMdSPE	Root Median Square Percentage Error
MRAE	Mean Relative Absolute Error
MdRAE	Median Relative Absolute Error
GMRAE	Geometric Mean Relative Absolute Error
MASE	Mean Absolute Scaled Error
MdASE	Median Absolute Scaled Error
RMSSE	Root Mean Squared Scaled Error
SKU	Stock Keeping Unit

Chapter 1

Introduction

1.1 Context

The retail market is highly competitive, consisting of numerous physical stores or online sales platforms that offer a huge variety of choices for the customer to buy their products. The retailers should decide which products they want to sell, in order to make good planning of their logistics operations, ensuring that their customers have available the products they need, in a short time and at the lowest possible cost for the company [6, 12].

Retail market sells a wide range of products, some of them characterized by their short shelf-life. The retailers should decide which products they want to sell, in order to make good planning of their logistics operations, ensuring that their customers have available the products they need, in a short time and at the lowest possible cost for the company. This often results in poor stock management, with situations of understocking or overstocking of products. Product understocking can lead to loss of sales, among other negative consequences. On the other hand, overstocking can lead to stock loss and degradation, as well as misuse of shelf space. In general, both lead to a loss of profit for the retailers, a bad impression for the customers, or even loss of customers for the competition, with a consequent weakening of their position in the retail market. For this reason, it is crucial that retailers have a sales forecasting mechanism that accurately previews the products' sells in order to make a good stock management [6].

1.2 Motivation

A lot of research was done in order to develop demand forecasting models that help retailers to predict their sales with high accuracy. Many of these studies focused on including the effect of demand influencing factors to help predicting the sales of retail products.These effects are accounted by including the factors as external variables in the demand forecasting models, improving the accuracy of the sales forecasts. There are already studies including the effect of factors such as

promotions, product prices, discounts, weather, store location, seasonality, holidays, events, substitution and cannibalization, etc [6, 16].

There is lack of studies that exploit the effects of product attributes/characteristics when forecasting the sales of a retailer. These attributes of a product may be its freshness/appearance, color, package size, quality, shelf-life, etc. As customers pay attention to these attributes during their purchases, the inclusion of these characteristics as external variables could increase the accuracy of forecasts.

Many characteristics of products and stores where they are sold are represented by static variables, that is, variables whose value does not change over time. As such, it becomes necessary to develop methods able to learn a global model from several time-series of different products and stores, thus including and exploiting their features/attributes. However, many of the methods studied in the state-of-art follow a classic approach where the parameters of a new model need to be fitted for each time-series individually, not allowing these variables to be included [24]. Furthermore, many Machine Learning models used in these studies require a lot of manual work to be adjusted to the task of learning a global model from many time-series, resulting in complex pipelines and presenting difficulties to scale for a large number of time-series [24]. To solve this problem, Deep Learning techniques have emerged, such as DeepAR models, which allow learning a global model from several time-series including static variables [24]. However, research on these models for forecasting purposes is still recent and precarious.

DeepAR are innovative deep learning models that use recurring neural networks in historical data. These methods proved to be very effective in generating predictions on problems that involve hundreds of related time-series and can be used in time-series of similar new products introduced on the market and with little sales history. Although descriptions of these DeepAR advantages already exist, the evidence is still limited and needs to be investigated.

As referred, Machine Learning models that are not Deep Learning, require extra manual work and expertise to be adapted to the task of learning a global model from different time series and are difficult to scale for a large number of series. However, in this study, we also aimed to develop supervised models based on Decision Trees, such as Random Forest and Gradient Boost, for their interpretability. By interpretability, we refer to the possibility of accessing the importance of the features used by these models as input to make decisions regarding the values of the target variable, and that made these models popular among the participants of Kaggle Competitions. Thus, these models make it possible to analyze the importance of the products and stores' attributes used as input and to what extent they are used together with past values of the time-series to help predict the future sales of the retail products.

1.3 Objectives and Hypothesis

Firstly, our objective was to select relevant datasets for our study. These datasets must contain historical data sales of different retail products, possibly in different locations or stores, as well as static variables with information on these products and sell locations/stores. The fulfillment of this task required the search and study of several datasets from websites such as Kaggle and Google Datasets.

After selecting the datasets that best fit our needs, we proceed to their preparation and cleaning. This processing is expected to make it easier for us to do a detailed and careful analysis of the datasets, which is crucial to understand the data, to choose the best horizon to forecast, as well as the best variables to use as input features in the forecasting models - for example, find out which lag values of time-series have the greatest predictive value and which static variables on products and stores seem to have the most impact on sales. This processing also aims to make it easier and more efficient the task of Feature Engineering, as well as the possible need to restructure the datasets in new formats that are accepted by the different models used in the study.

Once the datasets are ready, we want to develop models capable of including the attributes of retail products and stores where they are sold to help predict customers' demand. For this, we have to develop models capable of learning a global model from several time-series - Deep Learning techniques, such as DeepAR models, and Machine Learning models based on Decision Trees, such as Random Forest and Gradient Boost, with the appropriate adjustments. By doing this, we want to fill the lack of research concerning DeepAR models and the lack of studies that include products and stores' attributes on forecasting models that help predicting customer demand.

To make a reliable evaluation of our models, we also aim to make a careful and informed selection of the evaluation metrics that best fit the datasets on which the models were applied. Some metrics for evaluating forecasting models are useful when evaluating forecasts on aggregated and stable sales while becoming unreliable assessing the results of models that forecast more volatile, unexpected, and intermittent sales. We also intend to make an effective validation of our models using these metrics. Such validation must keep the natural order of the historical data and evaluate the performance of the models in different horizons of the time-series over time, during which new data are available to include in the models. We must be able to access whether the models remain consistent and accurate in predicting new horizons over time by including more recently available historical data.

Finally, we intend to compare the performance of the different methods predicting the time-series of the selected datasets, using the selected metrics. We also want to verify if the global models can outperform a statistical and classical model, such as Prophet, by learning from all time-series and including the effect of the products and stores' attributes. Prophet models were created by Face-

book and have gained popularity for their ability to adjust to the behavior of individual time-series. These Prophet models use a classic and common State-of-Art approach where the parameters of a new model are fitted to each time-series to be predicted, being a good baseline for the global models to outperform.

1.4 Thesis Contributions

This thesis contributes to the advance in the analysis and implementation of more complex models, such as Deep Learning techniques, capable of fastly learning a global model. These models include static variables, such as characteristics of products and stores where they are sold, in order to predict the demand of several products in a constantly growing market as it is the retail.

We develop and describe an efficient pre-processing pipeline for the Preparation and Cleaning of datasets, which facilitates the subsequent analysis and manipulation of the data - both to perform Feature Engineering and to restructure the data in new formats required by the different forecasting models.

We also add research on the adjustment and application of supervised models based on Decision Trees in the task of demand forecasting, in order to obtain insights into the impact of input features, such as products and stores' attributes, on the target variable - the demand of those products. We refer to strategies used for these models to learn a global model from several time-series - of different products in different stores - and to be able to perform multi-step forecasting.

The developed models are studied in different situations. First, they are applied to different levels of aggregation of the M5 dataset, according to the company's organizational hierarchy of the time-series described in Sub-subsection 3.1.2.1. In a first level, the sales time-series start to be stable and with high magnitudes, becoming successively more volatile and intermittent as they are separated by products and location. Then, the results of the models are analyzed and compared for different intervals of magnitudes of the time-series in each level of aggregation. In this way, it was possible to describe the behavior of the different models and select the best one in each situation - forecasting more stable sales with strong seasonality, as well as time-series of volatile, unstable, and intermittent sales. We also analyze the behavior of models in forecasting time-series of different magnitudes.

We also carry out a study and subsequent election of the best metrics to calculate the error of different forecasting models' predictions on the sales time-series of different datasets, depending on the behavior of such time-series. This also allows selecting the best metrics to evaluate the results of the models in the different situations described above.

We use and describe an effective and reliable method for validating forecasting models. The

implemented validation allows simulating what happens in a real situation of retailers: as time passes, the values of sales previously predicted by the model become available and can be used by the model to help predict the next unknown horizon of sales. Our validation uses the model to predict all time-series of a dataset in N different iterations, being N set to 4 in this study. At each iteration, the model predicts a new horizon for all time-series, including the most recently known values of their previous horizons. Thus, our validation obtains the results of a model when forecasting N different successive horizons, which allows us to verify if the model remains consistent predicting the sales of a retailer over time and assuring its performance is not limited to the luck or misfortune of a good or bad performance in a single time interval.

The Deep Learning methods and Decision Trees - that learn a global model from all time-series - were compared to the Prophet model, known for its ability to adjust to the behaviors of individual time series. The Prophet models need to be trained and fitted for each time-series, being a good representation of the more common approaches of the State-of-Art. Thus, it was possible to observe which models are able to learn to predict a greater amount of time-series with accuracy in the shortest possible time - which is an essential task for most retailers.
In summary, this thesis fills the lack of research concerning DeepAR models and the impact of product and store attributes on customer demand.

1.5 Document Structure

In order to make a proposal for a solution to the problem described in the context, we will review the state of the art in Chapter 2. In Section 2.1, we conducted a survey of models already developed for the purpose of demand forecasting of retail products, systematizing their advantages and disadvantages. In Subsection 2.1.4 we focus on the study of the advantages and innovations of DeepAR models. In Section 2.2 we grouped some external variables that were already included in some previous research of models for the purpose of forecasting sales of retail products. To close the State-of-Art, in Section 2.3 we researched different evaluation metrics used in the evaluation of forecasting models, as well as their advantages and disadvantages, in order to choose the one that best suits our data and models.

In Chapter 3 we describe the Methodology followed throughout the development of the thesis to solve the proposed problem. In Section 3.1 we explain the established requirements for selecting the datasets to be used to test the models developed in the study. Furthermore, in this section we describe the selected datasets. In Section 3.2 we describe the preparation and analysis performed on the selected datasets to clean the data and help understanding its information. In Section 3.3 we describe the feature engineering steps performed to create useful features for training the developed models. In Section 3.4 we describe how the models analyzed in the study were implemented and the *Python* libraries used. For the Global Models, we developed DeepAR models from the GluonTS library using two types of distribution - Negative Binomial and Poisson

- as well as two models based on Decision Trees - Random Forest and Gradient Boost - using the *scikit-learn* and *xgboost* libraries, respectively. For comparison, we also implemented a classic model, more specifically the Prophet model, using its library developed by Facebook. In section 3.5 we describe the evaluation and validation applied to the forecasting models.

In Chapter 4 we describe and analyze the results obtained by the different models developed on different levels of aggregation of the M5 dataset. In this analysis, we predominantly consider the MASE metric, as well as the MAPE metric whenever possible. The global and classic models were compared with each other and tested at different levels of aggregation of sales - described in Sub-subsection 3.1.2.1, where the description of the M5 dataset is made. Also, we analyze the performance of the models in time-series of different magnitudes.

In Chapter 5 we present our conclusions drawn during the phases of the study of the theme, and development and evaluation of the models, as well as obtained from the tests and results of their application in the datasets.

Chapter 2

State of the Art

2.1 Demand Forecasting Methods

2.1.1 Classical Statistical Methods

The classical statistical forecasting models use previous sales history to forecast demand for retail products into the future. Many of these models have emerged from the concept of exponential smoothing. According to exponential smoothing models, the forecasted values of a variable are weighted combinations of their past observations. The older the observation, the lower its weight in the combination. The decrease in the weight of a past observation is exponential to its age.

Exponential smoothing models, including Holt-Winters, and the ARIMA models have been applied in several studies focused on forecasting the demand for products in the retail area. These models are recognized for their ability to model trend and seasonal fluctuations present in aggregated retail sales [22] and for their excellent ability to adapt to forecasting problems with linear behavior [19].

Veiga et al. (2014) compared the accuracy of Holt-Winters and ARIMA predicting the sales of groups of dairy products. The study uses historical data of the monthly demand for a group of dairy products, from 2005 to 2013, provided by a large retailer in southern Brazil. Mean absolute percentage error (MAPE) and Theil's inequality index (U-Theil) were the metrics used to evaluate the performance of both models. According to the values of these metrics, the Holt-Winters model performed better than ARIMA in this specific context. However, both models were highly accurate in their results [11].

Holt-Winters and ARIMA models perform well when economic conditions are stable [5]. Under these conditions, scientific studies have shown that the ARIMA methodology is competitive in terms of accuracy and therefore widely used. The Holt-Winters model was also able to achieve the accuracy of more complex models in pratical applications [11].

In addition, these models are simple and easy to understand. The implementation is also easy, fast and affoardable, since it requires few resources, making them a preferable option to more complex models [17]. However, statistical models are not able to properly model non-linear behaviors of the sales. Also, these models cannot include demand influencing factors as external variables, such as prices and promotions [16].

In fact, Veiga et al. (2016) made a study on the application of non-linear forecasting models based on natural computing approaches and compared their performance to the classical linear methods, ARIMA and Holt-Winters, already studied in its previous article. To this purpose, monthly retail sales of three groups of liquid dairy products, from 2005 to 2013, of a Brazillian multinational company were used. The products were grouped according to their composition and production lines: Group A (50 SKUs of yogurt), Group B (4 SKUs of fermented milk), and Group C (9 SKUs of milk dessert). All groups presented trend and seasonality. MAPE and U-Theil were, once more, the evaluation metrics used to analyze the performance of the models. Both demand forecasting models based on a natural computation approach obtained more accurate results than linear models. Wavelets Neural Networks (WNN) had the most accurate results, due to their excellent approximation properties, suitable for general non-linear signals, and a more efficient capacity to learn than other conventional NN. The study also confirms the influence of the forecasting accuracy on customer satisfaction and in the economic performance of the retail business operations. WNN is computationally difficult, requiring both computation time and a large amount of in-house expertise to determine the initial parameters. Otherwise, the results will not be following the expectations and capabilities of the model. However, the financial compensation caused by the improvement in the accuracy of forecasts makes it important for companies to consider the investment in these models instead of the traditional ones [12].

In a nutshell, these models are tempting for their simplicity and practicality, as well as their ability to effectively model sales with linear behavior when economic conditions are stable. However, these models make many assumptions from the data that may not hold. Thus, when sales are irregular, due to the effect of demand factors, these models are unable to adjust to sales behavior, showing poor results. For these reasons, these models are more often used to forecast sales of aggregated products rather than individual products.

2.1.2 Machine Learning Methods

As previously stated, methods such as Winters exponential smoothing, multiple regression and Box-Jenkins ARIMA model have been broadly used in the analysis of data with trend and seasonal patterns. However, ANN showed up as a promising alternative to investigate seasonal patterns changes over time when classical statistical methods aren't good enough. In fact, Ilan Alon et. al (2001) established that through different forecasting periods and horizons the ANN performed the best, when compared with Winters exponential smoothing and Box-Jenkins models.

On the other hand, these classical models showed up a good performance when the macroeconomic circumstances were relatively stable [5].

Machine learning models can learn approximations to nonlinear functions directly from the data. Thus, these models have the potential to make more accurate predictions, especially when sales are highly volatile and behave erratically. However, this greater ability to adjust to sales behavior also increases the risk of over-fitting [16].

Machine learning models are also more flexible than statistical models, as they allow the inclusion of input variables external to the time series to forecast. While statistical models can only use past observations from a time series to predict future values of the same, machine learning models can use other input variables with predictive value and learn their effect on sales of a time series [14].

Tsoumakas (2019) reviewed existing machine learning approaches for forecasting food sales and the appropriate measures to evaluate their accuracy. This review states the choice of features to be used as input as a crucial, if not the most important, aspect to determine the success or failure of these models. For this reason, it is important to study the different features (input variables) used in previous projects with a focus on developing models for forecasting retail sales [29].

Pillo (2013) focused on the application of learning machines for sales forecasting under promotions. The machine learning models included Artificial Neural Networks (ANN), more precisely multilayer ANN and radial basis functions ANN (RBF ANN), as well as Support Vector Machines (SVM). The study also compares these models to statistical models, including ARIMA, exponential smoothing, and Holt-Winters. The study uses 2 input-output time series of the daily sales receipts of a particular kind of pasta during a 3-years period (2007 to 2009). Both time series were provided by two large retail stores, identified as stores #1 and #2. Store #1 is characterized by poor storage management, which results in several stock-outs and, consequently, irregular sales data. Store #2 is known for good storage management, with stock-outs occurring very rarely. The output of the time series corresponds to the target value, in this case, the daily sales of a specific type of popular brand pasta. The time-series input is a vector with the values of the attributes, also known as predictors, considered to help predict the target variable. The machine learning models were applied using different sets of input attributes [14]:

- 4 inputs: promotion, number of opening hours, price of the product and forecasted number of daily receipts.
- 12 inputs: promotion, number of opening hours, price of the product and the nine calendar attributes.
- 13 inputs: all attributes listed above.

Multilayer ANN models that use 4, 12 and 13 input attributes are referred to as Mul 4i, Mul 12i, and Mul 13i, respectively. Radial Basis Functions ANN models that use 4, 12 and 13 input

attributes are referred to as RBF 4i, RBF 12i, and RBF 13i, respectively. In the same way, Support Vector Machines that use 4, 12, and 13 input attributes are referred to as SVM 4i, SVM 12i, and SVM 13i, respectively. The Mean Absolute Percentage Error (MAPE) was the metric used to evaluate the accuracy of the models [14].

The input-output samples from 2007 and 2008 were used for training and validation. The input-ouput samples from 2009 were used for testing the models. Predictions are made using a sliding window method on the test dataset. The year 2009 of both stores was divided into 10 equal-sized intervals: 10 intervals of 36 days for store #1 and 10 intervals of 32 days for store #2. The predictions were made for all intervals using a sliding window method [14].

In-store #1, where sales data are irregular due to frequent stock-outs, machine learning models outperformed statistical models in all cases. According to the results of the 20 tests carried out in stores #1 and #2, the SVM and RBF models performed better 7 times on 20 (each corresponding to 35% of the total), ANN performed better 5 times on 20 (25% of the total), while the statistical models only performed better once (5% of the total). In each store, the machine learning model with the best performance was compared to the statistical model with the best performance. The machine learning model outperformed the statistical model mainly in promotion periods. In both stores #1 and #2, the machine learning models that used 4 attributes performed better than the same models when they used 12 and 13 input attributes to forecast the daily sales [14].

In Pillo (2013), the results achieved by SVM seemed more promising compared to other machine learning models. For this reason, Pillo (2016) made a study focused on the application of Support Vector Machines for sales forecasting under promotions. The SVM methods are, once more, compared to the statistical models ARIMA, exponential smoothing, and Holt-Winters. This study also uses 2 input-output time series from the daily sales receipts of 2 retail stores, but over a period of 5 years (2007-2011). The methodology is very similar to the methodology of the previous study. The study applied SVM with two sets of input attributes: the sets of 4 and 12 inputs already described above, resulting in the SVM 4i and SVM 12i models, respectively. This time, Mean Square Error (MSE) was the metric used to evaluate both statistical and machine learning models. The input-output samples from 2007 to 2010 were used to train and validate the models. The samples of 2011 were used to test the fitted models. The year 2011 of both stores was divided into 13 equal-sized intervals of 28 days. The predictions were made for all intervals using a sliding window method [13].

The smallest mean value of MSE was achieved by the SVM 4i model in both forecasts of store #1 and store #2. Among the 3 statistical models, ARIMA had the lowest average MSE value in both stores 1 and 2. The SVM 4i model performed better than the ARIMA model during the promotion periods (5 in total). This difference was especially noticeable in periods where the effect of promotions has completely deregulated the weekly sales trend. In these periods,

statistical models continue to follow the weekly sales trend, while SVM are able to catch part of the variation caused by the promotions. According to the mean value of MSE, SVM 12i performed better than the statistical models when forecasting the sales of store #2. However, the mean MSE error achieved by the ARIMA model when forecasting the sales of store #1 is lower than the value achieved by the SVM 12i model. In short, the SVM performed better than the statistical methods in the case of a suitable selection of the input attributes (4i) [13].

Both studies by Pillo (2013 and 2016) show that machine learning models can be more advantageous than statistical models, when applied to contexts where sales are highly volatile as a result of demand influencing factors such as promotions. However, they also highlight the importance of an appropriate selection of the input attributes to take full advantage of such models [14, 13].

Nonlinear models are difficult to scale for large amounts of time series. The limitations of current computing power make it impractical to apply them to tens of thousands of SKUs in hundreds of stores. For this reason, many studies of nonlinear models are applied to small quantities of products, usually in the order of tens [16].

In short, these machine learning models can adjust to non-linear behaviors, having the potential to make more accurate predictions than classical models when the sales have irregular behavior. Furthermore, these models allow the inclusion of external variables to express the effect of demand influencing factors. In contrast, the great ability of these models to adjust to sales behavior also increases the risk of over-fitting. Furthermore, it is required more effort in the selection and preparation of appropriate input variables to take full advantage of these models. Their implementation is more complex, time-consuming, and with a laborious preparation of the model parameters, which may even require some expertise.

2.1.3 Complexity Cost versus Model Accuracy

Complex models with good accuracy in their predictions make a positive contribution to the profitability of retailers. On the other hand, the higher complexity of models imply higher costs in their implementation and data preparation, as well as the need for expertise to set up and maintain them. Therefore, the retailer should only consider additional costs in the data preparation and the implementation of more complex models if this results in a significant increase in the accuracy of the forecasting model [4].

Gur Ali (2009) explored the cost of a higher complexity model in terms of setup, data preparation and maintenance, and its accuracy tradeoff when applied to the problem of forecasting grocery sales. For this purpose, models of increasing complexity were used: from exponential smoothing as the benchmark technique, through linear regression models, and ending in more sophisticated machine learning models, such as Support Vector Regression (SVR) and Regression Trees. Each model, with the exception of the benchmark, was combined with different sets of input features

	Benchmark	SVR Poly1	SVR RBF	SVR Poly1 + Sm	SVR RBF + Sm	RT Feat
Promotions	22.19	17.50 (−21.13%)	15.43 (−30.48%)	16.94 (−23.66%)	14.98 (−32.50%)	**7.73** (−65.17%)
No promotions	**2.60**	3.05 (+17.46%)	3.04 (+17.23%)	**2.58** (−0.64%)	2.72 (+4.83%)	2.91 (+12.12%)

Figure 2.1: MAE accuracy obtained by different models of [4] in periods of promotions and in periods with no promotions

with increasing complexity and cost in their preparation. The mean absolute error (MAE) was used to evaluate the performance of the candidate models. According to the results shown in the table of Figure 2.1, none of the complex techniques based on machine learning and data mining was able to improve the forecasting accuracy of the benchmark in weeks with no promotion. On the other hand, the benchmark has poor results predicting in weeks with promotions. In these cases, the remaining models surpass the benchmark accuracy in increases ranging from 21.13% to 65.17%, with the best accuracy achieved by the Regression Trees with features [4].

The study concludes Exponential Smoothing is performing well if sales are relatively stable and without promotions of any kind. However, in promotion periods, more complex models that include input variables substantially improve the forecast results. It was also found that the use of more detailed input features is only beneficial if more advanced techniques are used: The use of features in the linear regression model didn't add any benefits to it. The application of appropriate features in complex machine learning models led to significant improvements in its results [10].

In short, when sales are highly irregular due to the effect of demand influencing factors or volatile economic conditions, the application of simple traditional models to forecast the demand may provide unsatisfactory results. In these cases, it becomes important to consider more complex nonlinear models [10].

2.1.4 DeepAR: Probabilistic forecasting with auto-regressive recurrent networks

Most of the forecasting models currently used in the retail area have been developed to forecast individual time series or very small groups of time series. Many of these models belong to the class of models described in 2.1.1, which are based on the classic Box-Jenkins methodology, on the exponential smoothing techniques, or the state-space model [24]. The neural networks studied by the forecasting community have also been usually applied in the prediction of individual time series. As such, the parameters of these models need to be estimated independently for each time series from its past observations, which corresponds to fit a new model for each time series [24]. These models also require extra effort in their selection, manually in most cases, to include different factors such as the autocorrelation structure, trend, and seasonality of the time series [24].

In recent years the need has arisen to forecast large quantities of related time series, in the order

of thousands or even millions, as well as to learn the relationships that they maintain among themselves: an example would be to forecast the sales of the different Stock Keeping Units (SKUs) of a retailer [24].The objective would be to use the historical data of all related time series to learn how to predict the individual time series, which would allow to fit more complex and accurate models without the risk of over-fitting [24]. This could also reduce the time and labor-intensive steps required by classic methods in the selection and preparation of covariates, as well as in the selection of the models [24].

Despite the advantages of sharing information between related time series, the heterogeneous nature of real-world data makes this a difficult task. There are already some approaches to achieve this. Some examples are the use of clustering techniques, such as k-means to calculate seasonal indices [9], and the use of unsupervised learning techniques as pre-processing steps, such as handling promotional effects via pooled principal component analysis regression [28]. However, these approaches usually involve breaking down the forecasting problem into distinct sub-problems and then applying a dedicated model or chain of models to each one. Thus, these result in complex pipelines that are hard to tune and to maintain, presenting yet two drawbacks [24]:

- It is not always possible to break down the forecasting problem into a sequence of different procedures.
- Decomposing the problem requires a way to reach model ensembling. This increases the complexity even further.

Deep neural networks appear as a great alternative to these pipelines. These models only need some standard pre-processing of the data, and then they can learn an end-to-end model to solve the forecasting problem as accurately as possible. The pre-processing of the data itself is also optimized during the learning process of these models. These models depend mainly on what they can learn from the data, not depending on heuristics nor the knowledge of experts [24].

Salinas (2020) proposed the development of DeepAR, a forecasting model based on recurrent neural networks (RNN) to produce probabilistic forecasts. This model is capable of learning a global model from the historical data of a very large set of time-series, and then use it to predict individual time-series. RNNs are advantageous for their ability to model the sequential nature of time series, thus having a smaller number of parameters that need to be fitted. In fact, the proposed model uses a long-term memory-based (LSTM) architecture, which alleviates the explosive or vanishing gradient problem, quite common in RNNs that train via a gradient-based optimization procedure [24].

The DeepAR models have some crucial abilities that classic models lack. They can learn seasonal behaviors and dependencies on given covariates across time series while requiring minimal manual intervention in providing covariates in order to capture complex, group-dependent behavior. They

can also provide forecasts for items with little or no history available after learning the behavior of other similar items [24].

Another drawback of the classic models is that they rely on several assumptions about the data, such as gaussianity, stationarity, or homoscedasticity of the time series [24, 11]. However, data from real forecasting problems are often intermittent and irregular, which violates these assumptions. The DeepAR model does not make assumptions on the data and is able to incorporate a wide range of likelihood functions, allowing the user to choose the one that best suits the statistical properties of the data. As such, Salinas (2020) approaches the demand forecasting problem by incorporating a negative binomial likelihood, which is appropriate for intermittent data, and then applying non-linear transformations to the data learned by deep neural networks. The DeepAR model also makes probabilistic forecasts in the form of Monte Carlo samples, commonly used to compute consistent quantile estimates for all sub-ranges of forecasting horizons [24]. These factors together allow the DeepAR models to produce more accurate forecast distributions than the models previously developed with the purpose of forecasting.

Salinas (2020) evaluated the proposed DeepAR model on several extensive real-world data sets. The results showed that this model produces more accurate probabilistic forecasts than the other state-of-art models in a range of different input characteristics, requiring minimal manual work to do so [24]. It is important to emphasize that these models are able to make predictions on intermittent data, whose sales have consecutive 0 values, by integrating a negative binomial likelihood, as well as they are able to deal with time series of highly scattered magnitudes by applying a special treatment [24]. As such, deep learning techniques can be a good alternative to approach demand probabilistic forecasting problems.

2.2 Demand Influencing Factors as External Variables

Customers' buying patterns change continuously, making it more difficult to forecast sales at retail stores. These changes in customers' purchasing behavior can be influenced by internal factors, such as promotion, price reduction, and stock-outs, or external factors, such as weather, and holidays. It is important to include these factors that influence demand as external variables in the forecasting models, in order to improve the accuracy of their predictions [6, 7].

For this reason, Arunraj (2015) studied the development of Seasonal Auto-regressive Integrated Moving Average models with external variables (SARIMAX), to include the effect of demand influencing factors. These models result from the combination of Seasonal Auto-regressive Integrated Moving Average (SARIMA) models with demand influencing factors using linear regression. Two of these hybrid models were developed, one combining SARIMA with the demand influencing factors using multiple linear regression, SARIMA-MLR, and the other combining SARIMA with the same factors using quantile regression, SARIMA-QR. The latter was developed

to show the importance of dealing with uncertainty by predicting intervals instead of forecasting points. The study reviews the different demand influencing factors identified by the food sales forecasting literature. These factors revealed a significant impact on the behavior of demand and, according to the review, can be divided into the following categories [6]:

- **Events:** Regular holidays, festivals, and school vacations [6]. The effect on demand caused by these factors may depend on the location, demographics, and cultural habits of customers. In these festival and holiday seasons, demand in stores close to tourist points may vary as a result of visiting tourists. Likewise, demand in stores close to the borders can vary as a result of cross-border shopping and visits [7].
- **Weather:** Air temperature, precipitation, snow cover, sunshine duration, wind speed, and relative humidity [6]. Extreme weather conditions (rainfall, snowfall, very hot and cold temperatures) can affect the behavior of customers, keeping them at home or forcing them to visit nearby stores [7].
- **Seasonality:** Day of the week, day of the month, the month of the year, yearly seasons, and yearly quarters [6].
- **Price:** Normal price, price reduced by discounts or promotions [6]. Price reductions, planned or not, may encourage customers to buy more, resulting in more volatile demand [7].
- **Substitution and Cannibalization** [6].
- **Product characteristics:** Product freshness/appearance, package size, quality, and shelf-life (life of a product after it arrives in the store). The product freshness/appearance is perceived by the store managers (while discounting or discarding) and consumers. The quality of a product includes its physical condition, such as damage, disease, and mold [6].
- **Number of customers:** The customers visiting a store can be categorized into regular customers, irregular customers, or special visitors such as tourists [6].

These factors can be further classified into internal factors, which are controllable; partially internal factors which, as the name implies, can be partially controlled; and external factors, which are impossible to control. Among the factors listed above, prices and product characteristics are internal factors, substitution and cannibalization are partially internal factors, and the remaining are external factors [6].

The study [6] tried to include the effect of all demand influencing factors. However, the lack of information on products' characteristics/attributes, substitutions, and customer visits did not allow the inclusion of these factors. As such, the study included in the model the following external variables: Days of the week; Month of the year, as dummy variables (0 or 1) from February to December; Holidays, including regular holidays, festivals, upper Austrian holidays, and school

vacations incorporated as dummy variables (0 or 1); Promotions, planned and unplanned, both represented by a variable expressed in the percentage of the price reduction; Discounts; Weather, expressed in the values of maximum air temperature (ºC), precipitation (mm), fresh snow depth (cm), snow depth (cm), sunshine duration (hours), and relative humidity (%).

The proposed SARIMA-MLR and SARIMA-QR models performed better than other models used in the study, including SARIMA, MLPNN, and a seasonal naive method. The comparisons were made using the MAPE and RMSE values of each model. The MLPNN model also included the demand influencing factors and performed better than the SARIMA and Seasonal naive models. The results demonstrate that including the effect of the right demand influencing factors can improve the performance of forecasting models. The SARIMA-QR had better accuracy in the results when compared to all models used in the study, demonstrating the importance of dealing with uncertainty by predicting intervals instead of forecasting points [6].

In 2016, Arunraj carried out another study on the development of the SARIMA model with external variables (SARIMAX). The study again mentions the demand influencing factors listed above, except for the product's characteristics and the number of visitors. It also mentions the forecast accuracy of a time series relies strongly on the following aspects [7]:

- Data availability: It is important to have complete and extensive historical data available to identify external factors that affect sales.
- Data quality: The forecasting model depends on the input data it uses, both in the training and test stages. As such, it becomes important that the quality of this input data is guaranteed. The quality of the data used in the estimation period is crucial to better forecasts (for example, when data of the weather from the forecast period is used).
- Forecast horizon: Growing forecast horizons can increase uncertainty, leading to high inaccuracy in the forecast.

Arunraj (2016) included the effect of holidays, price reductions, and months in the SARIMAX model. According to the values of the coefficient of determination, R^2, the SARIMAX model including the effect of price reductions, holidays, and months was able to explain about 61% of the variations in demand, while the SARIMA model only explained 38%. The MAPE and RMSE values of both models also showed that the inclusion of external variables in the SARIMAX model increased its accuracy compared to the simple SARIMA model, reducing the error of both evaluation metrics [7].

In the review by Arunraj (2015), it is possible to see that there are several studies on forecasting in the food retail area that include the price effect (Arburto and Weber, 2007; Ali et al., 2009; Hasin et al., 2011), as well as the effect of its reduction, either through promotions (Ali et al, 2009;

Žliobaitė et al, 2012; Ramanathan and Muyldermans, 2010; Hasin et al, 2011) or discounts (Ali et al, 2009; Ramanathan and Muyldermans, 2010). The effect of promotions can include variables for the type, size, and duration of the promotion. The discount can be expressed as a percentage or discounted price. The effect of holidays (Aburto and Weber, 2007; Ramanathan and Muyldermans, 2010; Žliobaitė et al, 2012; Sharma and Sharma, 2012; Lee and Hamzah, 2010), festivals (Aburto and Weber, 2007; Ramanathan and Muyldermans, 2010; Žliobaitė et al, 2012), and school vacations (Aburto and Weber, 2007; Žliobaitė et al, 2012) are also included as external variables in previous studies. Variables about the time of sales, such as a week in the year (Ramanathan and Muyldermans, 2010), day of the week (Sharma and Sharma, 2012), and month or season of the year (Mirasgedis et al., 2014), as well as variables on weather conditions (Mirasgedis et al., 2014; Aburto and Weber, 2007; Hasin et al, 2011; Ramanathan and Muyldermans, 2010; Sharma and Sharma, 2012; Žliobaitė et al, 2012; Agnew and Thornes, 1995; Fearne et al., 2006), are also widely included in the models of previous studies. Some other factors are mentioned whose effect has been included in previous state-of-art studies. However, there seems to be a lack of study on the effect of product characteristics/attributes on sales of food retailers and the inclusion of such effect on forecasting models as external variables.

2.3 Forecast Evaluation Metrics

To compare different forecasting models and to know their performance in forecasting the demand of a retailer, it is necessary to evaluate their accuracy when predicting the time-series of the retailer's products. To this end, the time series of the sales of different products are usually divided at the same time point, resulting in two datasets each: the training set (data previous to the time point) and the test set (data after the time point). For each product, the model is trained using the training set, and then the fitted model is used to forecast the sales of the test set. The test set is used to assess the accuracy of the fitted model predictions, through the calculation of an accuracy measure that reflects the errors between the predicted values and the actual values of the test set. The accuracy measure is calculated to each product time-series, and then the results obtained by the model for all products are aggregated [29].

Several accuracy measures were proposed by different studies and authors when it comes to forecasting sales in the retail area. Different accuracy measures may have different properties. As such, the results from comparing different forecasting methods and the conclusions drawn about their performance depend very much on the chosen accuracy measures. Many of the proposed accuracy measures may be misleading and inapplicable for certain contexts of the data, leading to unreliable results [16]. For example, many of these measures are not suitable for predicting intermittent data, which have multiple sales close or equal to 0.

Hyndman (2006) reviewed accuracy measures proposed by different authors and used in the M3-competition, showing their inadequacies and alternative measures to overcome them. Likewise,

this section will summarize the different measures proposed by the authors, explore their limitations, as well as the cases in which their application is not adequate. Before proceeding, let's assume that Y_t is the real demand value of a time-series in period t and that F_t is the attempt to forecast Y_t. Then, the forecasting error at instant t is given by $e_t = Yt - Ft$ [18].

2.3.0.1 Scale-dependent measures

Many commonly used accuracy measures depend on the scale of the data. They are useful for comparing different methods applied to the same dataset [18, 22, 25]. On the other hand, these measures are not recommended for comparing the application of methods on different datasets, which data may have different scales [18, 25]. Also, these measures are very sensitive to outliers [25].

The most used scale-dependent measures are the Mean Error (ME), the Mean Absolute Error (MAE), the Median Absolute Error (MdAE), the Mean Squared Error (MSE), and the Root Mean Squared Error (RMSE) [18, 25, 22]. These measures are based on absolute errors, $|e_t|$, and square errors, e_t^2 [18, 25]:

$$ME = mean(e_t)$$

$$MAE = mean(|e_t|)$$

$$MdAE = median(|e_t|)$$

$$MSE = mean(e_t^2)$$

$$RMSE = \sqrt{median(e_t^2)}$$

The MAE measure is interesting to evaluate the performance of the methods in a dataset, as it is simple and easy to understand [22]. The RMSE and MSE values are popular and widely used [18]. RMSE is worthier than MSE as it is set to the same scale as the dataset [18]. The squared errors of both RMSE and MSE metrics make them more sensitive to larger errors. On the one hand, Ramos et al. (2015) state that this makes the RMSE measure more valuable than other measures that do not give more weight to major errors. On the other hand, this also makes the RMSE and MSE measures more sensitive to outliers when compared to the MAE and MdAE measures, so some researchers advise against their use [18, 25].

The scale of the target variable may vary widely between time series of different products. For this reason, the aggregation of these non-scaled errors from different products is not a viable evaluation [29].

2.3.0.2 Measures based on percentage errors

Measures based on percentage errors have the advantage of being independent of the scale of the data. For this reason, these measures are often used to assess the accuracy of methods in different datasets [18, 25, 22]. The values of these measures for different products can be aggregated [29]. The percentage errors are given by:

$$p_t = 100\frac{e_t}{Y_t}$$

Some of the most commonly used percentage error measures are the Mean Percentage Error (MPE), the Mean Absolute Percentage Error (MAPE), the Median Absolute Percentage Error (MdAPE), the Root Mean Square Percentage Error (RMSPE), and the Root Median Square Percentage Error (RMdSPE) [18, 25, 22, 16]:

$$MPE = mean(p_t)$$

$$MAPE = mean(|p_t|)$$

$$MdAPE = median(|p_t|)$$

$$RMSPE = \sqrt{mean(p_t^2)}$$

$$RMdSPE = \sqrt{median(p_t^2)}$$

These measures are disadvantageous when an observation at any instant t, Y_t, is equal or very close to 0. If an observation is equal to 0 at any time t, Yt = 0, these measures are infinite or undefined. When an observation Y_t is close to 0 at any time, these measures have extreme values [18, 25, 22, 16]. As such, these measures are not suitable for evaluating forecasts of intermittent data, which have many values close to or equal to 0. According to some authors, the measures based on percentage errors are very skewed, but they can be made more stable by applying transformations, such as logarithms [18].

2.3.0.3 Measures based on relative errors

One way to scale the accuracy measures is to divide the forecast errors of the proposed model by the errors obtained by a benchmark model. Being e_t* the error obtained by the benchmark model, the relative error, r_t, is given by $r_t = e_t/e_t*$. This process results in the following Mean Relative Absolute Error (MRAE), Median Relative Absolute Error (MdRAE), and Geometric Mean Relative Absolute Error (GMRAE) measures [18, 25]:

$$MRAE = mean(|r_t|)$$

$$MdRAE = median(|r_t|)$$

$$GMRAE = gmean(|r_t|)$$

A deficit of the measures based on relative errors is that e_t* may have small values [18]. However, Armstrong and Collopy (1992) recommend using this type of measure, emphasizing the GMRAE and MdRAE. Fildes (1992) also recommends the GMRAE measure. [18].

2.3.0.4 Relative measures

Another way to scale the accuracy measure of a model is to divide it by the value of the same accuracy measure obtained by a benchmark model. Assuming MAE_b to be the MAE value obtained by the benchmark model, the relative MAE is given by RelMAE = MAE/MAE_b [18, 25]. In the same way, the relative measures of RMSE, MAE, MAPE, among others, can be obtained.

The Theil's U statistic, also called U2, is the relative measure of RMSE when the reference model is a random walk and only one-step forecasts are made [18]. This nomenclature can also refer to the relative measure of RMSPE [18].

The random walk, also known as a naive method, is indeed the most commonly used benchmark in these types of measures. In this benchmark model, the forecasted value, Ft, is equal to the last observation [18]. Another alternative as the benchmark is the mean method, where the forecasted value, Ft, is equal to the mean of all observations [18].

Relative measures are easy to interpret: they measure how much a proposed model is capable of improving the accuracy of the benchmark model [18]. For example, if the relative MAE has a value less than 1, RelMAE < 1, the proposed model is better than the benchmark model. Otherwise, if RelMAE > 1, the benchmark model remains a better option for the forecasting purpose.

As a disadvantage, these measures can only be computed if there are several forecasts in the same time series, to make it possible to calculate the MAE or MSE value [18]. It is not possible to measure out-of-sample forecast accuracy over a single forecast horizon using these measures [18].

2.3.0.5 Scaled measures

Hyndman (2006) proposed a new type of measure that overcomes the difficulties of the commonly used measures described in the previous sections. The scaled measures proposed are obtained by scaling the error on the in-sample MAE value obtained using a naive forecast method [18]:

$$q_t = \frac{e_t}{\sum_{i=2}^{n} |Y_i - Y_{i-1}|}$$

The scaled error, qt, is less than 1 when the model forecast is better than the average one-step naive model. Otherwise, when the forecast is worse than the one-step naive model, the scaled error is greater than 1. Scaled errors are obtained using this scaled error, qt, as a basis, resulting in the Mean Absolute Scaled Error (MASE), the Median Absolute Scaled Error (MdASE), and the Root Mean Squared Scaled Error (RMSSE):

$$MASE = mean(|q_t|)$$

$$MdASE = median(|q_t|)$$

$$RMSSE = \sqrt{mean(q_t^2)}$$

As the name implies, these measures do not depend on the scale of the data. If the value of MASE is less than 1, the proposed method has, on average, smaller forecasting errors than the errors obtained by the naive one-step method [18]. The same reasoning applies to the other scaled measures, MdASE and RMSSE. When multi-step forecasts are being calculated, it is possible to scale the error, qt, by the in-sample MAE obtained from a multi-step naive method instead [18]. When computing the RMSSE measure, it is preferable to scale the error, qt, on the in-sample RMSE of the naive method [18].

Hyndman (2006) proposes the scaled measures as the standard approach for comparing the forecasting accuracy on different datasets with different scales. These measures have a scale with a meaning that is easy to interpret [18]. In addition, its value would only be infinite or undefined if all historical values were equal, a situation in which a forecasting model would be of no use [18]. Among these, Hyndman (2006) prefers the MASE measure, as it is easier to interpret and less sensitive to outliers than the RMSSE measure, as well as less variable than the MdASE measure in small datasets.

In short, the study by Hyndman (2006) concludes that MASE is the best measure to be used when there are data sets with very different scales, as well as when the data have many values close to or equal to 0. However, there are contexts where it may be preferable to use simpler measures, when conditions permit: for example, if all the series are on the same scale, it may be preferable to use the MAE measure, or if all the data is positive and much larger than zero, it may be preferable to use the MAPE measure [18].

Chapter 3

Methodology and Experiments Setup

In this chapter we will explore and describe the methods for selecting datasets to be used in the study, as well as the preparation and cleaning of the selected datasets. We also describe the Feature Engineering perdormed on the datasets variables and the implementation and evaluation of the Models used in this study.

3.1 Datasets Overview and Selection

To train the developed models, datasets were searched on the *Kaggle* and *Google Datasets* websites. From the different datasets selected, we take into account whether:

- datasets present historical information on products' sales or demand
- the data contains static variables, i.e. variables that do not change over time, with information related to the products (such as brand, color, category, etc.) or locations where they were sold (store, state, country, etc.)
- time-series are representative of the reality of sales in retailers (e.g. intermittent sales and sales volatility, time series of products with different sales magnitudes, etc.)

After an initial selection of some datasets from these websites, their information was summarized in Table 3.1 to help in a more careful and informed selection of the datasets to be used in the study of the models.

Many of the datasets found did not have static variables with product information - as some of these datasets aggregate sales of products by category or brand, losing these variables as possible attributes. Most of the datasets that presented historical sales data did not present static variables with information about the products or the stores/locations where they are sold. Although we found datasets with product titles/names, these were not very descriptive and, therefore, it was impractical to extract product attributes from them. When the datasets had information on the attributes of the products, already organized in columns or contained in well-structured product titles, the time-series were not suitable for the study: time-series were too short, with 5 time-points or less, or they were time-series for products that sell in very small quantities regardless of

the aggregation, as is the case of electronic products, and that are therefore not suitable for our work.

In short, for several reasons, it is difficult to find datasets that reconcile historical information on sales of different products, as well as static variables with information on products and the stores or locations where they are sold. After intense filtering, only 2 datasets were left that met these conditions - the Synthetic dataset, and the M5 Competition's dataset. These datasets are described and analyzed in more detail in the next subsections.

Table 3.1: Overview of Datasets Search

Title	Context	Columns Information	Advantages	Disadvantages	Final Decision
A synthetic data set with new product demand and characteristics	Dataset artificially created to support the study of a model developed to predict the demand for new products without historical demand data from the sales record of products with similar characteristics []. Information on 18 weeks of sales for 2000 products. Each product is characterized by category, brand, color, and price, whose values were also artificially generated.	**Products Information:** Category, Brand, Color, and Price **Sales Information:** Demand01, Demand 02, .., Demand18: Number of unit sales of a product in weeks 1, 2, .., 18.	The data are generated in such a way that relationships are guaranteed between the characteristics of the products and their unit sales over the 18 weeks of the time-series. That is, it is certain that the Category, Brand, Color, and Price of the products somehow impact on product unit sales and its variations over the 18 weeks. This may allow testing whether the model captures these impacts/effects.	There may be a risk that time series are too short for complex models to learn from them. However, if there are 2000 products, it is possible for the length of the time series to be compensated with information from several products. This dataset also does not represent what happens in a real scenario of the retail market, where sales are volatile, irregular, and the relationships between variables are unlikely to be so obvious. As such, we need to select another dataset with real data from a retailer.	These data present time series of several products and guaranteed relationships between static variables with product information and the demand values to be forecast. As such, it is an interesting model to keep in the next steps of the study.
Supermarket sales	Dataset with historical sales of a company in 3 of its branches - A, B, and C. The data identifies the type of product sold, price, and quantity (but does not identify the specific product). Sales in the different Branches were recorded daily for 3 months.	**Purchase Information:** Product Line (category of the product sold), Unit Price, Quantity Date and Time, Tax, Total price, Payment used by customer - "Cash", "Credit card" and "Ewallet" **Store Information:** Branch of supercenter - "A", "B", and "C" City - Location of supercenters **Customer Information:** Gender, Type	Static Variables on locations where products are sold - Branch, City	Missing products' identifications - the products sold are identified by the category (product line) to which they belong. Thus, this static variable is lost as a feature, since we have no way of identifying the different products of each category and using this as a feature.	The dataset does not have any static feature with product information, since the sales of products are grouped by category - only 1 time series per category. It seems the dataset was created for analysis and customer segmentation.
Retail Dataset Analysis V.3	Dataset with information on sales of 5 chocolate brands by customers of different segments (gender, martial status, occupation, etc.).	**Customers Information:** Gender, Martial Status, Level of Education, Annual Income, Occupation, Size of the City the customer lives in **Purchase Information:** Identifier of Customer Day of purchase Quantity of chocolates of a Brand purchased by the customer **Products Information:** Prices of each Brand of chocolate over time	Variables on products - Prices for each Brand of Chocolate	No static variables with information on products, since the dataset only has a time series of sales of each chocolate brand.	Participants in this dataset seem to focus on the customer segmentation task rather than the retail forecasting task. The dataset format focuses on the transactions made by a customer when purchasing chocolates from one of the 5 brands (quantities bough of a brand in a specific date). Thus, this model seems to focus more on the behavior of different types of customers, characterized by the columns of their information, when purchasing the 5 brands of chocolate. As such, the dataset structure does not suit our problem.
English Data for Predict Future Sales Competition	Information on daily sales of electronic products in different stores. It contains supplementary information about the items sold, more specifically the title of the products, the price and the category of products to which it belongs to.	**Sales Information:** Date, Item, Shop **Products Information:** Item's Name/Title Category Price **Shops Information:** Shop's Name	Static Variables on Products - Category Dynamic Variable on Products - Price Titles for Products from where we may extract some new Static Variables/Attributes	The titles seem to be little or not descriptive at all, which makes it difficult to extract product attributes from these pieces of text. The description in Kaggle mentions that it also contains information about the stores where the products are sold, but opening the dataset it appears to have only the name of the store.	After analyzing some-time series of the dataset, we saw that most of the values of these series remain between 0 and 2, or a little above. This is due to the fact that the products in this dataset are largely electronic, with a tendency to sell in smaller quantities per day. As such, excluding time-series whose median sales magnitudes are very low could lead to the exclusion of most time-series. Even trying different aggregations these values remain quite low, so we decided to analyze other datasets.
Online Product Sales	This dataset is used in a competition where the participants must predict the online sales of consumer goods based on the features data of these same products. As such, the available dataset contains information on products' sales as well as their categorical and quantitative features.	**Sales/Demand Information:** Outcome_M1, .., Outcome_M2: 12 columns with monthly online sales for the first 12 months after the product lanches **Marketing Information:** Date_1 : Day that the major advertising campaign began and the product launched Date_2: Day that the product was announced and a pre-release advertising campaign began **Product Information (Features):** Quant_x: quantitative variables (x has different numbers to identify different features) Cat_x: categorical variables (x has different numbers to idenfity different features). Binary categorical variables measured as 1 or 0.	The data present several quantitative and qualitative attributes of the products. It can be a good dataset to test our models, as we also want to develop models capable of exploiting product attributes in order to improve sales forecasting of products.	The features - Quant_1, ..., Quant_x, Cat_1, .., Cat_x - were subject to anonymity and lost meaning, so it is not possible to make any interpretation of them - their analysis and preparation becomes more difficult and risky.	The anonymity of the various features of the dataset's products and, consequently, their difficult interpretation and analysis was the aspect that made us not include this dataset in the rest of the study (at least not as a priority).
Grupo Bimbo Inventory Demand	In this competition, participants are challenged to accurately predict sales of more than 1000 Bimbo consumer products in more than 80,000 stores. These products have descriptive titles, from which you can easily extract attributes such as brand, product type, weight, number of pieces, ingredients, etc.	**Products Information:** Descriptive Title: with brand, type of product, weight, number of pieces, ingredients. Price. **Unit Sales and Stores Information**	This dataset presents actual data for various products from a renowned consumer products retailer. In addition, the structure of product titles allows the easy extraction of attributes such as weight, number of pieces, brand, type of product, ingredients it contains, etc.	Time-series present very little information, many of them containing only sales for 3 or less timepoints in the past. Thus, this dataset is used in the study and development of models capable of predicting new products without any sales history from the sales history of products with similar characteristics. This problem is also known as attribute-based forecasting.	Despite being an interesting dataset, we will not take it as a priority in the study, since such a small amount of timepoints per time-series can make an initial phase of the development of our study too difficult.
M5 Competition	Latest in a lineage of competitions created to promote the development of models and methods capable of accurately forecasting retail product sales	**Calendar Information:** Date, Events, Snap Days **Prices Information:** Week, Store, Item, and Price **Sales Information:** Item id, Category, Department, Store, State, Demands	The dataset used in this competition was provided by Walmart and presents 2 key aspects to represent the real sales situation at retailers: 1. The inclusion of explanatory variables such as sales prices, promotions, days of the week, and special events that typically affect unit sales and may improve forecast accuracy. 2. The inclusion of time-series that show intermittency.	N/A	This dataset fits our purpose as it was provided by a larger retailer and contains static variables on products, such as category, and location, such as department, store, and state.

3.1.1 Synthetic Dataset

This synthetic data was created to test the model developed in the study [30]. The developed model aimed to predict the sales of new products inserted in a retailer and without any sales history through the analysis of previously available sales history of similar products, that is, products with similar characteristics.

As such, this dataset was artificially created with information on the sales and features of 2000 products. Each dataset entry corresponds to a product, with 18 columns corresponding to its sales (product units) in the first 18 weeks after its launch and 4 columns corresponding to the product's features, namely *Color*, *Category*, *Brand*, and *Price*.

The data were generated so that there are guaranteed relationships between the characteristics (features) of the products and their weekly or total demands, more precisely:

- **Color** and **Price** influence the level (magnitude) of product's **Demand** (number of product units sold) throughout 18 weeks.
- **Category** and **Brand** influence the products' **Profile**, that is, how the demand for products (number of units sold) varies over the weeks.

More information on how the data was generated is available in the article [30] and the link of the dataset [26].

3.1.1.1 Synthetic Dataset Description

Each entry has information on the weekly sales of a specific product - 18 columns that keep the number of units sold in the first 18 weeks after its launch. The dataset has 2000 entries corresponding to 2000 time-series (one for each product) with 18-time points each (weekly demand).

The demand for a product in the first 18 weeks of its launch is defined by the combination of its **Total Demand** and **Profile**:

- **Total Demand:** Sum of the total demand for a product in the 18 periods. In this dataset, the Total Demand values of the products are generated using a *Gamma distribution* with parameters $\alpha = 2$ and $\beta = 3$ [x].
- **Profile:** A product's Profile indicates how its demand varies from period to period over the 18 weeks after its launch. In this dataset, each product has assigned one of **3 possible profiles**:
  - **Increase:** 10% exponential increase of the product's demand per time-period.
  - **Decrease:** 10% exponential decrease of the product's demand per time-period.
  - **Stable**

The demand values of a product for each of the 18 weeks are obtained by multiplying its Profile to its Total Demand. Some noise is applied to each week's demand values, which is generated through a *Normal Distribution* with a coefficient of variation of 0.25.

- Demand Columns:
  - **Demand01, Demand02, ..., Demand18:** Demand values for a product in each of 18 weeks (column number corresponds to a week number, ordered from 1 to 18)
  - **TotalDemand**

Each product is characterized by 4 features, namely its **Color, Category, Brand**, and **Price**. Color, Category, and Brand are categorical variables, while Price is a numeric variable. These features are represented by the columns described below.

- Product Features:
  - **Color:** Discrete variable with 10 possible values - previously chosen arbitrarily - *"Black", "White", "Gray", etc.*
  - **Category:** Discrete variable with 10 possible values - previously chosen arbitrarily - *"Accessories", "Sound", "Smart home", etc.*
  - **Brand:** Discrete variable with 10 possible values - previously chosen arbitrarily - *"Octozzy", "Otise", "Mudeo", etc.*
  - **Price:** Continuous numeric variable. Each row of this dataset, corresponding to a product, has a price value. As such, the price of a product does not vary over time.

As mentioned above, the products features/characteristics values are assigned so that there are guaranteed relationships between the products' features and their demand values (total and over the 18 weeks). For such:

- The Category and Brand of a product relate to its Profile - increase, decrease, or stable. As such, 80% of the values assigned to the Brand and Category of a product relate to a specific profile - using categorical distributions. The remaining 20% assigned values relate to a random profile [26].
- The total demand of a product is divided into 5 equal segments (0-20th, 20-40th, 40-60th, 60-80th, and 80-100th percentiles). 80% of the values assigned to the Color of a product relate to a specific segment [26].
- The numeric values of the Price column are assigned to the products in a way that it is inversely proportional to the products' total demand ($Price = 2000/Demand$). Some noise is added to the Price values [26].

The Demand Segment and Profile of each product are identified in the **TrueDemandSegment** and **TrueProfile** columns of the dataset. The columns are not included in the analysis of the dataset, as they were created for the purpose of generating the synthetic data.

3.1.2 M5 Competition

The M5 competition is the latest in a lineage of competitions created to promote the development of models and methods capable of accurately forecasting retail product sales. The M5 competition was subdivided into two competitions: 1. in the first one, participants aim to develop methods capable of predicting point forecasts for all time-series of the dataset with maximum precision; 2. in the second, participants are challenged to develop methods that include confidence intervals in their predictions (which correspond to median values together with 50%, 67%, 95%, and 99% prediction intervals) [20].

The dataset used in this competition was provided by Walmart and presents 2 key aspects to represent the real sales situation at retailers:

- The inclusion of **explanatory variables** such as sales prices, promotions, days of the week, and special events that typically affect unit sales and may improve forecast accuracy.
- The inclusion of time-series that show **intermittency**.

3.1.2.1 M5 Dataset Description

The M5 dataset was provided by Walmart and contains information on sales of 3049 products in 10 stores across the United States in the form of time-series. Products can be classified into 3 different categories - *"FOODS"*, *"HOBBIES"*, or *"HOUSEHOLD"* - as well as one of the 7 departments where they are sold (and in which the mentioned categories are disaggregated). Stores can be located in one of 3 states - *"CA" (California)*, *"TX" (Texas)*, or *"WI" (Wisconsin)*.

Each time-series contains historical sales data from 2011-01-29 to 2016-06-19, excluding the 28-day horizon suggested as a test by the competition. Thus, each product-store pair is grouped into a time-series that contains a sales history of 1941 days. The organization of the time-series is represented in the figure 3.1 provided by the competition:

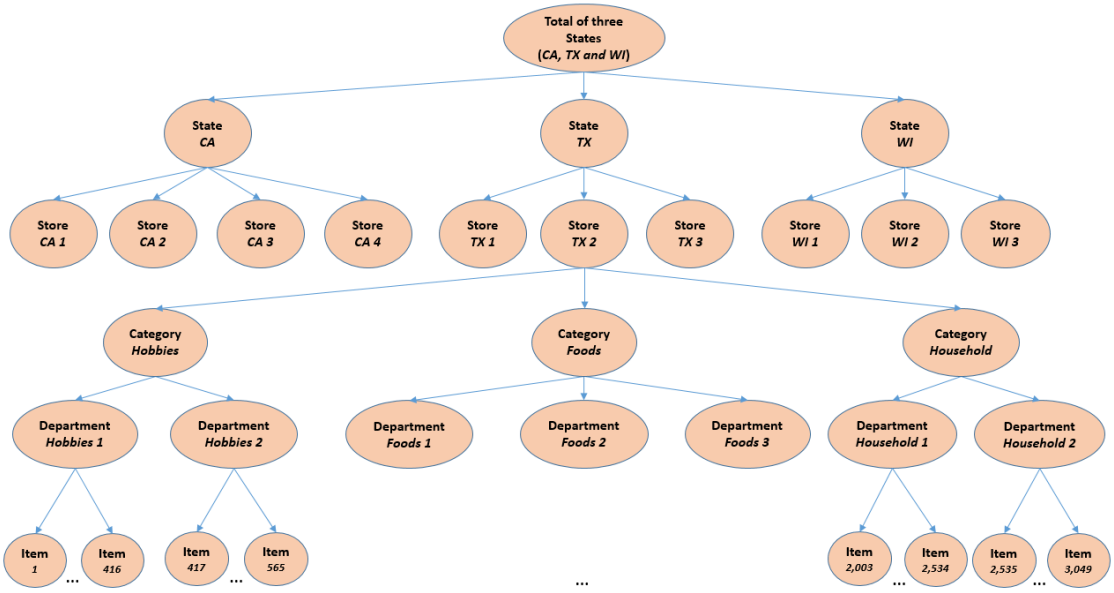


Figure 3.1: M5 Competition's time-series organization [20].

Thus, the units sold at the base level, that is, for each product-store pair, can be aggregated by product departments, product categories, stores, and geographical areas (3 states), resulting in different levels of aggregation with different numbers of time-series, as shown in the following table:

Level 1	Unit sales of all products, aggregated for all stores/states	1 time-series
Level 2	Unit sales of all products, aggregated for each State	3 time-series
Level 3	Unit sales of all products, aggregated for each Store	10 time-series
Level 4	Unit sales of all products, aggregated for each Category	3 time-series
Level 5	Unit sales of all products, aggregated for each department	7 time-series
Level 6	Unit sales of all products, aggregated for each State and Category	9 time-series
Level 7	Unit sales of all products, aggregated for each State and Department	21 time-series
Level 8	Unit sales of all products, aggregated for each Store and Category	30 time-series
Level 9	Unit sales of all products, aggregated for each Store and Department	70 time-series
Level 10	Unit sales of product x, aggregated for all stores/states	3049 time-series
Level 11	Unit sales of product x, aggregated for each State	9147 time-series
Level 12	Unit sales of product x, aggregated for each Store	30490 time-series

The dataset information is distributed in 3 files with Calendar, Prices, and Sales information:

- Calendar File's Columns
  - **date:** date in a "YYYY-MM-DD" format.
  - **wm_yr_wk:** identifier of the week the date belongs to.

  - **weekday:** day of the week with 7 possible values - *"Monday"*, *"Tuesday"*, *"Wednesday"*, *"Thursday"*, *"Friday"*, *"Saturday"*, *"Sunday"*.
  - **wday:** identifier of the weekday, that is, integers starting at 1 and identifying the 7 values of *weekday* column, starting with Saturday.
  - **month:** month of the date, identified by integers from 1 to 12.
  - **year:** year of the date, comprising values from 2011 to 2016.
  - **event_name_1:** if date includes an event, the name of that event. Null otherwise.
  - **event_type_1:** if the date includes an event, the type of that event, Null otherwise.
  - **event_name_2:** if date includes a second event, the name of that event. Null otherwise.
  - **event_type_2:** if the date includes a second event, the type of that event. Null otherwise.
  - **snap_CA**, **snap_TX**, and **snap_WI:** binary variable indicating whether the stores of *"CA"*, *"TX"* or *"WI"* allow SNAP purchases on the examined date. If one of this variables is set to 1, the SNAP purchases are allowed in the corresponding State on the examined date.

- Prices File's Columns
  - **store_id:** identifier of the Store where the Product is sold.
  - **item_id:** identifier of the Product being sold.
  - **wm_yr_wk:** identifier of the week in which the identified Product sells at the specified Price in the specified Store.
  - **sell_price:** the price of the Product for the given Week/Store. The price is provided per week (average across seven days).

- Sales File's Columns
  - **item_id:** identifier of the Product being sold.
  - **dept_id:** identifier of the Department the Product belongs to.
  - **cat_id:** identifier of the category the Product belongs to and has three possible values - *"FOODS"*, *"HOBBIES"*, *"HOUSEHOLD"*
  - **store_id:** identifier of the Store where the Product is sold.
  - **state_id:** identifier of the State where the Store is located, and has three possible values - *"CA"*, *"TX"*, *"WI"*
  - **d_1, d_2, . . . , d_i, . . . d_1941:** number of units sold at day i, starting from 2011-01-29.

3.2 Datasets Preparation and Preliminary Analysis

The process of preparation and cleaning of the Synthetic and M5 datasets will be described in the following subsections.

3.2.1 Preparation and Cleaning

3.2.1.1 Synthetic Dataset

Cleaning: Since the synthetic data was artificially generated, there are no null-values in any of the Color, Price, Brand, or Category columns. The same is true for the columns relating to the demand for products, that is, both the 18 columns corresponding to weekly sales and the column for total demand only have non-null and non-negative values. The equality between the TotalDemand column and the sum of the weekly sales columns (Demand01 to Demand18) was also verified in order to confirm their correctness. Thus, as expected, there was no need to clean up the data since all values generated for the columns are within domains that make sense.

Melting: We rearrange the data to a format that makes it easier to plot the demand over time using the *matplotlib* and *seaborn* libraries of the *Python* language. In the new format, each row stores information on a product's time-point, that is, the demand for a specific product in a given week (being the weeks ordered from 1 to 18 for each product). The **ArticleCode** (identifying the product)**, Color, Brand, Category,** and **Price** columns are kept. the TotalDemand column is removed and the 18 weekly demand columns (Demand01, Demand02, ..., Demand18) are reorganized into 2 new columns:

- **Week:** For each row, it stores the week a product (identified by the ArticleCode column) was sold. Values in this column range between 1 and 18 for each product.
- **Demand:** For each row, it stores the demand value (number of units sold) of the product (identified by ArticleCode) in a given period (identified by the Week column).

As such, the new format has 36000 rows, corresponding to the different time-points of the 2000 products of the original dataset. Both original and new sets of data are used to generate descriptive graphs of the data in the analysis.

3.2.1.2 M5 Dataset

The three files with calendar, sales, and products information are loaded as Dataframes using *Pandas*. The objective is to aggregate the information from these files in a single dataset with a tabular format, which can be easily restructured and processed to serve as input to the developed models.

Downcast: The Sales dataframe has daily information on the sales of products in the different

stores for a period of 1941 days, which corresponds to a total of information of 59181090 time-points. Information from the remaining dataframes will be added to each of these time-points, to include information on the product sold and the day on which the sales occur. To make this process of combining large amounts of tabular data, as well as future processing of them, more efficient and less time-consuming, it is necessary to reduce the memory occupied by these dataframes as much as possible without losing any information.

To this end, Calendar, Sales, and Products dataframes were downcast using Python code. This process consists of:

- **Numerical Columns:** reducing *Float* and *Integer* columns of a dataframe to the smallest number of bytes that allow representing all values contained in those columns.
- **Categorical Columns:** converting *String* columns to the *Category* type of the Pandas library.

These steps considerably reduced the amount of memory occupied by the three dataframes.

Melting: As can be seen in the description of the datasets, the Sales data frame has 1941 columns, from *d_1 to d_1941*, which store the demand values of the different Product-Store pairs on the different days of the time-series, where each row corresponds to a product-store pair and, thus, to a time-series. As such, the Sales dataframe was rearranged in a similar process to that performed in the Synthetic dataset.

The *item_id, cat_id, dept_id, store_id, state_id* (identifying the product, its category, as well as the department, store, and state where they are sold) are kept. The 1941 daily demand columns (*d_2, d_2, ..., d_1941*) are reorganized into 2 new columns:

- **d:** For each row, it stores the identifier of the day (*d_2, d_2, ..., d_1941*) it corresponds to. The row holds information on the demand of a specific Item-Store pair on that day.
- **quantity:** For each row, it stores the demand value (number of units sold) of a specific Product-Store pair (identified by *item_id* and *store_id* columns) in a given day (identified by the *d* column)

The new format has 59181090 rows, corresponding to the different time-points of the 30490 product-store pairs of the original dataset. The *d* column of this new format allows merging the Sales information with the Calendar data frame, which contains information on the date, events, and discounts of the day identified by this column.

Combination: As stated above, the sales information (melted to the new format) is merged with calendar information by column *d*. The dataset resulting from this combination is further merged

with information on the prices of the different products at the different stores each week, through the columns *item_id*, *store_id*, and *wm_yr_wk*.

Cleaning: In this step, we remove null values from the resulting dataset.

- **Numerical Columns:** in numerical columns, such as *sell_price*, null values are replaced by the mean of the remaining column values of the time-series it belongs to.
- **Categorical Columns:** in categorical columns, null values are replaced by a new default category identifying the abscence of value. For example, in the event columns such as *event_type_1*, *event_type_2*, *event_name_1*, and *event_name_2*, the null value is used when no event takes place in that day. As such, the null values of those columns are replaced by the "No Event" category. Creating a new category to replace null values will simplify the analysis of datasets and the generation of features from categorical variables.

The dataset resulting from the preparation encompasses all the information needed to train the forecasting models. Regarding the format, each row of the dataset corresponds to the demand for a specific product in a specific store. As such, the dataset has information on all time-points of all Product-Store pairs, which corresponds to level 12 of aggregation of the M5 dataset (as can be seen in table 3.1.2.1). The format of this dataset can be easily re-structured to one of the 12 levels of aggregation referenced in the table 3.1.2.1.

3.2.2 Preliminary Analysis

3.2.2.1 Synthetic Dataset

In Figure 3.2, one can observe that the demand values of products of the same color tend to remain in a consistent range of values (level) over the 18 weeks. This is made even more evident by the fact that demand plots of different colors rarely intersect on the chart. Products with colors *"Red"* or *"Brown"* tend to have much higher demand values over the 18 weeks compared to products with other colors, with the values remaining above 25 and below or close to 40 in all periods. Further down are the products with color *"Blue"* or *"Purple"*, whose demand values remain above 15 and below or close to 25 over the 18 weeks. In the same way, the products of the other colors also keep the demand within a range of values over all periods of the plot. Figure 3.3 also shows this property of the products of different colors.

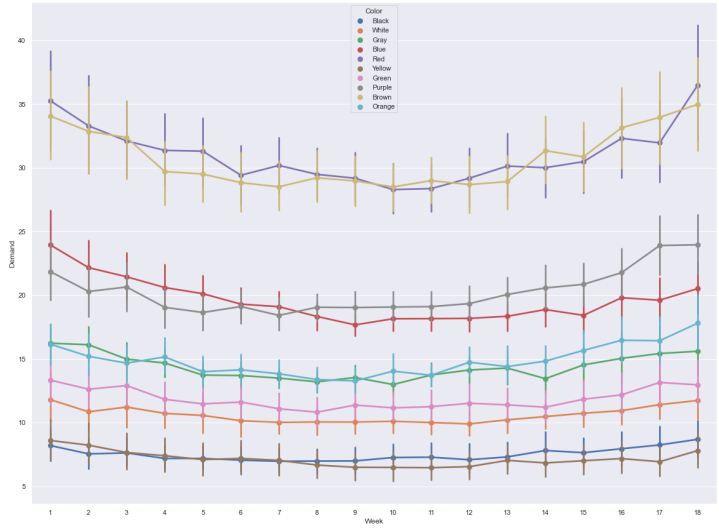


Figure 3.2: Pointplot of products' demands over time grouped by colors - Synthetic Dataset.

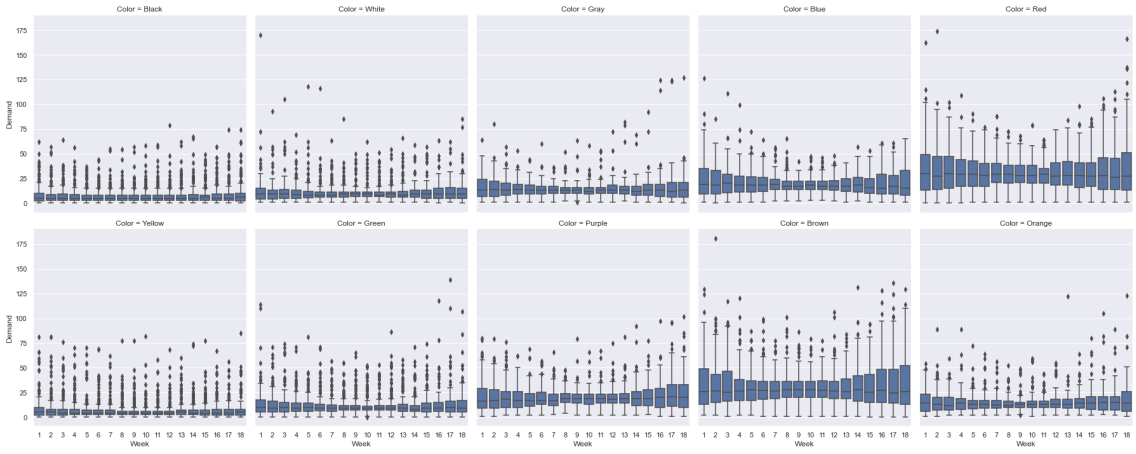


Figure 3.3: Boxplot of products' demands over time grouped by colors - Synthetic Dataset.

Figure 3.4 (and the one in Appendix A.1) show that products from each category may exhibit 1 of 3 well-defined behaviors:

- Demand values increase from week to week over the 18 weeks - products from categories *"Sound"*, *"Smart home"*, *"Television"*, and *"Kitchen"*.
- Demand values decrease from week to week over the 18 weeks - products from categories *"Accessories"*, *"Photography"*, and *"Tablets"*.
- Demand for products remains constant over the 18 weeks - products from categories *"Telephone"*, *"Games"*, and *"Computers"*.

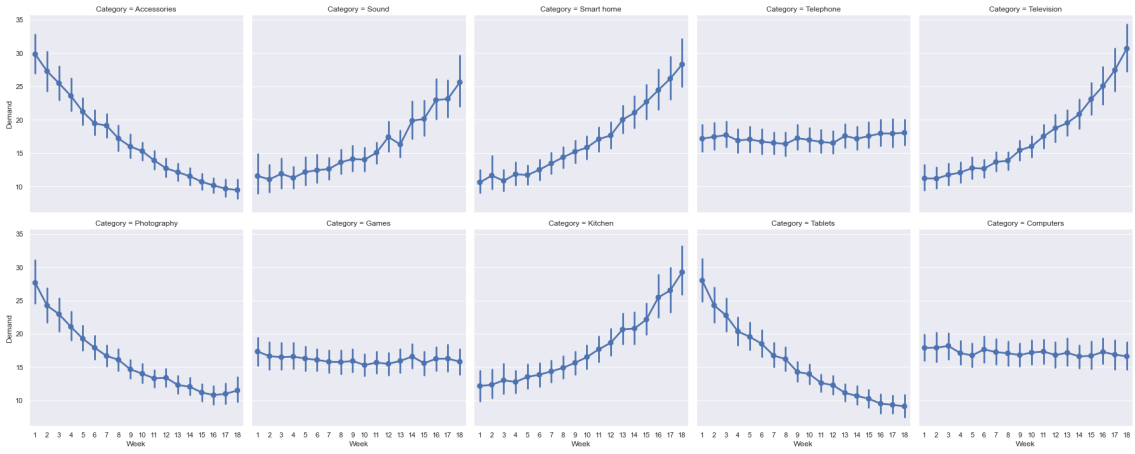


Figure 3.4: Pointplot of products' demands over time grouped by categories - Synthetic Dataset.

Figure 3.5 (and the one in Appendix A.2) show that products from a specific brand exhibits 1 of 3 well-defined behaviors:

- Demand values increase from week to week over the 18 weeks - products from brands *"Octozzy"*, *"Outise"*, *"Mudeo"*, and *"Animity"*.
- Demand values decrease from week to week over the 18 weeks - products from brands *"Supranu"*, *"Transible"*, and *"Kayosis"*.
- Demand for products remains constant over the 18 weeks - products from brands *"Dynotri"*, *"Hyperive"*, and *"Verer"*.

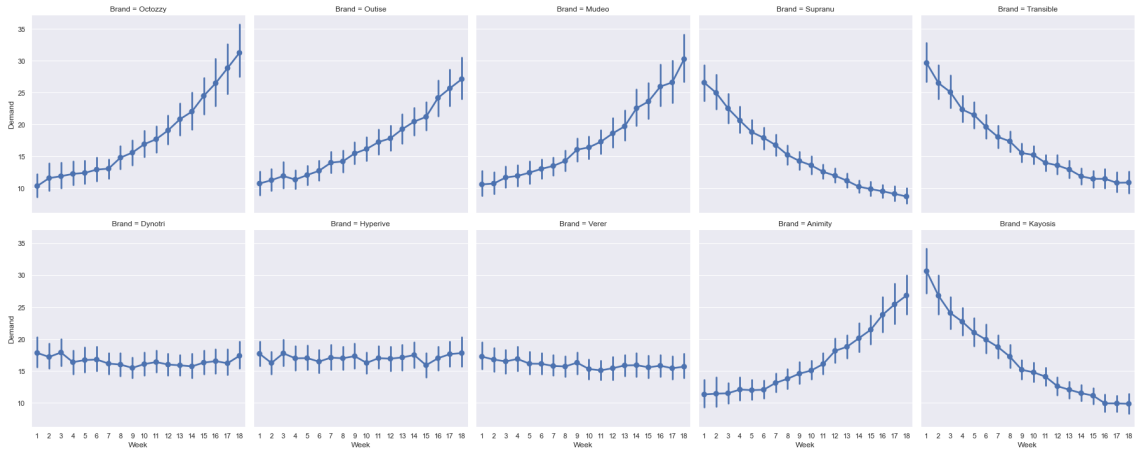


Figure 3.5: Synthetic Dataset - Pointplot of products' demands over time grouped by brands.

The scatterplots in 3.6 show that, in all 18 weeks, the demand values of the products maintain an inversely proportional relationship to the values of their prices.

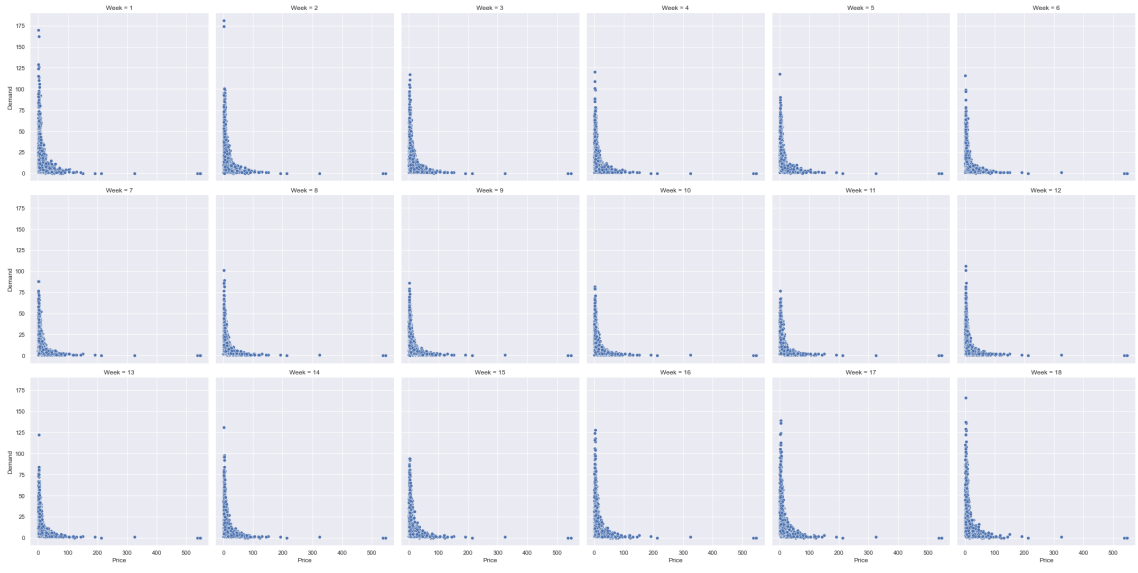


Figure 3.6: Plotting the prices of all produts in each week against their demands - Synthetic Dataset.

3.2.2.2 M5 Dataset

We consider 4 different levels of aggregation to analyze the M5 dataset:

- **Level 9:** 70 time-series with information on the unit sales of all products, aggregated for each Store and Department. The *item_id* column is discarded as the sales of different products are grouped by Department and Store. However, this aggregation keeps static variables with information on the category (*cat_id*) of product groups sold in the different

Department-Store pairs, as well as information on the Department, Store, and State (*dept_id*, *store_id*, and *state_id*) where these product groups are sold.

- **Level 10:** 3049 time-series with information on the total sales of each Product in all stores/states. The *store_id* and *state_id* columns are discarded as all unit sales are aggregated by Product. This aggregation keeps static variables on the Category of the Product being sold (*cat_id*), as well as the Department (*dept_id*) where it is sold.
- **Level 11:** 9147 time-series with information on the sales of each Product aggregated by State. The *store_id* column is discarded. The static variavles on the Category (*cat_id*) of the Product, as well as the Department (*dept_id*) and State (*state_id*) where it is being sold are kept.
- **Level 12:** 30490 time-series with information on the sales of each Product for each Store. All static variables with information on the product and location are kept.

Analyzing Figure 3.7 (as well as the ones in Appendix A.6, A.12, and A.19), there seems to be an increasing trend in the demand values for products in the three categories (*"FOODS"*, *"HOBBIES"*, and *"HOUSEHOLD"*), even if tenuous, at all levels 9, 10, 11, and 12 of aggregation. As these line plots are zoomed by shortening their time range, this trend becomes less and less evident or undetectable, while it becomes possible to verify what appears to be a weekly seasonality.

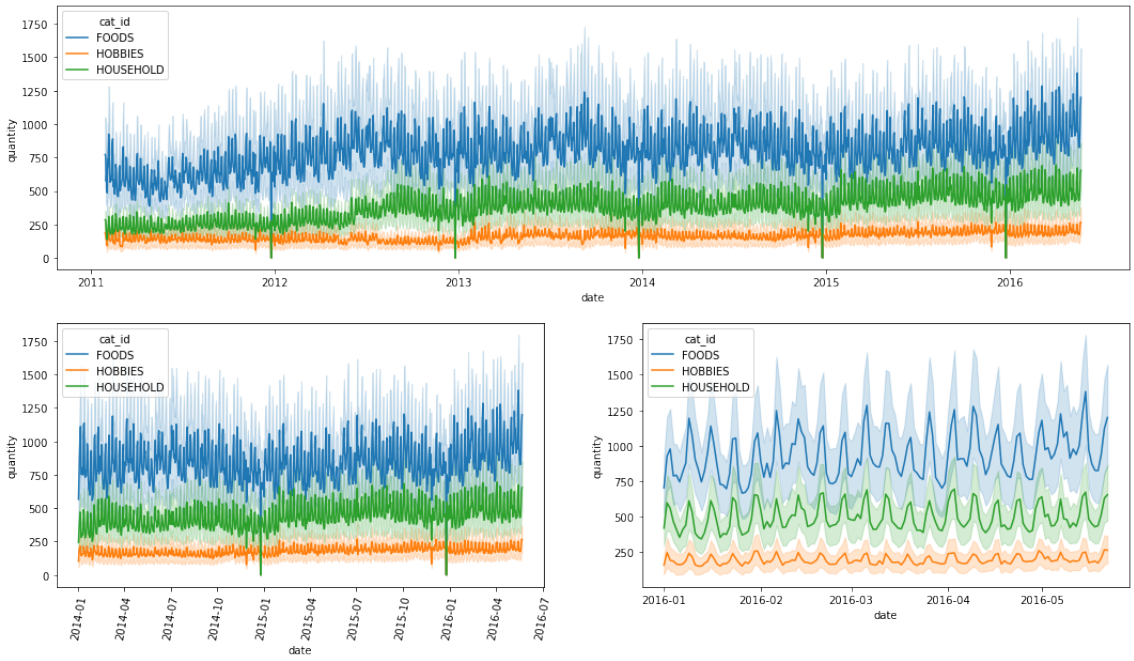


Figure 3.7: M5 Level 9 - Expected value and confidence interval of demand for products, by category, over time.

According to the Figure 3.7 (as well as the ones in Appendix A.6, and A.12), the products in

the *"FOODS"* category seem to tend to have higher unit sales values compared to products in the remaining categories - *"HOBBIES"*, *"HOUSEHOLD"* - for levels 9, 10, and 11 of aggregation. The box-plots in A.3, A.7, and A.13 confirm the statement, since the distribution of the values of units sold of products in the *"FOODS"* category reaches higher values in the three states, as well as in the 10 stores, compared to the other categories.

The boxplots in A.3, A.7, and A.13 also show that products in the *"FOODS"* category sell in larger quantities in the *"FOODS_3"* department at levels 9, 10 and 11 of aggregation. Likewise, products in the *"HOBBIES"* and *"HOUSEHOLD"* categories seem to sell in greater quantities in the *"HOOBIES_1"* and *"HOUSEHOLD_1"* departments, respectively. For aggregation level 9, products in the *FOODS*, and *HOUSEHOLD* categories tend to be sold in greater quantities at store *CA_3* than in other stores. For aggregation level 11, sales of products in the *FOODS*, *HOBBIES*, and *HOUSEHOLD* categories tend to reach higher values in the California (*CA*) state, when compared to the values of the other states (*TX* and *WI*).

From the scatter plots in Figure 3.8 (as well as in Appendix A.8, A.14, and A.21), it seems that the highest values of unit sales are achieved by the products with the lowest prices, which correspond mostly to products in the *"FOODS"* category. This may be one of the reasons for the tendency for food products to be sold in larger quantities, as mentioned above. Analyzing the scatter plot 3.8 corresponding to aggregation level 9, it seems evident that products of the same category are separated by departments through the prices at which they are sold. As such, the products in the *"FOODS"* category with the lowest prices are sold in the *"FOODS_3"* department, and they are also the ones with the highest sales quantities. The products of the *"HOUSEHOLD"* category with the lowest prices are sold in the *"HOUSEHOLD_1"* department, being those with the highest unit sales values within this category. On the other hand, the products in the *"HOBBIES"* category that achieve the highest sales values within the category are those that sell at the highest prices in the *"HOBBIES_1"* department.

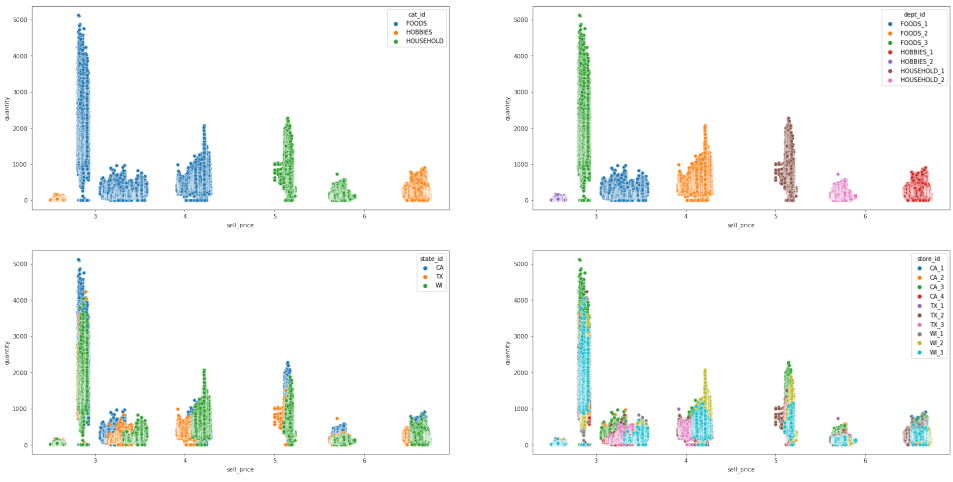


Figure 3.8: M5 Level 9 - Prices versus Demand of products on different days.

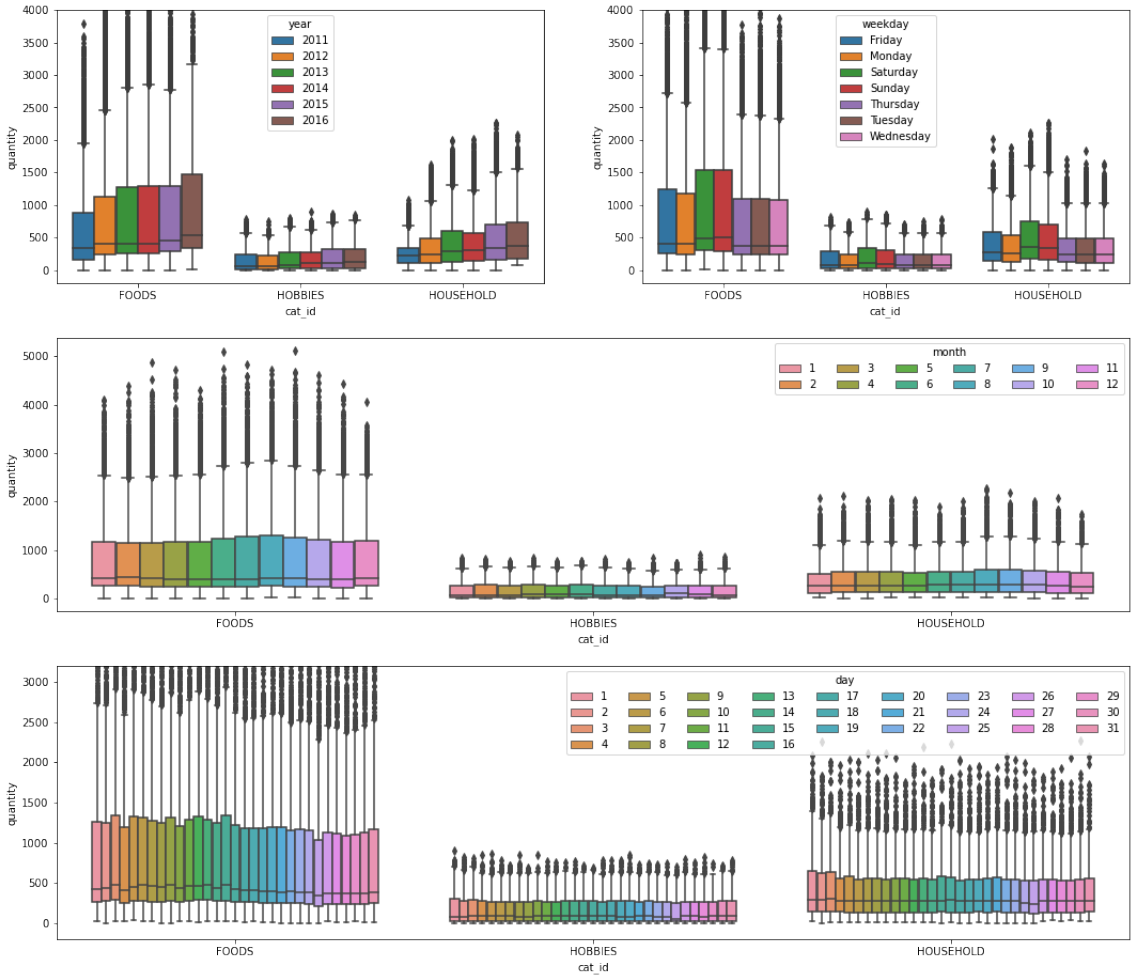


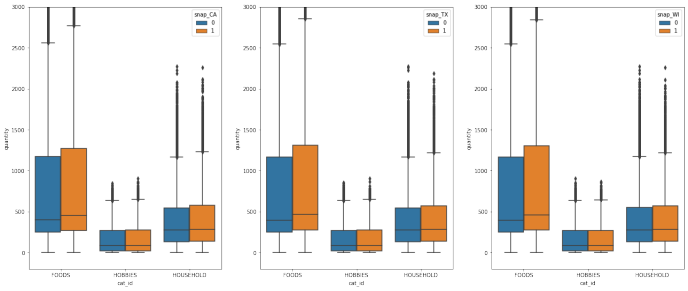
Figure 3.9: M5 Level 9 - Distribution of demand for products from different categories by Day of the Week, Day of the Month, Month, and Year.

The annual boxplots in 3.9, A.9, and A.15 show an increasing trend in the unit sales values of products in the *"FOODS"*, *"HOBBIES"*, and *"HOUSEHOLDS"* categories from year to year for levels 9, 10, and 11. At some levels of aggregation, this annual trend may be more evident in the products of one or more categories. The distribution of unit sales by days of the week shows that at aggregation levels 9 and 10, products in the *"FOODS"*, *"HOBBIES"*, and *"HOUSEHOLD"* categories have higher unit sales values on Saturdays and Sundays - this difference is more evident in products of *"FOODS"* and *"HOBBIES"* categories. At aggregation level 11, products in the *"FOODS"* category sell in greater quantities on *Saturdays* and *Sundays*, while products in the *"HOBBIES"* category sell in greater quantities on Fridays, Saturdays, and Sundays - there seems to be no evident difference in the unit sales of *"HOUSEHOLD"* products on different days of the week.

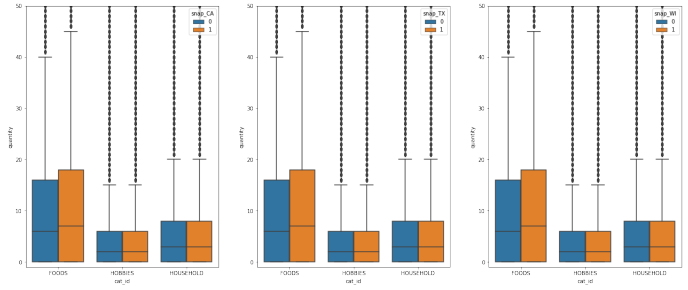
In the boxplots of A.4, A.10, and A.16 it is possible to observe that the values of unit sales for

products of the *"FOODS"*, *"HOBBIES"*, and *"HOUSEHOLDS"* categories decrease considerably during Easter. At aggregation level 9, unit sales increase considerably for the *"FOODS"* category during *"Father's Day"*, *"Cinco de Mayo"*, and *"Orthodox Easter"*. Sales of the *"HOUSEHOLD"* category also appear to increase during *"Cinco de Mayo"* and *"Father's Day"*. At aggregation level 10, unit sales values for products in the *"FOODS"* and *"HOUSEHOLDS"* categories seem to increase during *"Cinco de Mayo"* and *"Father's Day*. At level 11, the values of unit sales for products of the *FOODS* category increase during *Cinco de Mayo*, *Father's Day*, and *Orthodox Easter*.

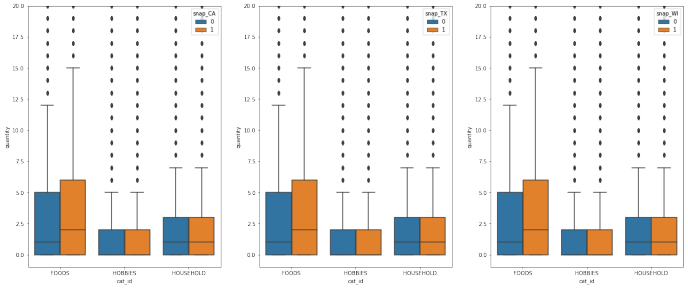
According to the boxplots 3.10a, 3.10b, and 3.10c of levels 9, 10, and 11 of aggregation, and as expected, the sales values of products in the FOODS category tend to increase when the SNAP variable of one of the states is set to True (snap_CA, snap_TX, or snap_WI is set to 1). The same is not true for the products of the remaining categories at these levels of aggregation.



(a) M5 Level 9



(b) M5 Level 10



(c) M5 Level 11

Figure 3.10: M5 Levels 9, 10, and 11 - Distribution of demand for products from different categories by snap_CA, snap_TX, and snap_WI.

In 3.11, A.11, A.18, and A.26 some examples of time-series were plotted for each of the 4 levels of aggregation, together with the respective partial (PACF) and complete (ACF) auto-correlation values. In the time-series examples of levels 9 and 10 of aggregation, the weekly seasonality of the target variable (unit sales) is evident - it is possible to observe a strong auto-correlation throughout the time-series with higher values in the lags multiples of 7. It is also possible to observe the increasing trend of product sales in several of these time-series. In the time-series examples of levels 11 and 12 of aggregation, the weekly seasonality becomes less evident. On the other hand, the intermittence and high volatility of products' sales become increasingly evident - time-series with time intervals in which there are no sales of a product (target variable equal to 0), followed by sales peaks.

	Level 9	Level 10	Level 11	Level 12
count	135870.0	5918109.0	17754327.0	59181090.0
mean	492.58	11.31	3.77	1.13
std	603.80	28.08	10.65	3.87
min	0.0	0.0	0.0	0.0
25%	136.0	0.0	0.0	0.0
50%	288.0	4.0	1.0	0.0
75%	552.0	11.0	4.0	1.0
max	5118.0	2532.0	1315.0	763.0

Table 3.2: Statistical (and descriptive) values of the target variable for each of the 4 levels of aggregation of the M5 competition.

After analyzing the graphs made for the aggregation level 12 of the M5 dataset, there seems to be no evidence of the relationship between exogenous variables, including static variables with information on product categories and sales locations, and the sales behavior of these products. This is mainly due to the fact that the unit sales values at this level of aggregation are distributed in a range of very low values, which seem to vary between 0 and 1, or 0 and 2, in most cases. The statistical values of the target variable of level 12 calculated in table 3.2 are proof of this, showing that about 75% of the time-points have a unit sales value between 0 and 1, with 50% of the sales values being equal to 0. This eliminates the need and purpose of a forecasting model for this level of aggregation. As such, **the aggregation level 12 of the M5 dataset was excluded in the training and testing of the models developed later in the study**.

At aggregation levels 9, 10, and 11, unit sales values are spread over a wider range of values. This can also be seen in the increase in quartiles and median values in table 3.2 for these levels of aggregation. Although 25% of the target values of aggregation levels 10 and 11 are equal to 0, this results mostly from products that did not have sales during the first years of the dataset, probably because they were not on the shelf, which resulted in time series where the first years

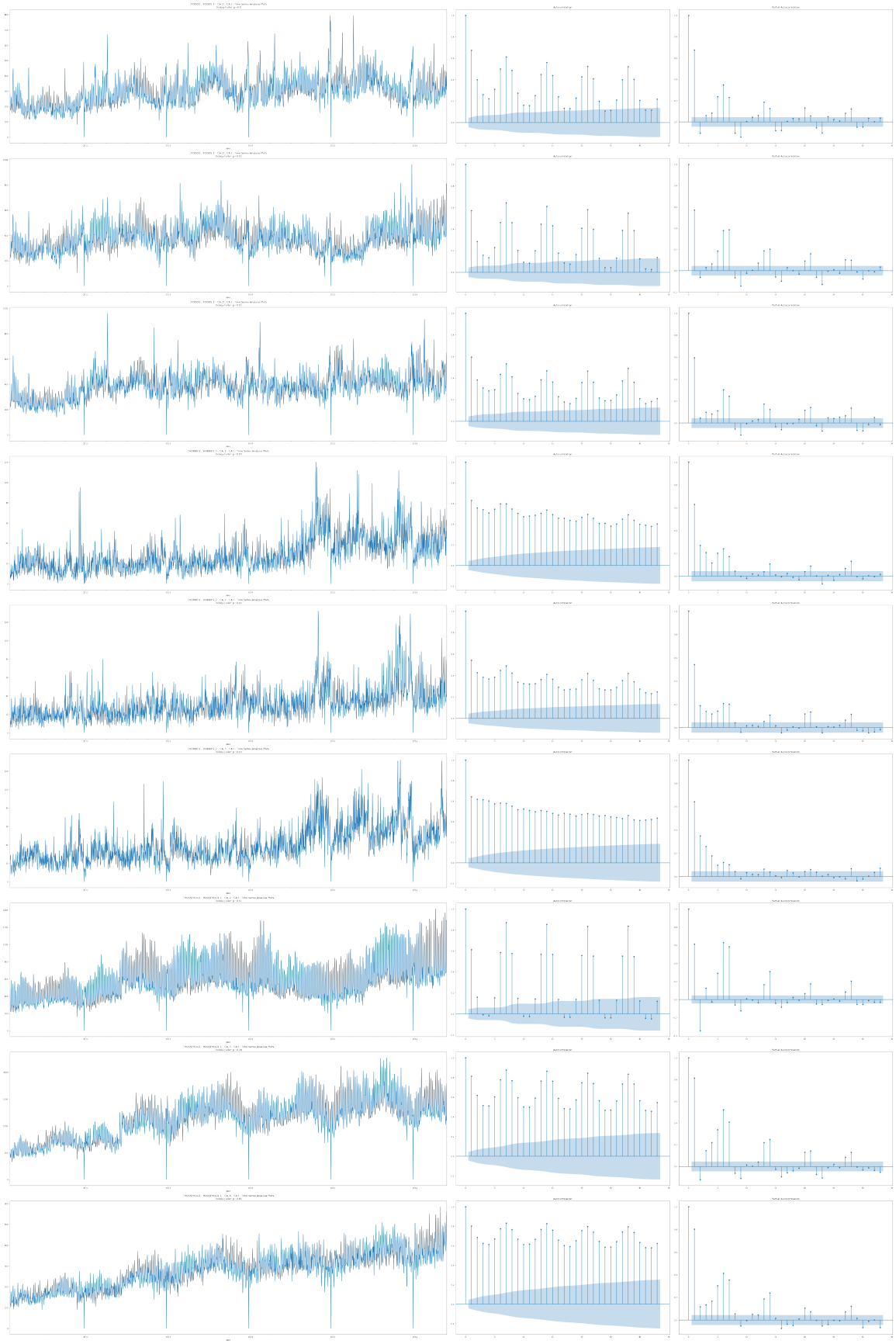


Figure 3.11: M5 Level 9 - Examples of dataset Time-Series, with respective partial (PACF) and complete (ACF) auto-correlation values.

are represented by a sequence of values equal to 0 (as shown in the example of figure x). Also, the analysis of these 3 levels of aggregation showed that the static variables with information on the products (category) and the places where they are sold (department, store, state), can hold valuable information on their sales behavior and probably increase the predictive power of the right forecasting models. As such, **levels of aggregation 9, 10, and 11 of the M5 dataset were used to train and evaluate the models developed later**.

3.3 Feature Engineering

Feature Engineering is the term assigned to any process that allows the creation of new useful input features for Machine Learning models, capable of increasing their predictive power, from existing features or variables.

3.3.1 Categorical Variables Encoding

The vast majority of Machine Learning models, including the neural networks of Deep Learning models, require all input variables to be numerical values. Thus, it is necessary to convert categorical variables, ie, variables where each value/category is represented by a Label/String, to a numeric format. There are several approaches to perform this conversion, being the following some of the most popular:

- **Ordinal Encoding:** This approach assigns a unique integer to each category value. The assigned integers usually start at 1 or 0 and can easily be reversed. This is used to encode categorical variables whose values have a **natural order or rank**, also known as ordinal categories.

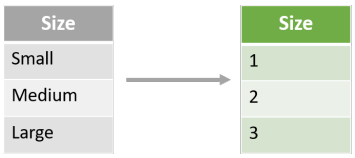


Figure 3.12: Example of applying Ordinal Encoding to the Size variable

  Integer values have a natural relation to each other that can be used to represent the relation between different categories of an ordinal variable, so Machine Learning models can understand and take advantage of these relations.

  Precautions: This type of encoding should not be used on categorical variables whose categories do not have a natural rank or order with each other. The Machine Learning model could misunderstand the existence of relationships between the different values of the categorical variable that do not exist.

- **One-Hot Encoding:** When there is **no relationship or order** between the different values of a categorical variable, the ordinal encoding may disrupt Machine Learning models by

allowing them to assume the natural ordering between categories that do not exist, leading to poor results. It becomes more advantageous to use one-hot encoding, in which a binary variable is created for each unique value of the categorical variable, that is, for each category. The binary variable corresponding to the value of a row is set to 1 and the remaining binary variables corresponding to the categorical variable are set to 0.

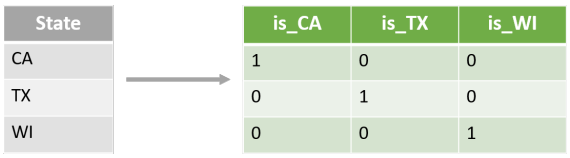


Figure 3.13: Example of applying One-Hot Encoding to the State variable

Precautions: If a categorical variable has a high number of categories, this encoding results in the creation of a large number of binary variables. This can cause an overload of features fed to the model, harming its performance and efficiency.

- **Dummy Variable Encoding:** As seen earlier, One-Hot encoding creates a binary variable for each category of a categorical variable. For example, for the 3 possible values (*CA*, *TX*, *WI*) of the *State* variable, 3 binaries (*is_CA*, *is_TX*, *is_WI*) are created. This type of representation presents redundancy, since we know that the *CA* category is represented by *is_CA*=1 and the remaining binary variables at 0, as well as the TX category is represented by *is_TX* = 1 and the remaining binary variables at 0, the last category, *WI*, does not need the extra binary variable *is_WI* as it can be represented by *is_CA*=0 and *is_TX*=0.

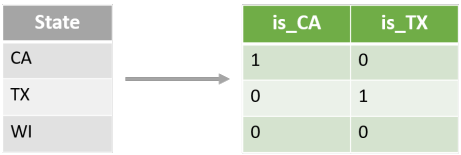


Figure 3.14: Example of creating Dummy Variables to represent the State variable

Dummy variables are an adaptation that allows representing *C* categories of a categorical variable using *C-1* binary variables, thus **removing the redundancy** of One-Hot encoding.

Precautions: Despite eliminating redundancy, creating Dummy Variables raises the same problems as One-Hot Encoding when a categorical variable has a very large number of categories (unique values).

Thus, we use **Ordinal Encoding** to represent categorical variables that store date information where there is an order between the different values: *Day of Week*, *Month*, *Day of Month*, *Year* in the M5 datasets.

We also use **Dummy Variables** to represent categorical variables where there is no order between the different values, such as:

- Columns with ***Products' Information***: *Color*, *Brand*, and *Category* columns in the Synthetic dataset, as well as *cat_id* column in the M5 datasets.
- Columns with ***Locations' Information***: *Department*, *Store*, and *State* columns in the M5 datasets.
- Columns with ***Events' Information***: *event_type_1*, *event_name_1*, *event_type_2*, and *event_name_2* columns in the M5 datasets.

3.3.2 Scaling

In real data, different numerical variables may have very different ranges of values. In this way, Machine Learning models are not able to compare different columns of numerical variables, as they may give greater importance to columns whose values are distributed over ranges of much higher magnitude.

It is important to carry out the scaling of the **numerical variables** so that they **become distributed over the same range**. There are two ways of scaling numerical columns:

- **Normalization:**

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

  Scales the values of all numeric variables to the same range. The simplest way of normalization scales all values for a range between 0 and 1, also known as min-max normalization 3.15a.

- **Standardization:**

$$X_{stand} = \frac{X - \mu_X}{\sigma_X}$$

  Scales the values of different numerical variables, taking into account the standard deviation of each numerical variable (σ_X). This scaling reduces the effect of outliers on a feature 3.15b.

X_1	X_2
1	100
2	200
3	300

X_1	X_2
0	0
0.5	0.5
1	1

(a) Min-Max Normalization

X_1	X_2
1	100
2	200
3	300

X_1	X_2
-1	-1
0	0
1	1

(b) Standardization

Figure 3.15: Examples of applying different scales to the X_1 and $X2$ features.

The *scikit-learn* library has many data preprocessing methods implemented and ready to use, including several Scaler objects to perform different types of normalization and standardization of numerical features, such as:

- *MinMaxScaler*, and *MaxAbsScaler* for normalization
- *StandardScaler*, and *RobustScaler* for standardization

Thus, ***MinMaxScaler*** was applied to all numerical features with different ranges of values, including numeric features that resulted from ordinal encoding:

- Columns with ***Prices' Information***: *Price* column on Synthetic and M5 datasets.
- Columns that ***resulted from Ordinal Encoding***: *Week of Day*, *Month*, *Day of Month*, *Year* on M5 datasets.

3.3.3 Lags

When Supervised Machine Learning models, such as decision trees, are used for time-series forecasting, the forecasting problem must be framed into a supervised problem. This process consists of creating lag features of the target variable to be fed to the supervised model as input features.

Being Y_t the value of the target variable at instant t and to be predicted, the lag values correspond to values of the time-series of the target variable before that instant. The first lag corresponds to the Y_{t-1} value, the second lag to the Y_{t-2} value, and so on.

Y_t
1
2
3
4
5

Y_{t-1}	Y_{t-2}	Y_{t-3}
-	-	-
1	-	-
2	1	-
3	2	1
4	3	2

Figure 3.16: Example of creating lag values for the target variable to be predicted (for a context window of length $N = 3$).

First, we choose the length for the context window, N, which will be the number of past values of the time-series to be predicted, Y, the model is able to observe. Then, the first N lags of the time-series Y are created, $X = Y_{t-1}, Y_{t-2}, ..., Y_{t-N}$, and fed to the supervised model as input features to

help predicting the value of the target time-series at instant t, Y_t. The *Pandas* library allows the easy creation of lag columns for the target variable.

There are other processes with higher complexity to create new features from existing features or variables. However, these extra feature engineering processes were not explored but kept as a possible further step.

3.4 Models

Many characteristics of retail products, as well as the stores where they are sold, are expressed in static variables, that is, variables that do not change over time and that, therefore, remain constant in a given time-series. These can be variables such as color, brand, the type and capacity of the product's package, or the weight of a product. As such, it becomes important to develop models capable of learning a global model from several time-series of different products, thus being able to take advantage of the variables of their characteristics, as well as other static variables.

3.4.1 Machine Learning Models - Decision Trees

Machine learning models based on decision trees, such as Random Forest and Gradient Boost, are frequently used by participants in Kaggle competitions. These models allow accessing the importance of the features used to train the models, though the percentage of nodes where a feature is used to divide the tree into decision groups. Thus, these models allow us to see which external variables are most used by the model in forecasting the sales of a product after being fitted.

Using these models for time-series forecasting purposes requires the historical sales data to be framed into a supervised problem using a sliding window approach [x]. Despite the utility described above, the use of these models raises some anticipated concerns:

- Unlike the Deep Learning models, decision trees are not able to learn new relevant features or relations from the ones provided as input during training. These models require the prior and manual creation and preparation of all features that will be fed as input and that hold relevant information. Creating such features often requires expertise.
- The results of these models are highly dependent on the quality and relevance of the features fed to the model during training and prediction. Including redundant features or features with little predictive information in the input data can result in the deterioration of fitted model predictions. This reinforces the need for expert labor in the creation and preparation of helpful features.
- After converting the time-series to a supervised problem, these models can train and learn to predict a target variable (y) that corresponds to the demand value of a time-series in the next unknown time period, also known as one-step forecasting. In order to do multi-step forecasting, that is, to predict more than one time-step into the future, it is necessary to adapt the supervised forecasting problem by opting for one extra step such as the direct strategy or recursive forecasting, among others [27].

In short, the extra manual work required to create features, the need for expert knowledge, as well as the extra code to prepare the models for the multi-step forecasting task, make these models more time-consuming, labor-intensive, and prone to errors. Despite the risks, these models can be fed

with information from all time-series, allowing the inclusion of static variables, including product or store characteristics, as features. Adding this to the possibility of accessing the importance values of the features used by the decision trees, made us consider both *Random Forest* and *Gradient Boost*. For this, we use the *Random Forest Regressor* implementation from the *scikit-learn* library [2] and the *XGBoost Regressor* from the *xgboost* library [3] with 500 estimators each.

To frame the data as a supervised problem we used the method described in 3.3.3, creating context windows of *lag* values that seem to have greater predictive power in the future values of the time-series, throughout the training periods of all time-series. To identify the lag features, we use a *t-{lag_value}* format, that is, *t-1*, *t-2*, ..., *t-n*, correspond to lags $1, 2, \ldots, n$. The analysis in 3.2.2 helped to select the lag values to be used as features. For example, as the weekly seasonality is very strong in the M5 competition datasets 3.2.2.2, to model the time-series we use the lag of the previous day, *t-1*, as well as lags that are multiples of 7 for the sales of the previous 4 weeks, *t-7*, *t-14*, *t-21*, and *t-28*. To adapt the supervised model to the multi-step forecasting task, we use a recursive approach [27, 8].

3.4.2 Deep Learning Models

We also seek to develop Deep Learning models, such as Long Short-Term Memory (LSTM) networks and DeepAR models, which allow us to easily cover the above requirements 3.4 with as little manual work as possible.

3.4.2.1 DeepAR - Probabilistic Forecasting

To implement the DeepAR models we use the *GluonTS* library, built above *MXNet*, which contains a DeepAR Estimator ready to use [1]. This model corresponds to an encoder-decoder structure of *LSTM* cells allied to probabilistic cells that allow selecting a likelihood function that best fits the properties of the data. As such, we tested 2 different likelihoods: *Negative Binomial* and *Poisson*.

The DeepAR Estimator of this library still has a considerable number of Hyper-Parameters to adjust network structure and the way it trains with the data, some of them being:

- **num_layers**
- **num_cells**
- **dropout_rate**
- **epochs**
- **num_batches_per_epoch**
- **batch_size**
- **learning_rate**

To train the models with our datasets, we used the default values suggested by the library for these parameters, except for the number of *epochs* which was reduced to 20.

After defining how the model will be trained, it is possible to feed it with all the time-series of the dataset, with the respective static and dynamic variables, both numerical and categorical. The model is capable of dealing with the different magnitudes of time-series, using static variables to identify groups of time-series with similar sales behavior, and also performing automatic feature engineering from the features it received as input. As a result, we have a global fitted model capable of predicting sales for different time-series, including series with little historical data.

3.4.3 Classical Models - Prophet

Prophets are statistical models developed by Facebook that have been gaining popularity for their ability to adjust to the behavior of time series and capture dramatic shifts in their trends [15]. These models have the best performance in time series with strong seasonality (daily, weekly, monthly, or yearly) and trends, and which contain historical data from several seasons. Thus, they seem to be candidates to obtain good predictions in the time series of the M5 competition.

Unlike the global models in 3.4.2 and 3.4.1, Prophet works like the classic models and its parameters need to be fitted for each time-series individually, creating a model for each time-series. As such, Prophet models cannot include static variables, that is, variables that do not vary within a time-series. In order to tune the model's Hyper-Parameters, this must also be done for each time-series individually. For each cutoff, the model is fitted using only data up to that cutoff, and the predictions are made for data from that cutoff up to $cutoff + horizon$. Thus, we obtain for each time-series as many results as cutoff points used in the validation. These results correspond to predictions in different horizons of the time-series, which gives us a more robust evaluation of the *Prophet* model in a given time-series. In the datasets of the M5 competition, we used as cutoff points the *Limit* dates 2016-01-31, 2016-02-28, 2016-03-27, 2016-04-24, which give us 4 results in the validation of a model in each time-series.

The Prophet library already includes a method for doing **cross-validation** of models in time-series. In this method we can select a number of cutoff points along the time series history. Throughout this work, when we talk about validation, we will also refer to these cuttof points as *Limit* dates, and they correspond to dates that divide the time-series into training data (past timepoints) and test data (future timepoints to be predicted).

We also **tune the Hyper-Parameters** by experimenting with different combinations of the *changepoint_prior_scale* and *seasonality_prior_scale* parameter values. The parameters *weekly_seasonality* and *yearly_seasonality* are set to True to train the time series models of the M5 datasets.

```
1 param_grid = {
2     'changepoint_prior_scale': [0.001, 0.01, 0.1, 0.5],
3     'seasonality_prior_scale': [0.01, 0.1, 1.0, 10.0],
```

```
}

param_set = {
    'yearly_seasonality': True,
    'weekly_seasonality': True,
}
```

Given the possible combinations of the *changepoint_prior_scale* and *seasonality_prior_scale* parameters values, we get 16 different combinations of Hyper-Parameters to experiment fitting a model in each time-series. For each combination of Hyper-Parameters in each time-series, we perform cross-validation using the 4 cutoff points mentioned above, in order to obtain a robust evaluation of the model trained with this set of Hyper-Parameters to predict that time-series. After obtaining the evaluation for all sets of Hyper-Parameters in a single time-series, calculated using the RMSE metric, the set of **best Hyper-Parameters** is selected to fit a final model using the last cutoff of the given time-series, thus obtaining a final evaluation of its predictions.

Thus, to tune and fit the model in a single time-series, 64 workouts are performed before the final results are obtained. This number of iterations increases even more if we want to use Prophet models to predict more than one time series. As such, we have to be careful about the amount of time series to be predicted using this model, since the amount of time required is very high. This is already a **limitation of classical models** in relation to global models, which only need to fit and tune one model using all time-series and to predict several time-series, which is quite useful in the retail market, where the number of time-series of different products in different stores can reach thousands or even millions.

Prophets were developed to compare the performance of global models to these more **classic and prevalent** state-of-art approaches where each model is **tuned and fitted to a specific time-series** to capture its behavior.

3.5 Evaluation

After preparing the data and developing the forecasting models, it becomes necessary to evaluate the models' predictions in order to compare and analyze their performance.

In common Machine Learning problems, the **dataset is divided into training and test sets**. The division is done so that the proportion of training data is greater than the proportion of test data: 80%/20% or 70%/30% are quite common divisions. Training data is fed to the model to fit its parameters. Then, the fitted model is used to predict the unknown target values corresponding to the test sets. To evaluate the model, the predicted results are compared to the actual values of the test set, using an **evaluation metric carefully chosen**.

An important step to make a more robust evaluation of models is to do their **Validation**. One of the most accepted validation techniques in the area of Machine Learning is the *k-folds Cross-Validation*, where the data is divided into k chunks known as folds, randomly chosen and with equal length. Then the model is tested in each of the k folds using the remaining $k-1$ folds as training set to fit its parameters. Thus, k results are obtained from the application of the model in different training and test sets, as can be seen in the figure 3.17. The averaging of the k results offers a robust evaluation of the model's performance, allowing to adjust its hyper-parameters with less risk of overfitting.

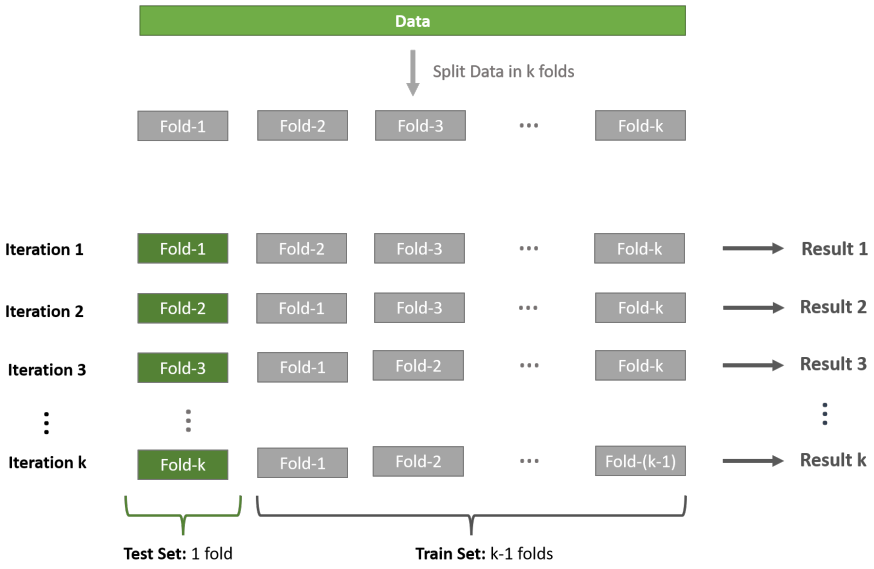


Figure 3.17: Example diagram of applying k-fold Cross-Validation

However, this type of validation cannot be used in the evaluation of forecasting models. In forecasting problems, time-series observations are highly time-dependent and the **temporal order of historical data must be conserved** in the training and test sets. It makes no sense to use future data as training data to help predicting a test set of past data. Thus, the *k-fold Cross-Validation* must be adapted to ensure that the temporal order of historical data is maintained, so that past data is used as training data to fit the models, and future data is used as test data to be predicted by the fitted models. This adaptation results in a validation for forecasting models also known as **Walk-Forward Validation**.

3.5.1 Walk-Forward Validation

Over time, the retailer obtains new sales data, while it becomes necessary to forecast new horizons of sales values into the future. Thus, a forecasting model must be able to include the new data in the training set to readjust its parameters and then predict the new prediction horizon while maintaining the consistency of the predicted results. Walk-forward Validation allows to simulate

and assess the **model's ability to maintain consistent results** in its predictions **over time** by following these steps:

1. Choosing a horizon length of h time-points we want the model to be able to predict. Choosing the number of different horizons, N, we want the model to predict before assessing its performance over time ($N = 4$ in 3.18).

2. Setting an initial subset of data to train the models, which excludes the last N horizons of h time-points of the time-series (*Iteration 1* in 3.18). The initial train set is used to train the forecasting model. Then, the fitted model is used to predict the later h data points of the data, corresponding to the test set. The predicted values for the horizon are compared to the true values of the test set, obtaining the result of the first iteration of the model (*Result 1* in 3.18).

3. The known values for the predicted horizon (test data) is added to the next iteration's train data. The new train data is used to fit the model, and the later h time-points are the new test set to be predicted. This process is repeated until N predictions with a length of h time-points obtained (*Iteration 2*, *Iteration 3*, and *Iteration 4* in 3.18).

4. Each iteration generates predictions that allow us to calculate a result. Thus, N iterations generate N results (*Results 1*, *Results 2*, *Results 3*, and *Results 4* in 3.18), which allow assessing whether the model remains consistent over time, while new data are known and new horizons are predicted.

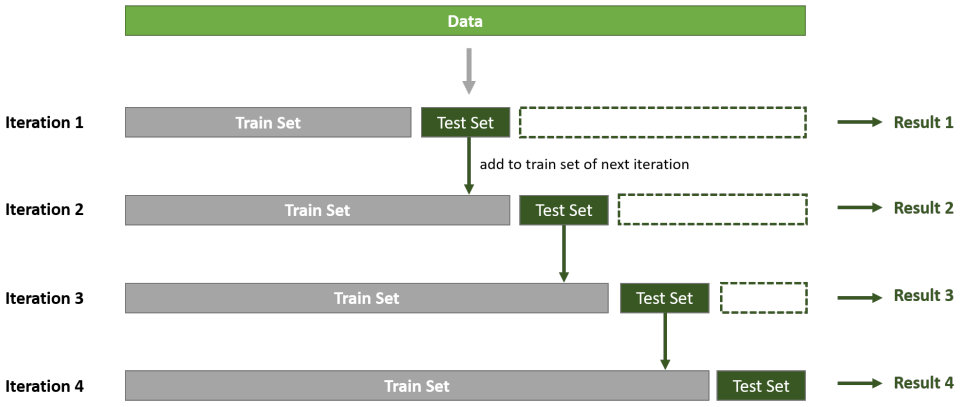


Figure 3.18: Example of applying the Walk-Forward Validation to evaluate the model in 4 consecutive horizons ($N = 4$)

This validation was implemented and used to evaluate all forecasting models developed in the study. The only exception was the Prophet model, for which we used the already implemented cross-validation suggested in the Prophet library [15], and described in section 3.4.3. For both validations, we consider $N = 4$, to obtain an evaluation of the models in the forecast of 4 different horizons over time. For the M5 datasets, we used a horizon of 28 days, $h = 28$, as suggested by the competition. For the synthetic dataset we used a horizon of 1 week, $h = 1$.

Throughout the thesis, we will refer to the N successive dates used to split the time-series into training and test data in each iteration (1 to N) as *Limit* dates. To train and predict in the M5 competition datasets, the *Limit* dates used for each of the $N = 4$ validation iterations are:

- **Iteration 1:** 2016-01-31
- **Iteration 2:** 2016-02-28
- **Iteration 3:** 2016-03-27
- **Iteration 4:** 2016-04-24

3.5.2 Metrics

To evaluate the model's predictions at each iteration, we need to calculate a metric that measures the distance of the prediction values from the actual demand values. Bearing in mind the analysis in the Section 2.3, the following metrics were calculated and taken into account with due precautions.

- **MASE:** It is the measure that best suits the evaluation of our models in the chosen datasets - mainly because of the M5 competition datasets. It does not depend on the magnitudes of the different time-series and can be used when there are intermittent time-series, that is, with periods where the value of sales is equal to 0. As such, it is the most used metric in this study to analyze and compare the performance of different models.

- **MAPE:** It has the advantage of being independent of the magnitude of the time-series, giving a fair assessment of the performance of the models in the different time-series. However, this metric becomes unreliable when the time series show strong intermittence. For this reason, this metric is unreliable at the levels of aggregation 10 and 11 of the M5 dataset, where the time-series show more frequently intermittent sales. As such, we additionally use this metric in the evaluation of models in datasets whose time-series rarely show intermittence and, as such, where it is possible to obtain a considerable number of *MAPE* values from the forecasts of different time series - this is the case of the aggregation level 9 of the M5 competition.

- **RMSE:** Derived from *MSE*, this measure gives more penalization to larger errors, while being measured in the same units as the target variable. For this reason, both this metric and the *MSE* measure are more commonly used for training and fitting model parameters.

- **MAE:** This metric was calculated for its easy interpretation and for a first analysis during the development of the models. However, due to its simplicity and dependence on the magnitudes of the time-series, it was not used to analyze the performance of the models.

Chapter 4

Tests and Results

As the time-series of the Synthetic Dataset were considerably short, the results ended up not being representative of the performance of the models and, therefore, did not add scientific value to the thesis. For this reason, we will only consider the results of the models in the datasets corresponding to the different levels of aggregation - levels 9, 10, and 11 - of the M5 competition.

The developed models are used to forecast sales of different time-series corresponding to sales of different retail products. The magnitudes of sales over the time-series vary from product to product. We divided the different time-series and the results obtained by each model into 4 groups, depending on the magnitude of the sales:

- **MIN-Q1:** time-series of products (or product departments) with low to medium-low sales values.
- **Q1-Q2:** time-series of products (or product departments) with medium-low to medium values.
- **Q2-Q3:** time-series of products (or product departments) with medium to medium-high values.
- **Q3-MAX:** time-series of products (or product departments) with medium-high to high values.

To make this division, we calculate the median value of sales for each time-series of the dataset. Then, we calculate statistics with information on the distribution of the median values of all time-series in the dataset, namely the minimum, the quantiles 25%, 50%, and 75%, as well as the maximum. These values were used to divide the different time series into the groups described above: *MIN-Q1*, *Q1-Q2*, *Q2-Q3*, and *Q3-MAX*. Thus, we can evaluate how each model performs in time-series with different sales magnitudes.

The development of the models mentioned above in 3.4 allowed the:

- Analysis of **Deep Learning** models, such as *DeepAR* and *LSTM*, which are capable of doing automatic feature engineering with minimal manual work.

- Analysis of **Decision Trees**, such as *Random Forest* and *Gradient Boost*, which are Interpretable Models as they allow to see the importance of the different features used as input.
- Analysis of a **Statistical Model**, in this case the *Prophet* model, created by Facebook and which has been gaining popularity for time-series forecasting purposes.
- Analysis and comparison of the performance of **Global Models** with **Classic Models**. As Global Models, we refer to models capable of including static variables and learning a global model that can be used to forecast sales from different time series. These include *Deep Learning* models and an adaptation of *Decision Tree* models. By Classical Models, we refer to methods where the model parameters need to be fitted to each time-series individually, creating a model for each time series. These models, such as *Prophet*, cannot include static variables, that is, variables that do not vary within a time series. In order to tune the model's Hyper-Parameters, this must also be done for each time series individually.
- Analyze models that predict **Point Forecasts** and models capable of doing **Probabilistic Forecasts**, which include the uncertainty range of predictions. *DeepAR* models and *Prophet* models are capable of performing Probabilistic Forecasting, while the remaining models return Point Forecasts.

The application of models at different levels of aggregation of the M5 dataset also allows analyzing the performance of different models at different levels of aggregation of sales:

- **Level 9:** aggregates the sales of all products of each Department in each Store. This aggregation results in time series with **much higher magnitudes** and **more regular behavior** than levels 10 and 11. Therefore, these time-series are expected to be easier to predict than the ones of levels 10 and 11.
- **Level 10:** time-series with aggregated sales of each product in all stores and states. The magnitudes of these time series tend to be much smaller than at level 9. Sales behavior tends to be much **more volatile and difficult to predict** than at level 9.
- **Level 11:** time-series with aggregated sales of each product by State. The magnitudes of these time-series tend to be even smaller than at level 10. Thus, these time series also show a much **more volatile and difficult to predict** sales behavior.

At levels 10 and 11 of aggregation we exclude time-series whose median sales value was less than 3. These are time-series in which half of the values are less than 3 - less than or equal to 0, 1, or 2 - which makes it unnecessary to use forecasting models to predict such time-series. This filtering was not necessary at level 9, due to the much higher magnitude of its time series.

To calculate statistics on forecast errors of a model predicting the time-series of each group - *MIN-Q1*, *Q1-Q2*, *Q2-Q3* and *Q3-MAX* - we consider its results in all validation Iterations (N=4

in total). Thus, for each time-series we have 4 results corresponding to the model's predictions in 4 different horizons, one for each Iteration. Each Iteration is identified by the *Limit* date that divides the training and test data from the time-series: 2016-01-31, 2016-02-28, 2016-03-27, and 2016-04-24. The only exception occurs in the *Prophet* model, where for each time-series, we only include the results of predictions in the last horizon (last *Limit* date) after the hyper-parameters are tuned and the parameters fitted for that specific time series, as described in 3.4.3.

4.1 Global Models

4.1.1 Decision Trees - Random Forest and Gradient Boost - Point Forecasts

As mentioned earlier, these models allow us to verify which features are most used and, therefore, more important for forecasting the sales of the time-series. Furthermore, it is possible to train these models on all time-series of the dataset, leveraging static variables with information about products and stores' locations. To carry out some tests and get results when necessary, we select a smaller sample of time-series from each of the 3 levels of aggregation datasets of the M5 competition, to train the model and obtain results on the predictions of future values of these time-series:

- **Level 9:** all *70 time-series* of this level of aggregation.
- **Level 10:** *60 time-series* with information on demand for 20 products of each category.
- **Level 11:** *90 time-series* with information on demand for 10 products of each category in all 3 states.

In these models, the categorical features were encoded using label-encoding, unlike what was done in the other models, where the categorical features were encoded as dummy variables.

4.1.1.1 M5 - Level 09 Aggregation

The figures 4.1 and 4.2 show the importance of the features fed to the *Random Forest* and *Gradient Boost* models.

The *Random Forest* model assigns higher importance to the lag features *t-28*, *t-7* and *t-1*. In these 3 features, the importance is greater, the greater the distance from the lag to the value to be predicted. The importance given to the remaining features, other than the lags, is much lower, practically irrelevant 4.1.

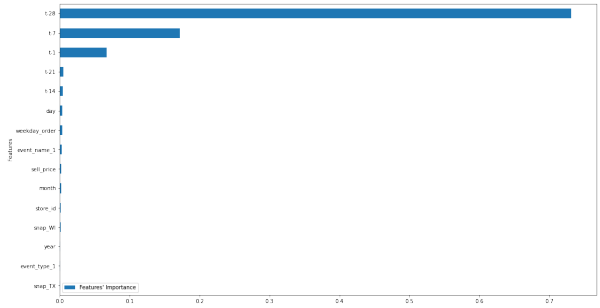
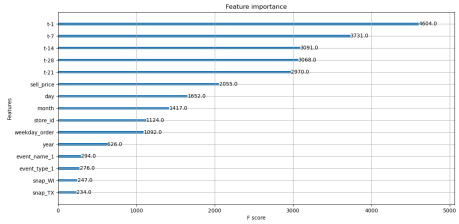
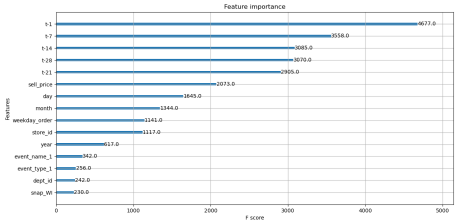


Figure 4.1: **M5 Level 09 - Random Forest** - Feature Importances - the order and importance of features did not change in the different Iterations (*Limit* dates).

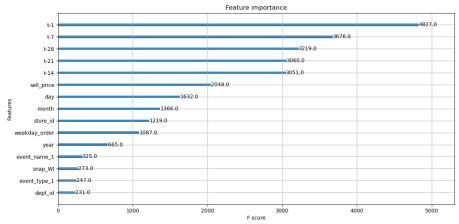
The *Gradient Boost* models give greater importance to all lag features: *t-1*, *t-7*, *t-14*, and *t-21*, and *t-28*. The highest importance is given to the most recent sales values, closest to the value to be predicted, *t-1*, *t-7*, and *t-14*, followed by the values *t-28* and *t-21*, in descending order of importance 4.2. In the different iterations 4.2a, 4.2b, 4.2c, and 4.2d, we can see that these models still use product and sales location features, such as *sell_price*, *store_id*, and *dept_id*, as well as some features on date, events, and snap variables.



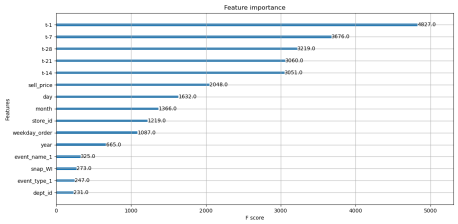
(a) Limit 2016-01-31



(b) Limit 2016-02-28



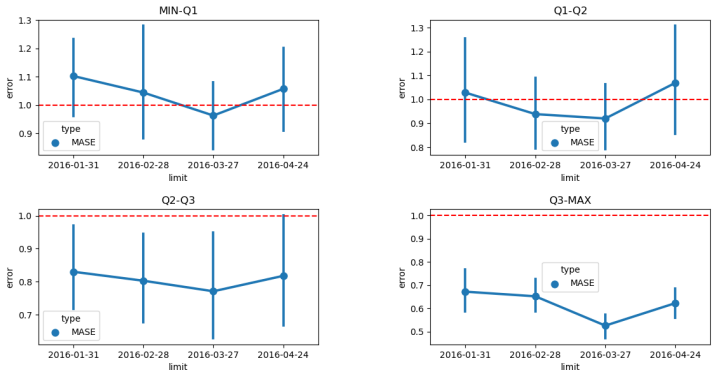
(c) Limit 2016-03-27



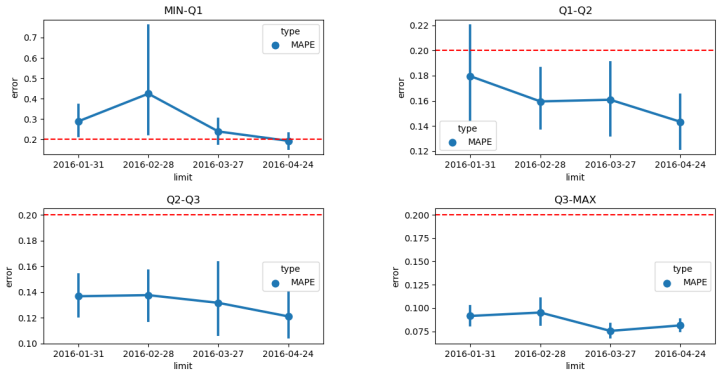
(d) Limit 2016-04-24

Figure 4.2: **M5 Level 09 - Gradient Boost** - Feature Importances.

The *MASE* and *MAPE* error plots for the level 9 of aggregation show that the *Random Forest* model obtains better results the greater the sales magnitudes of the time-series 4.3. This can be seen as the errors get smaller and smaller for groups *Q1-Q2*, *Q2-Q3* and *Q3-MAX*, in that order. In the same way, the plots of *MASE* and *MAPE* errors in 4.4 show the same tendency of the *Gradient Boost* models to obtain better results in the time-series of groups with higher magnitudes of sales.



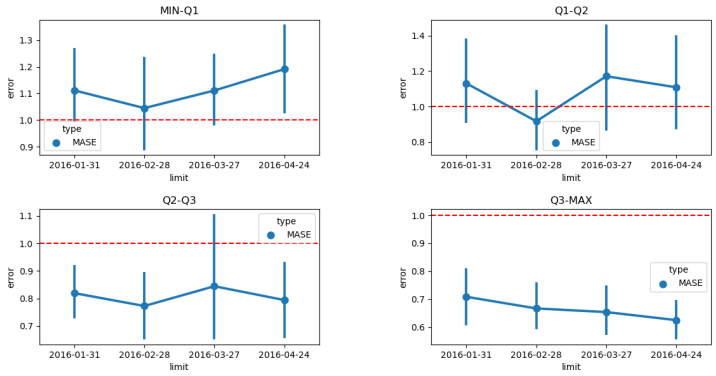
(a) MASE errors.



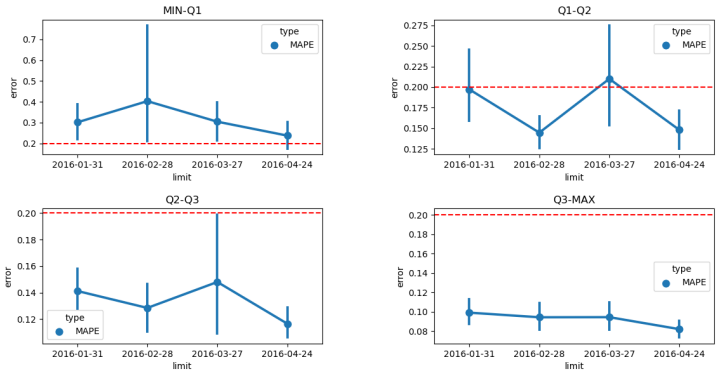
(b) MAPE errors.

Figure 4.3: **M5 Level 09 - Random Forest** - Expected value and confidence interval of the MASE and MAPE errors over training sets with increasing sizes (increases in *Limit* date).

According to Table B.1, *Random Forest* models seem to perform better than the *Gradient Boost* models at this level of aggregation, resulting in the lowest statistical values for the MASE error in all groups of time-series. However, according to the *MAPE* errors, both the *Random Forest* and *Gradient Boost* models obtain very good predictions for the time-series of the different magnitude groups. The statistics confirm that both models obtain better results in time-series of the groups of higher sales magnitudes.



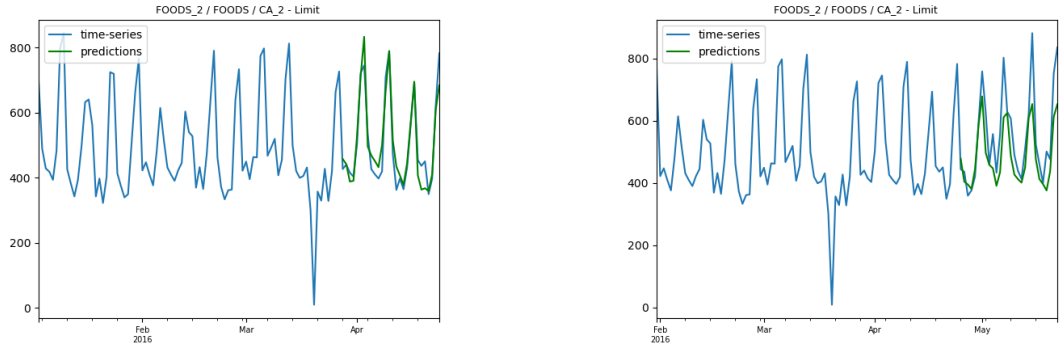
(a) MASE errors.



(b) MAPE errors.

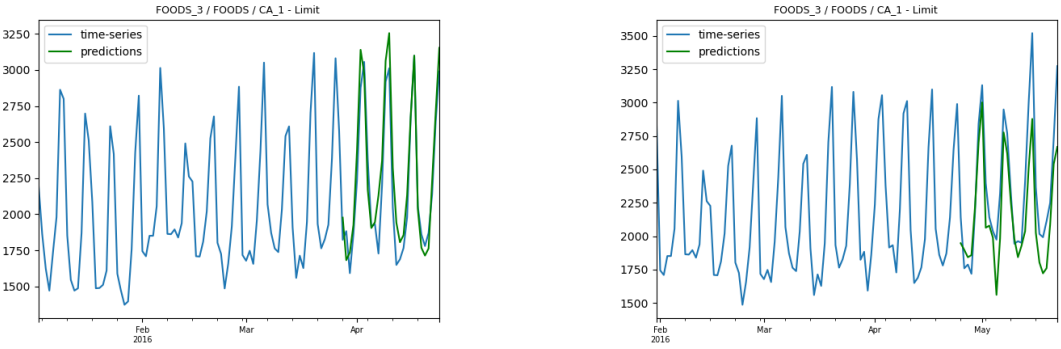
Figure 4.4: **M5 Level 09 - Gradient Boost** - Expected value and confidence interval of the MASE and MAPE errors over training sets with increasing sizes (increases in *Limit* date).

The *Gradient Boost* models easily adjust to the behavior of the time-series, quickly capturing seasonality or trends in them in the different magnitudes of sales at this level of aggregation (4.5, 4.6). This ability is also verified in highly volatile time-series, where sales vary rapidly 4.7. The same is true for *Random Forest* models at this level of aggregation (B.1, B.2, and B.3).



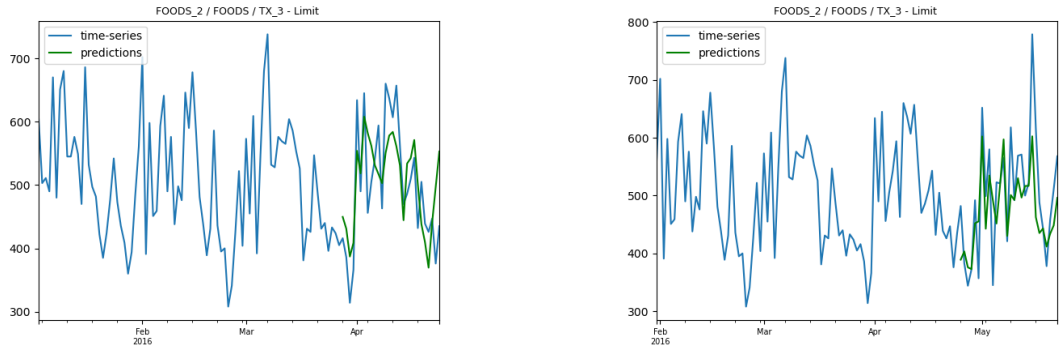
(a) Limit 2016-03-27 (b) Limit 2016-04-24

Figure 4.5: **M5 Level 09 - Gradient Boost** - MIN-Q1 forecastig example.



(a) Limit 2016-03-27 (b) Limit 2016-04-24

Figure 4.6: **M5 Level 09 - Gradient Boost** - Q3-MAX forecastig example.

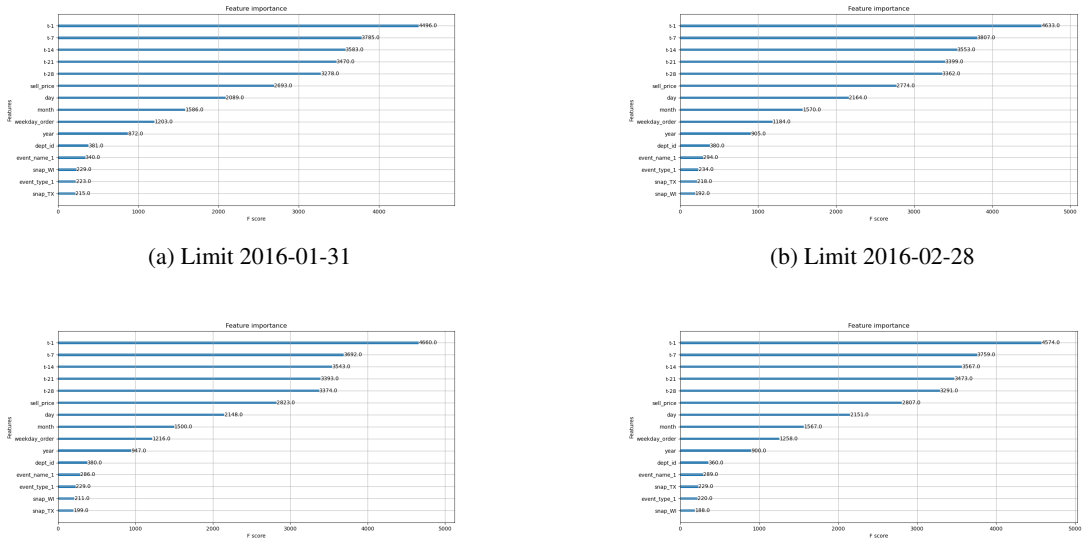


(a) Limit 2016-03-27 (b) Limit 2016-04-24

Figure 4.7: **M5 Level 09 - Gradient Boost** - forecastig volatile sales.

4.1.1.2 M5 - Level 10 Aggregation

The relative importance of the features fed to the *Gradient Boost* model in the different iterations of walk-forward validation 4.8 is similar to those observed at aggregation level 9.



(a) Limit 2016-01-31

(b) Limit 2016-02-28

(c) Limit 2016-03-27

(d) Limit 2016-04-24

Figure 4.8: **M5 Level 10 - Gradient Boost** - Feature Importances.

At this level of aggregation, *Random Forest* models continue to predominantly use lag features to predict future sales values. The only difference is that at this level the *Random Forest* seems to give a little more importance to the prices of the products sold (*sell_price*) to predict future sales 4.9.

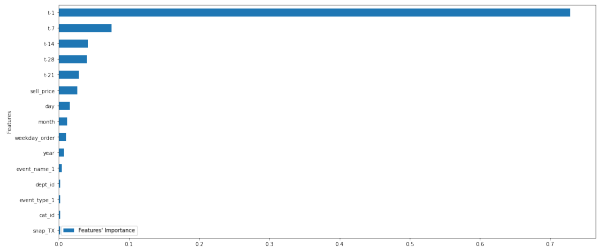
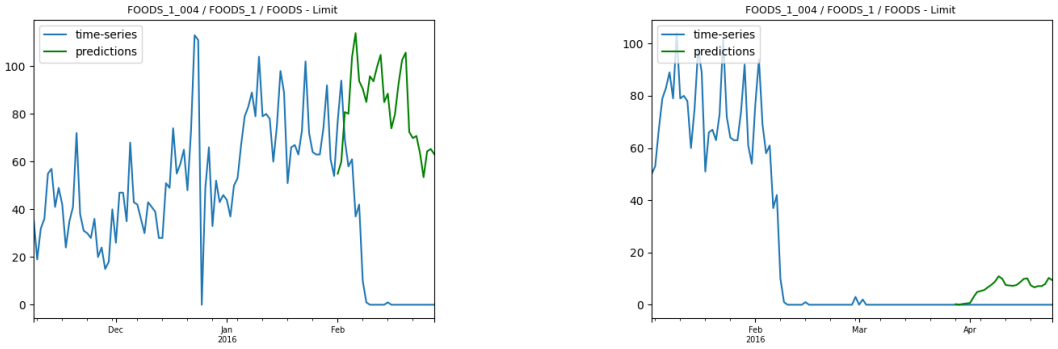


Figure 4.9: **M5 Level 10 - Random Forest** - Feature Importances.

Table B.2 shows that the *MASE* and *MAPE* errors obtained by both *Gradient Boost* and *Random Forest* models are higher at this level of aggregation than at level 9. *MAPE* values worsen in practically all groups, while *MASE* values deteriorate in groups *Q2-Q3* and *Q3-MAX*. This would be expected, as sales of time-series for individual products have much lower magnitudes, and the behavior of those time-series becomes more irregular and difficult to predict. There are also very high MAPE errors, or even tending to infinity. This is a consequence of the presence of sales equal

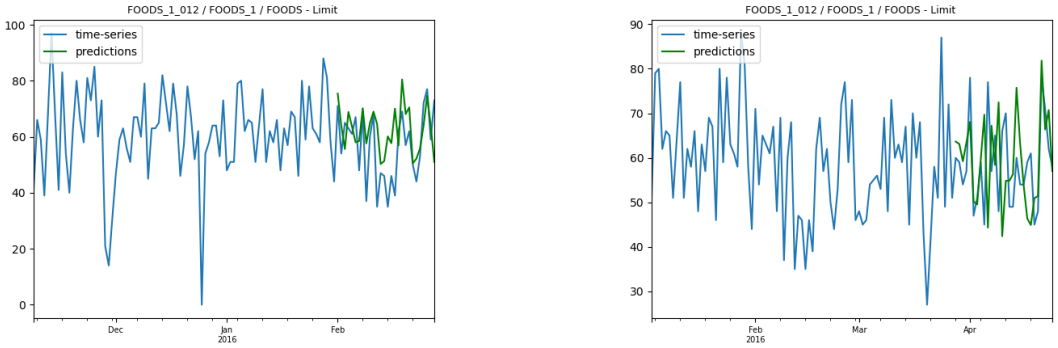
to 0 in the time-series of this level of aggregation, where the **intermittence becomes evident and quite frequent** 4.10.

Both *Random Forest* and *Gradient Boost* models have more difficulty modeling highly volatile time-series behaviors at this level of aggregation (4.11, 4.12). However, they are still able to detect some weekly seasonality and follow some sales variations, when these are not too sudden.



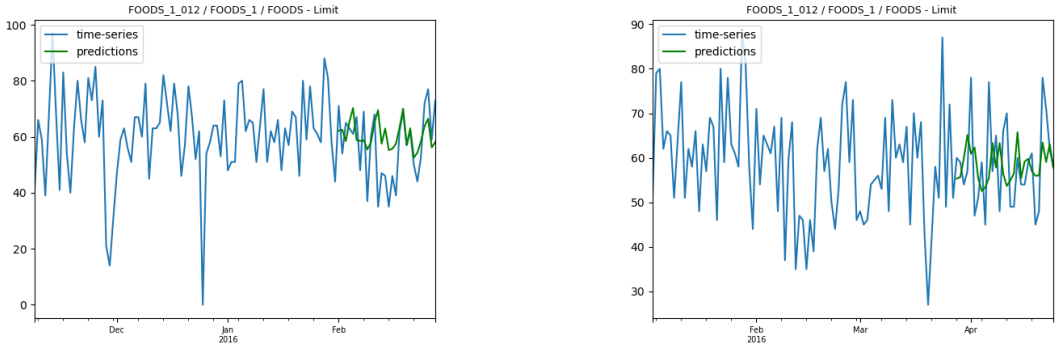
(a) Limit 2016-01-31 (b) Limit 2016-03-27

Figure 4.10: **M5 Level 10 - Gradient Boost** - Intermittent time-series example example.



(a) Limit 2016-01-31 (b) Limit 2016-04-24

Figure 4.11: **M5 Level 10 - Gradient Boost** - Q3-MAX forecasting example.



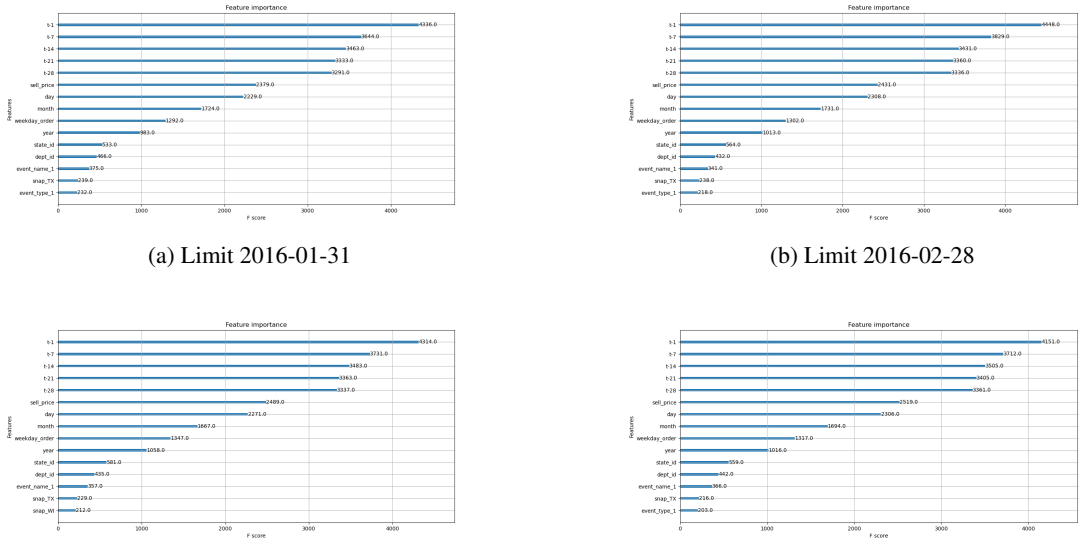
(a) Limit 2016-01-31

(b) Limit 2016-04-24

Figure 4.12: **M5 Level 10 - Random Forest** - Q3-MAX forecasting example.

4.1.1.3 M5 - Level 11 Aggregation

At this level of aggregation, the *Gradient Boost* continues to give greater importance to lag features, the importance being greater the closer they are to the value to be predicted - *t-1*, *t-7*, *t-14*, *t-21*, and *t -28*, in descending order of importance 4.13. This model also uses features on the date, events, products' prices, and location of sales, such as *dept_id* and *state_id* in the different validation iterations (4.13a, 4.13b, 4.13c, 4.13d).



(a) Limit 2016-01-31

(b) Limit 2016-02-28

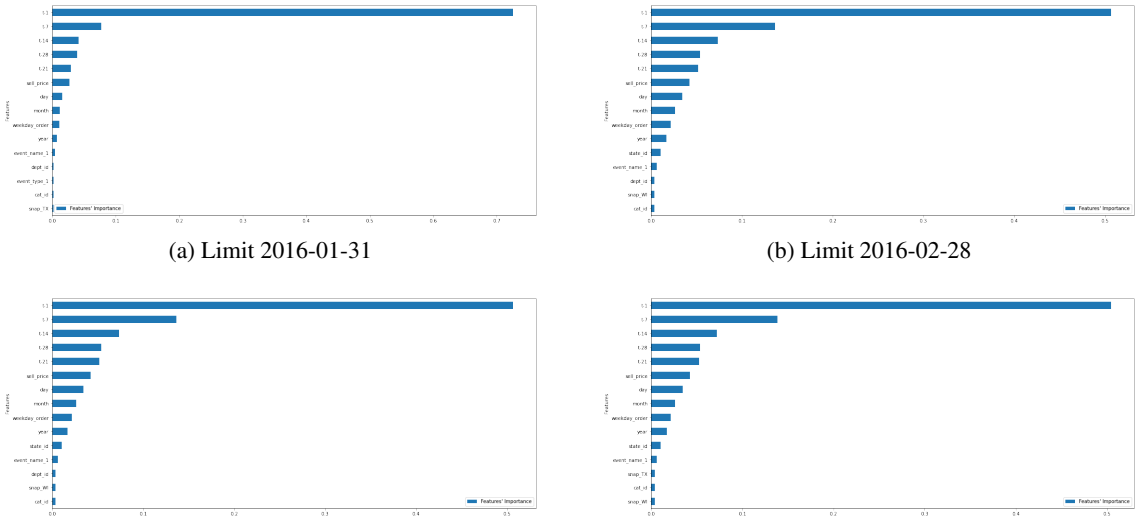
(c) Limit 2016-03-27

(d) Limit 2016-04-24

Figure 4.13: **M5 Level 11 - Gradient Boost** - Feature Importances.

In Random Forest, the most important feature corresponds to lag *t-1*, followed by lags *t-7*, *t-14*, *t-28*, and *t-21* 4.14. At this level of aggregation, these models seem to give a little more importance

to product prices, as well as static variables on products' categories, *cat_id*, and sales location, such as *dept_id* and *state_id*, unlike what happened at level 9. This may be an indication that when time-series sales become more volatile and irregular and, therefore, more difficult to predict, it becomes more important for the model to use other external variables, including static variables with information on products and sales location, to help predict future sales for these time-series, in addition to their previous sales. In these cases, the previous sales of time-series, also referred to as lags, are not enough to obtain good forecasts.



(a) Limit 2016-01-31

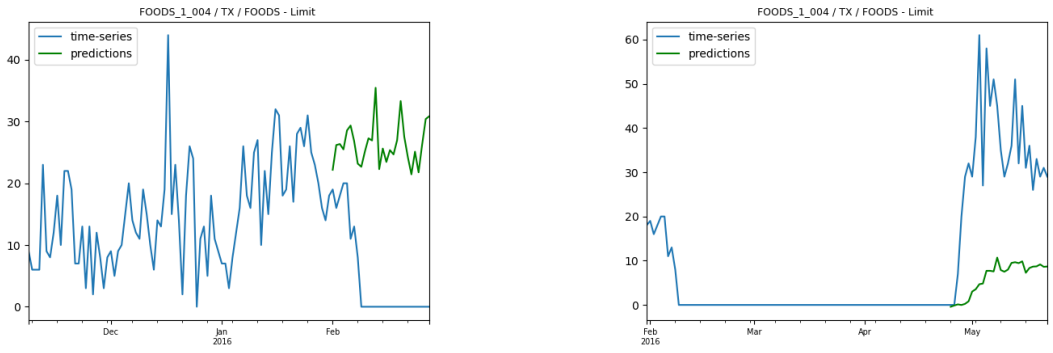
(b) Limit 2016-02-28

(c) Limit 2016-03-27

(d) Limit 2016-04-24

Figure 4.14: **M5 Level 11 - Random Forest** - Feature Importances.

As can be seen in Table B.3, the *MAPE* errors become maladaptive at this level of aggregation, as happened at level 10, being mostly very high values or tending to infinity as a consequence of the strong existence of intermittent sales in the time-series of these levels of aggregation 4.15.



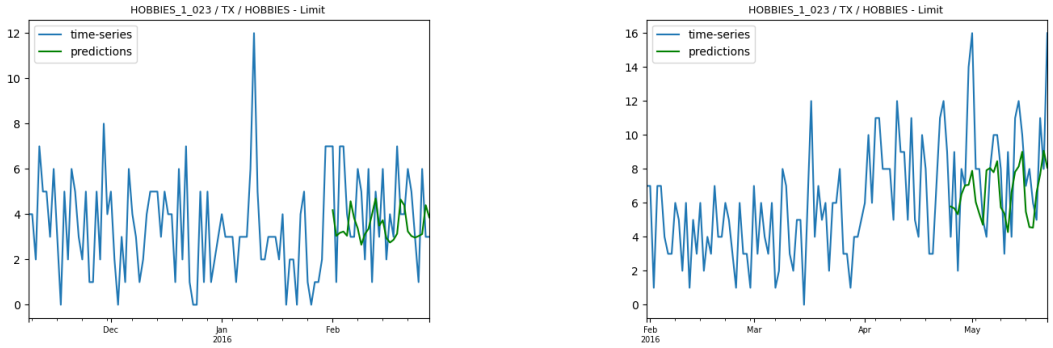
(a) Limit 2016-01-31

(b) Limit 2016-04-24

Figure 4.15: **M5 Level 11 - Gradient Boost** - Intermittent time-series example.

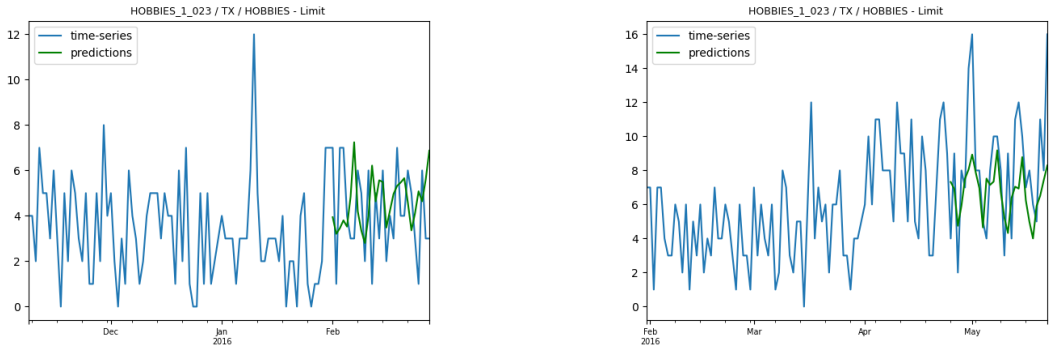
The forecast examples in Figures 4.16, 4.17, 4.18, 4.19, 4.20, and 4.21, show that although

sales become more volatile, unstable, and difficult to predict, the Random Forest and Gradient Boost models continue to capture some of these variations and trends in sales, albeit with less success. compared to level 9 results.



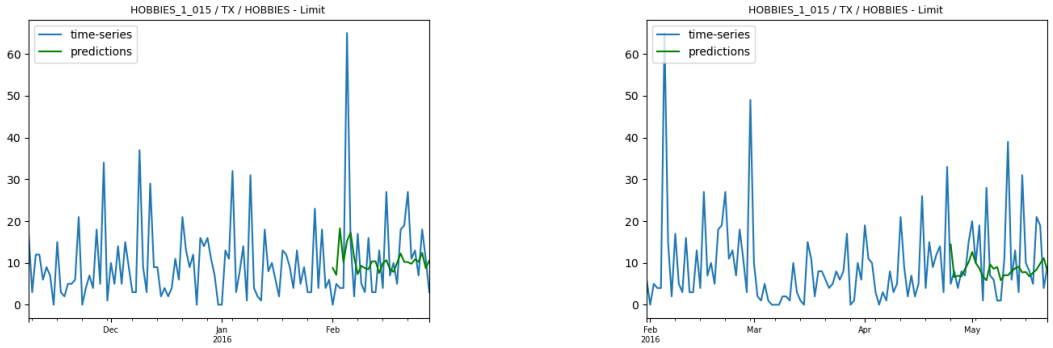
(a) Limit 2016-01-31 (b) Limit 2016-04-24

Figure 4.16: **M5 Level 11 - Gradient Boost** - Q1-Q2 time-series example.



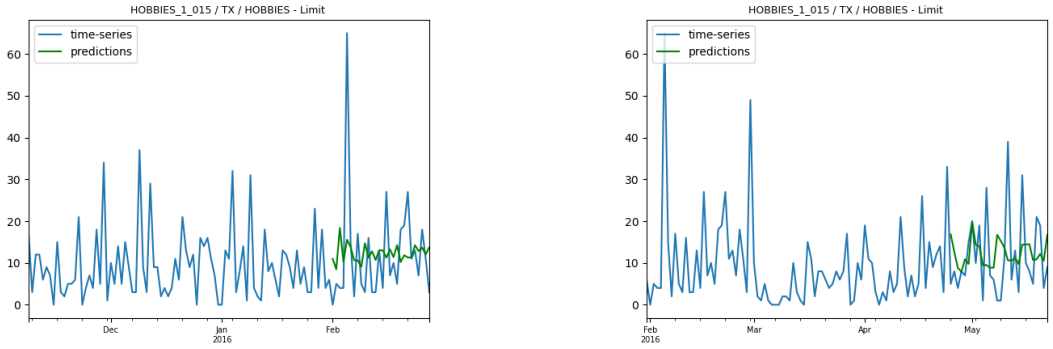
(a) Limit 2016-01-31 (b) Limit 2016-04-24

Figure 4.17: **M5 Level 11 - Random Forest** - Q1-Q2 time-series example.



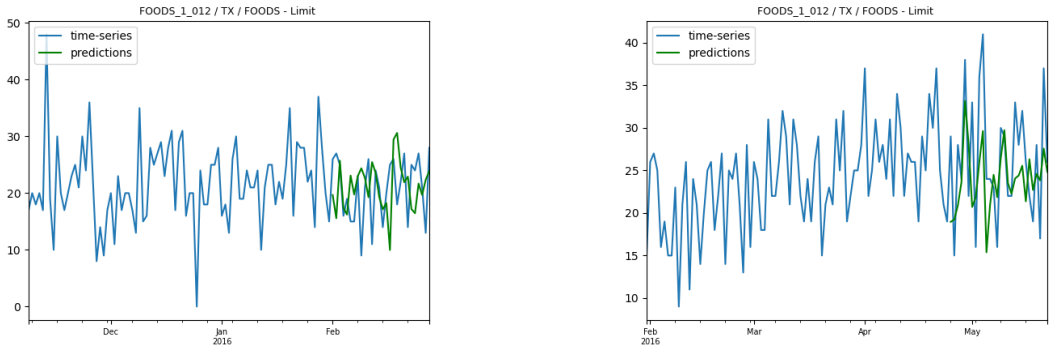
(a) Limit 2016-01-31 (b) Limit 2016-04-24

Figure 4.18: **M5 Level 11 - Gradient Boost** - Q2-Q3 time-series example.



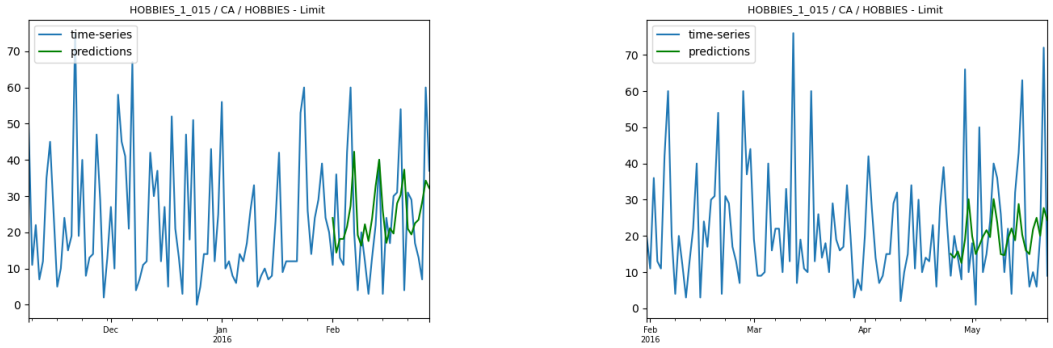
(a) Limit 2016-01-31 (b) Limit 2016-04-24

Figure 4.19: **M5 Level 11 - Random Forest** - Q2-Q3 time-series example.



(a) Limit 2016-01-31 (b) Limit 2016-04-24

Figure 4.20: **M5 Level 11 - Gradient Boost** - Q3-MAX time-series example.



(a) Limit 2016-01-31

(b) Limit 2016-04-24

Figure 4.21: **M5 Level 11 - Random Forest** - Q3-MAX time-series example.

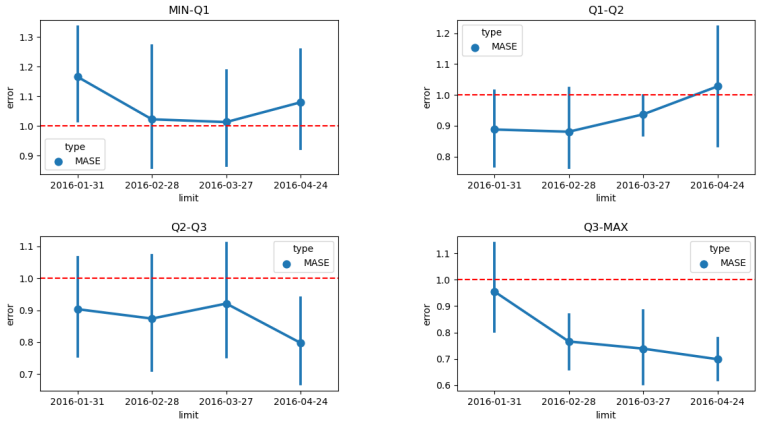
4.1.2 Deep Learning - DeepAR - Probabilistic Forecasts

DeepAR models were tested using 2 different types of distribution to make probabilistic sales forecasts: *Negative Binomial* and *Poisson*. Since these models are faster in training with multiple time-series, it is possible to use much larger samples of time-series. To get started, we selected the following samples for each level of aggregation of the M5 Competition:

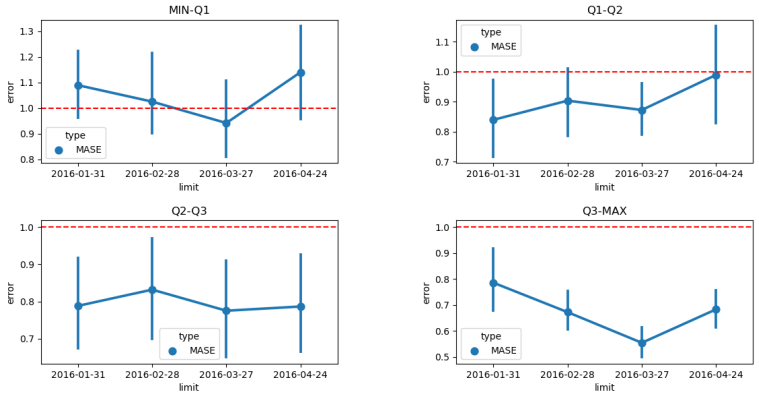
- **Level 9:** all *70 time-series* of this level of aggregation.
- **Level 10:** *600 time-series* with information on demand for 200 products of each category.
- **Level 11:** *900 time-series* with information on demand for 100 products of each category in all 3 states.

4.1.2.1 M5 - Level 09 Aggregation

For this level of aggregation, the model was trained with historical information from 70 time-series. Then, the fitted model was used to predict future sales for all 70 time-series. The Figures 4.22a and 4.22b show the evolution of the *MASE* errors obtained by the DeepAR models using the Negative Binomial and Poisson distributions, respectively, for the time-series of the different magnitude groups (*MIN-Q1*, *Q1-Q2 Q2-Q3*, *Q3-MAX*).



(a) Using the Negative Binomial Distribution.

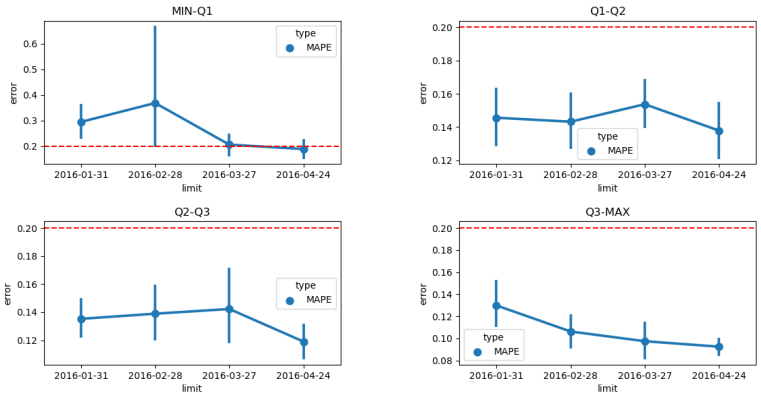


(b) Using the Poisson Distribution.

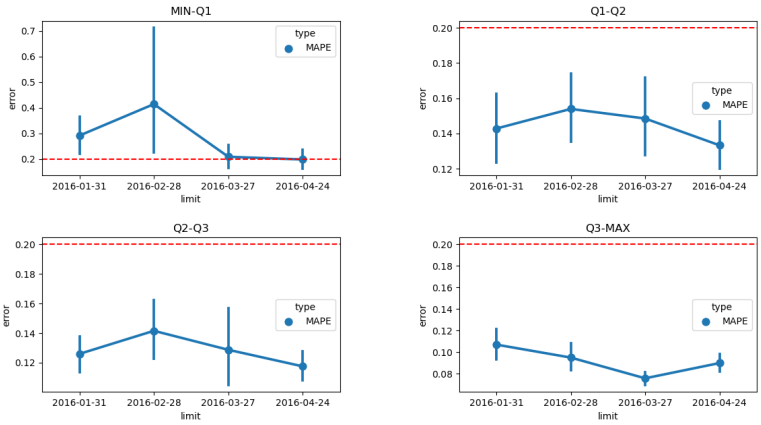
Figure 4.22: M5 Level 09 - DeepAR - Expected value and confidence interval of the MASE errors over training sets with increasing sizes (limit).

Figures 4.23a and 4.23b show the evolution of the *MAPE* errors obtained by the DeepAR models using the Negative Binomial and Poisson distributions, respectively, in a similar way the figures 4.22a and 4.22b show the evolution of the *MASE* error.

According to the 4.22 plots, the DeepAR model seems to get smaller MASE errors the greater the sales magnitudes of the time-series when using both the Negative Binomial and Poisson distributions. The same trend can be seen for MAPE errors in figures 4.23. The models tend to obtain the highest *MAPE* and *MASE* errors for the time-series with the smallest sales magnitudes, MIN-Q1. These errors are successively smaller in groups *Q1-Q2*, *Q2-Q3*, and, finally, *Q3-MAX*, where the best results are obtained. Furthermore, the learning capacity of the models with the increase in the amount of training data is more evident in time-series with high magnitudes of sales, *Q3-MAX*, where the results' errors decrease rapidly with the increase of the *limit* date that divides the training data from the test data of the time-series.



(a) Using the Negative Binomial Distribution.



(b) Using the Poisson Distribution.

Figure 4.23: M5 Level 09 - DeepAR - Expected value and confidence interval of the MAPE errors over training sets with increasing sizes (limit).

Table B.4 includes statistics of forecast results of all time-series of the different groups of magnitude - *MIN-Q1*, *Q1-Q2*, *Q2-Q3*, and *Q3-MAX* - aggregated for all iterations of the Walk-Forward Validation (described in 3.18). Each iteration is identified by a *Limit* date, which separates the training and test data from the time series, and which increases from iteration to iteration, resulting in a new horizon to be predicted for all the time series.

In summary, we calculated the 25%, 50%, and 75% quantiles of the *MASE*, *MAPE*, *MAE*, and *RMSE* errors in the different groups of time-series to get a sense of the distribution of those errors. We also calculate the mean and the standard deviation of the different errors obtained in the different groups of time-series to have a general assessment of the model's performance and how stable this assessment is.

The DeepAR model that uses a Poisson distribution obtained the lowest statistical values of all

errors in almost all groups of time-series in all iterations when compared to the one that used the Negative Binomial distribution. As such, using the Poisson distribution seems to be the best option at this level of aggregation.

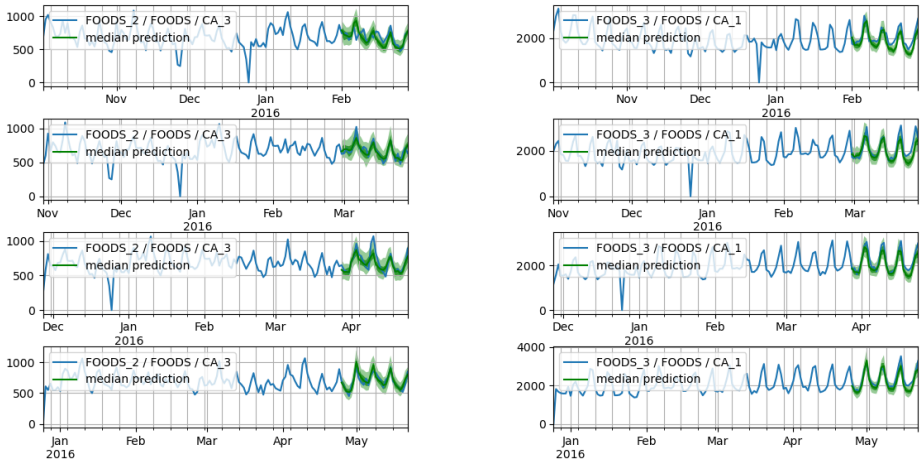


Figure 4.24: **M5 Level 9 - DeepAR - Negative Binomial Distribution -** Q3-MAX forecasting examples

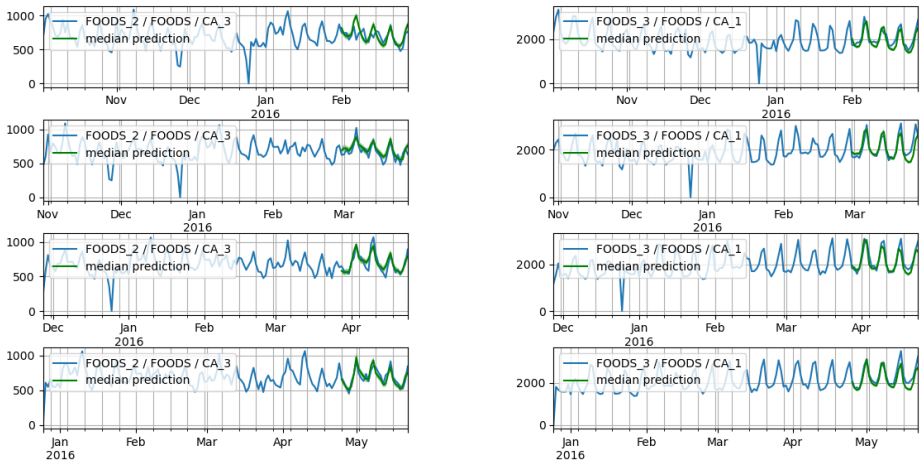


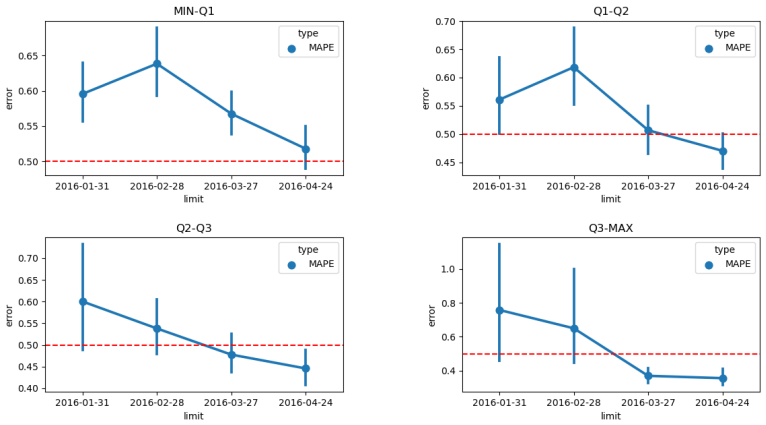
Figure 4.25: **M5 Level 11 - DeepAR - Poisson Distribution -** Q3-MAX forecasting examples

In the plots of B.4, B.5, B.6, and 4.24 there are some examples of forecasts in time series of the groups *MIN-Q1*, *Q1-Q2*, *Q2-Q3*, and *Q3-MAX*, respectively, done by the *DeepAR* models that use the Negative Binomial distribution. The B.7, B.8, B.9, and 4.25 graphs show examples of predictions made by the *DeepAR* model using a Poisson distribution. Both DeepAR models using

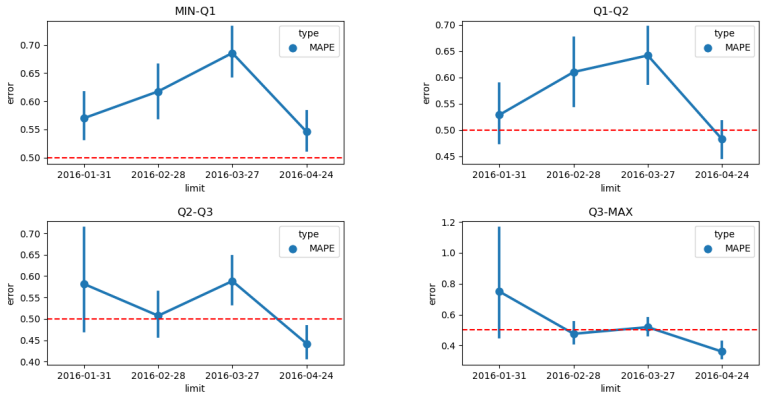
the *Negative Binomial* distribution and the *Poisson* distribution seem to adjust well to the behavior of the time series at this level of aggregation, capturing its weekly seasonality.

4.1.2.2 M5 - Level 10 Aggregation

Analyzing the evolution of the *MASE* error in figures and B.22a e B.22b, there does not seem to be the same tendency for this error to decrease in the time-series groups with higher sales magnitudes using both Negative Binomial and Poisson distributions.



(a) Using the Negative Binomial Distribution.



(b) Using the Poisson Distribution.

Figure 4.26: M5 Level 10 - DeepAR - Expected value and confidence interval of the MAPE errors over training sets with increasing sizes (limit).

However, analyzing the *MAPE* errors in figures 4.26a and 4.26b, it appears that this error decrease the higher the magnitudes of the time series group are, using both the Negative Binomial and Poisson distributions.

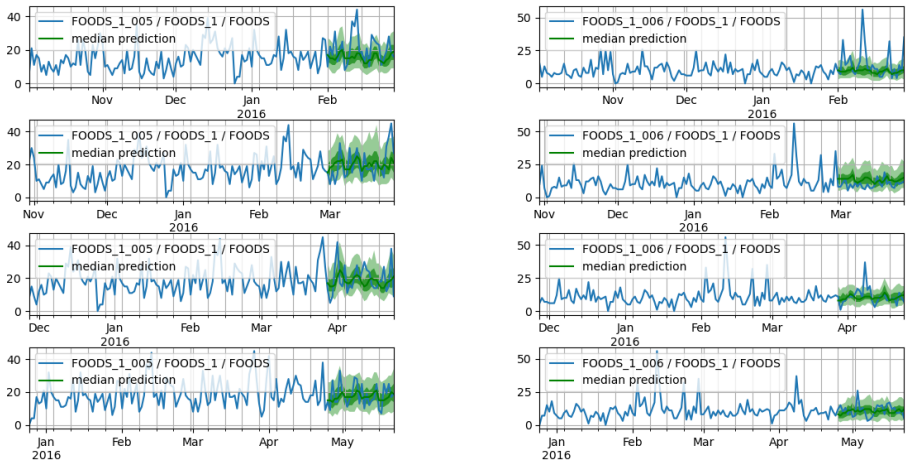


Figure 4.27: **M5 Level 10 - DeepAR - Negative Binomial Distribution -** Q2-Q3 forecasting examples

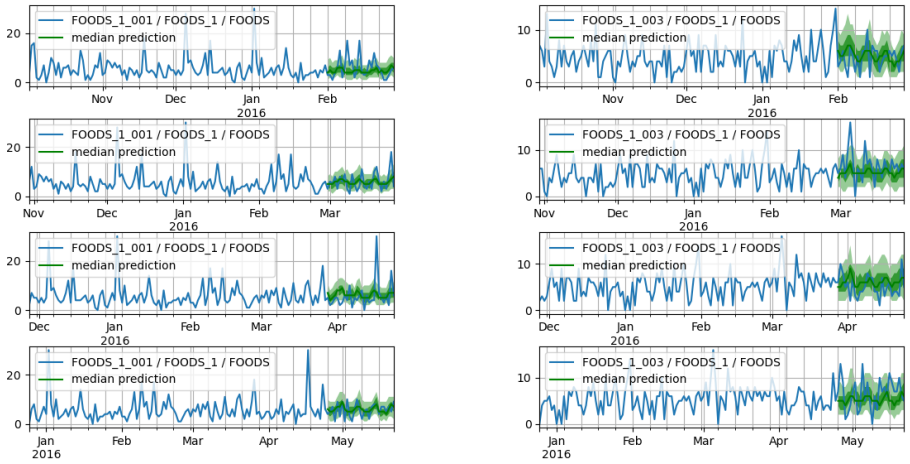


Figure 4.28: **M5 Level 10 - DeepAR - Poisson Distribution -** Q1-Q2 forecasting examples

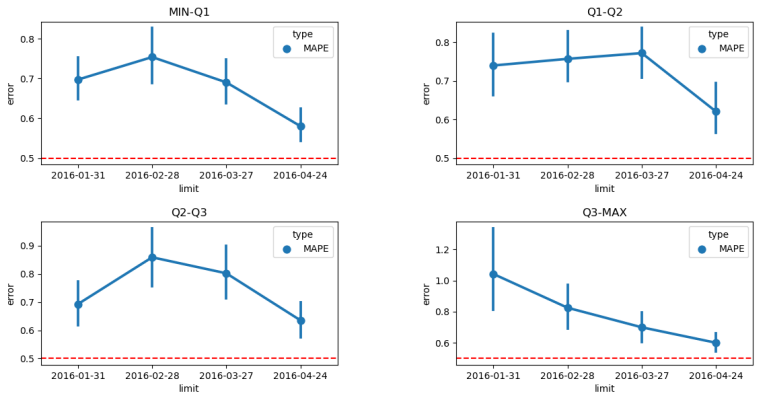
In the plots of B.10, B.11, 4.27, and B.12 there are some examples of forecasts in time series of the groups *MIN-Q1*, *Q1-Q2*, *Q2-Q3*, and *Q3-MAX*, respectively, done by the *DeepAR* models that use the Negative Binomial distribution. The B.13, 4.28, B.14, and B.15 graphs show examples of predictions made by the *DeepAR* model using a Poisson distribution.

The time-series behavior appear to be much more irregular and more difficult to predict from previous sales at this level of aggregation than at level 9. The model seems to have more difficulty capturing a pattern of sales over time, getting much wider uncertainty intervals than those obtained at the level 9 of aggregation, which were much narrower and with well-defined behavior. The

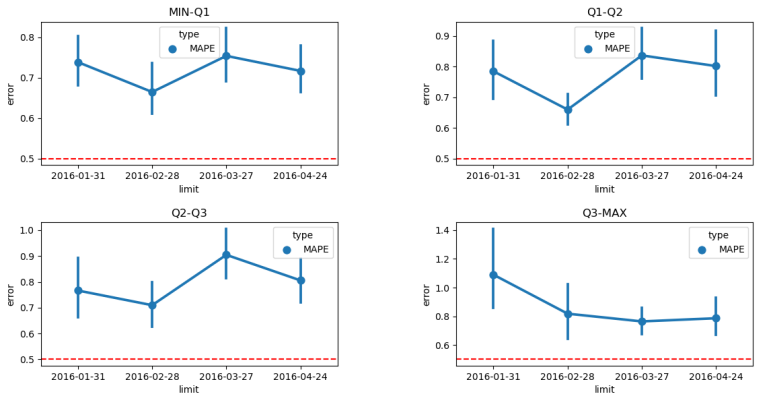
MAPE and *MASE* results in B.5 are also higher than those obtained at aggregation level 9, as expected.

4.1.2.3 M5 - Level 11 Aggregation

As happened in the level of aggregation 10, in figures B.23a and B.23b there does not seem to be the same tendency for the MASE error to decrease in the time series groups with higher sales magnitudes using both Negative Binomial and Poisson distributions. On the other hand, in figures 4.29a and 4.29b, the MAPE error shows that the model obtains better results in time-series of groups where the magnitudes of sales are higher.



(a) Using the Negative Binomial Distribution.



(b) Using the Poisson Distribution.

Figure 4.29: **M5 Level 11 - DeepAR** - Expected value and confidence interval of the MAPE errors over training sets with increasing sizes (limit).

As done for the remaining levels, the plots B.16, 4.30, B.17, and B.18 show some examples of forecasts in time-series of the groups *MIN-Q1*, *Q1-Q2*, *Q2-Q3*, and *Q3-MAX* using DeepAR

with Binomial Negative distribution. The plots B.19, B.20, 4.31, and B.21 show some examples using a Poisson distribution.

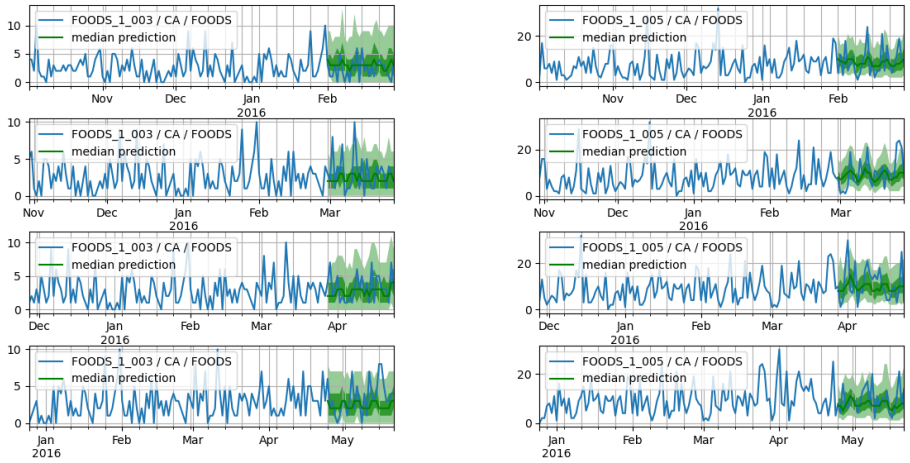


Figure 4.30: **M5 Level 11 - DeepAR - Negative Binomial Distribution -** Q1-Q2 forecasting examples

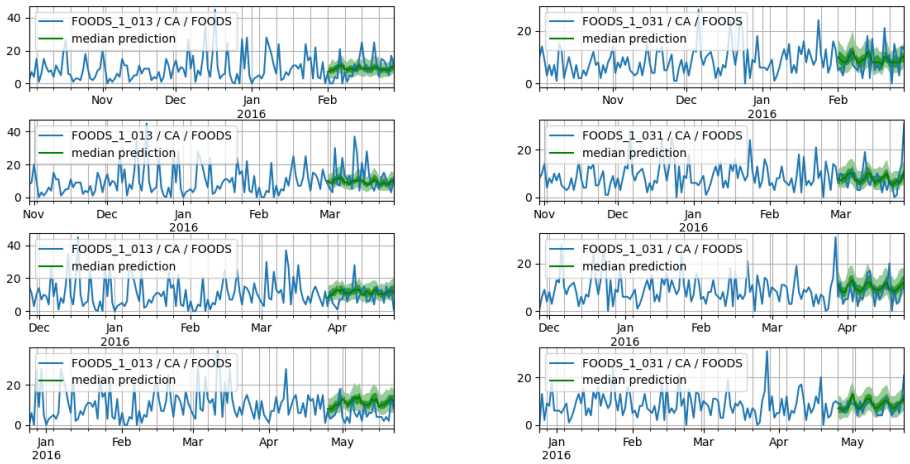


Figure 4.31: **M5 Level 11 - DeepAR - Poisson Distribution -** Q2-Q3 forecasting examples

Analyzing the plots and errors in Table B.6, both *DeepAR* models seem to present the same difficulties at this level of aggregation as they did at level 10: time series with low values, irregular or even intermittent behavior, making it difficult to predict the which results in forecasts with wider uncertainty intervals and larger errors.

4.2 Classic Models

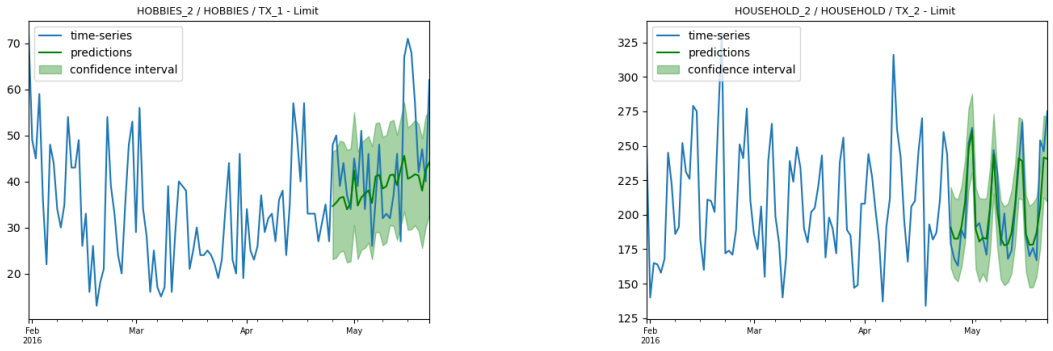
4.2.1 Statistical Models - Prophet - Probabilistic Forecasts

As mentioned in 3.4.3, we must select a small quantity of time-series to be predicted with Prophet. This model requires new training to fit its parameters for each time-series it will predict. In fact, for each time-series the Prophet is tuned in order to find the best Hyper-Parameters that will be used to fit the parameters of a new model on the specific time-series. As such, a new model is created to predict a new time-series, which is time-consuming. As such, for each aggregation level of the M5 competition we select a sample of time-series from the dataset:

- **Level 9:** all *70 time-series* of this level of aggregation.
- **Level 10:** *60 time-series* with information on demand for 20 products of each category.
- **Level 11:** *45 time-series* with information on demand for 5 products of each category in all 3 states.

4.2.1.1 M5 Dataset - Level 9 of Aggregation

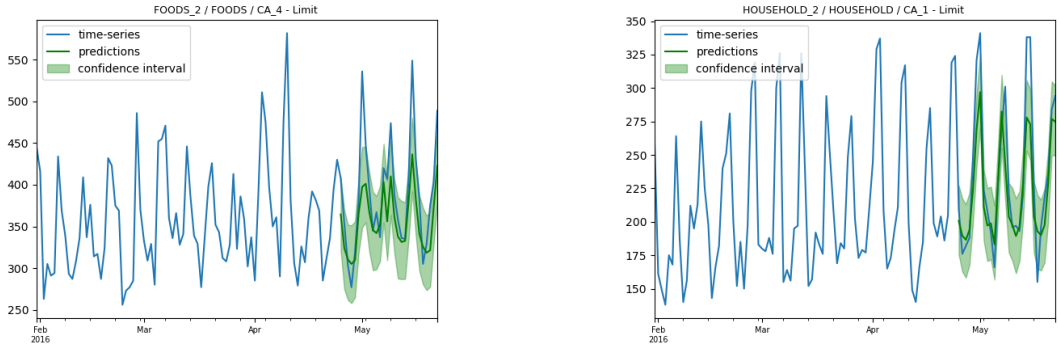
The Figures 4.32, 4.33, 4.34, and 4.35, show some examples of predictions of Prophet models in time-series of groups *MIN-Q1*, *Q1-Q2*, *Q2-Q3*, and *Q3-MAX*, respectively. It is possible to observe that at this level, where sales are aggregated by Department and Store, with the time-series having more regular and well-defined behavior over time, the model predictions follow the variations of the time series with accuracy. Thus, they are also able to capture possible sales trends over the horizon, as seen in 4.32a.



(a) HOBBIES_2 - TX_1

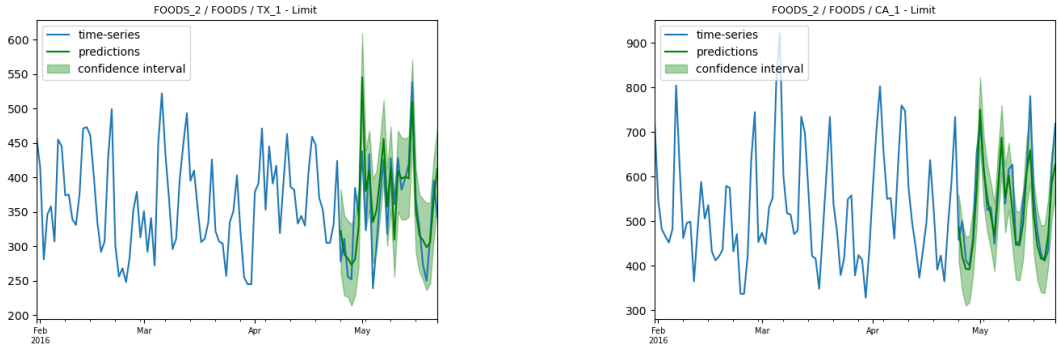
(b) HOUSEHOLD_2 - TX_2

Figure 4.32: **M5 Level 09 - Prophet** - MIN-Q1 forecastig examples.



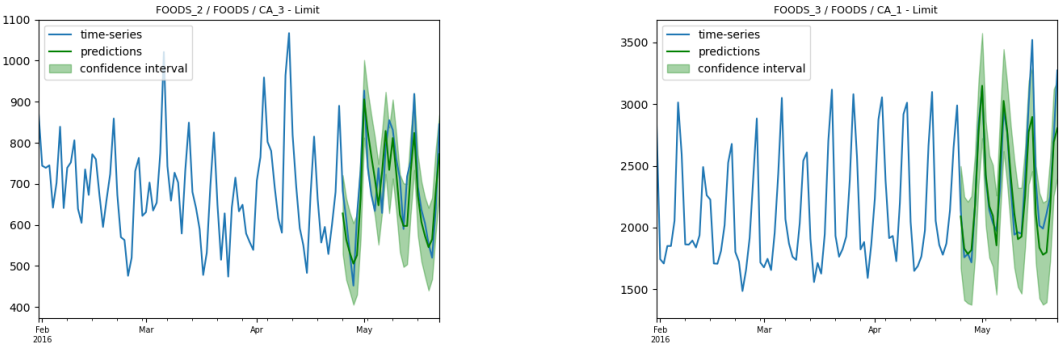
(a) FOODS_2 - CA_4 (b) HOUSEHOLD_2 - CA_1

Figure 4.33: **M5 Level 09 - Prophet** - Q1-Q2 forecastig examples.



(a) FOODS_2 - TX_1 (b) FOODS_2 - CA_1

Figure 4.34: **M5 Level 09 - Prophet** - Q2-Q3 forecastig examples.



(a) FOODS_2 - CA_3 (b) FOODS_3 - CA_1

Figure 4.35: **M5 Level 09 - Prophet** - Q3-MAX forecastig examples.

Table B.7 has statistics on errors obtained in the time-series of different groups. The results confirm that these models obtain very good time-series predictions. The mean of MAPE errors

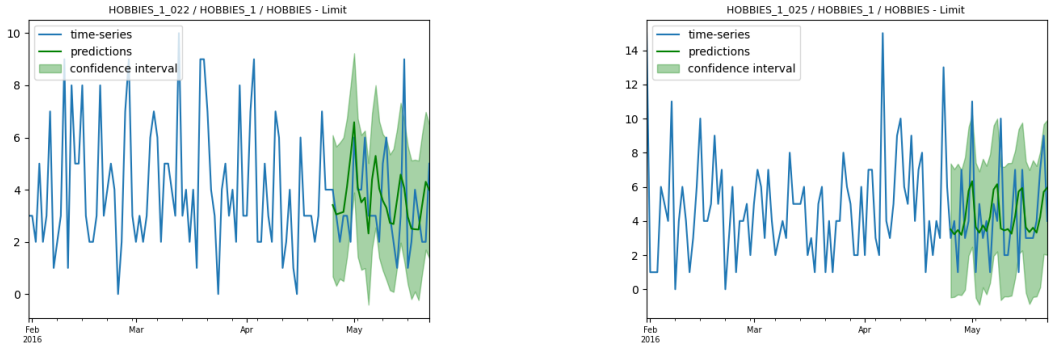
is always less than 20%, decreasing for groups with successively higher sales magnitudes: means of 18.7%, 13.94%, 11.95%, and 8.27% for groups MIN-Q1, Q1-Q2, Q2- Q3, and Q3-MAX. These results are also consistent across all time series of a group, which can be seen in the small values of standard deviation in all groups: 7.62%, 3.73%, 2.69%, and 1.49%, respectively. These models present **better MASE results** for this level of aggregation compared to Decision Trees and DeepAR models, obtaining mean values of 0.825, 0.919, 0.651, 0.582, and standard deviations of 0.283, 0.354, 0.190, 0.176 for the MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX groups, respectively.

As such, the Prophet only needs past time series values and some event, calendar, and price variables to get good predictions on time-series with well-defined behaviors. The good results of these models at this level of aggregation can also be noticed in the narrow confidence intervals of the forecasting plots. However, tuning and fitting the parameters of a new model for each time-series takes time. Thus, Prophet models take much more time to create models to predict 70 time-series - and, consecutively, to get results for all sample time-series - when compared to the times required by Decision Trees and DeepAR models.

In order to have a comparison of the training and forecast times of the different models, we use the aggregation level 9. At this aggregation level, all models train with 70 time series and obtain forecasts of future sales for all of them. In the 4 validation iterations, the DeepAR model using a Poisson distribution fits and obtains predictions in 556.798 seconds, 563.729 seconds, 593.962 seconds, and 663.067 seconds, that is, an average of almost 10 minutes to fit and predict. values of the 70 time-series in one iteration. On the other hand, the DeepAR models using the Negative Binomial distribution, obtained an average of just over 8 and a half minutes to train and obtain results in 70 time-series. The Gradient Boosting models took an average of about 13 minutes and 48 seconds, while the Random Forest models took an average of 22 minutes and 10 seconds to get results for all 70 time-series. Finally, the Prophet models took an average of 698.59 seconds to tune and fit its parameters for 1 individual time-series, taking more than 13 and a half hours to obtain results for the 70 time-series of this level of aggregation. Thus, despite the better results of the Prophet models, we must reconsider their use in retail forecasting environments where it is necessary to forecast the sales of several different products. In these cases, the DeepAR models are faster than the other models, being their drastic difference in relation to the prediction training time of the prophet models.

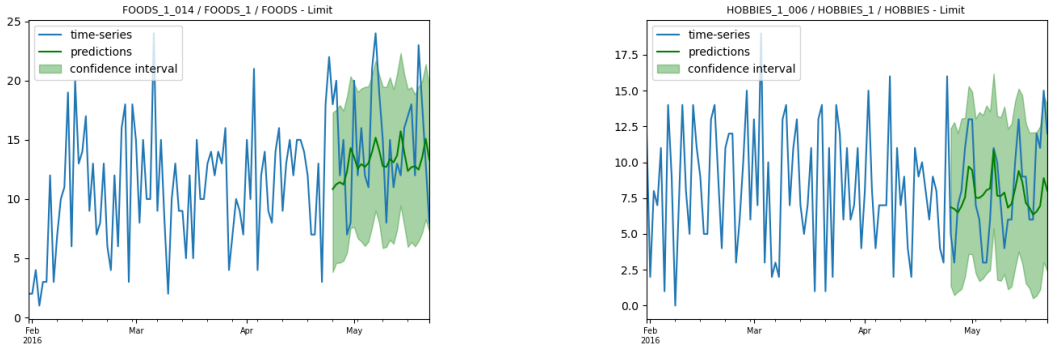
4.2.1.2 M5 Dataset - Level 10 of Aggregation

At aggregation level 10, sales become more volatile, irregular and difficult to predict. However, the Prophet models are still able to capture some weekly seasonality and follow the sales variations of the different groups, but with wider confidence intervals in relation to level 9 - as can be seen in 4.36, 4.37, 4.38, 4.39.



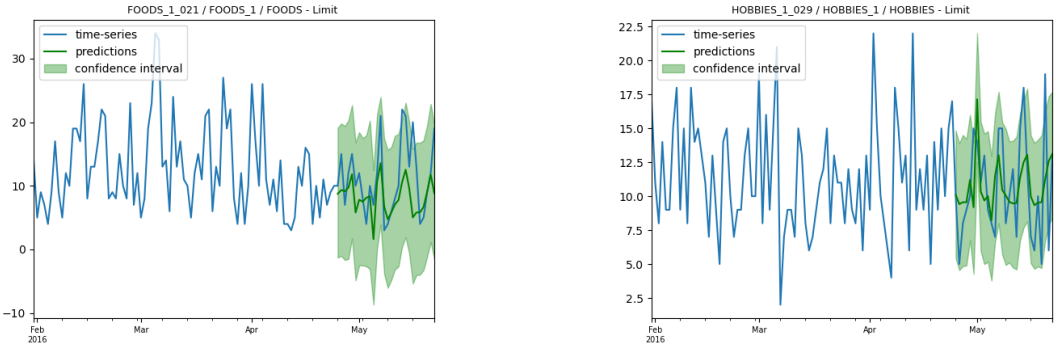
(a) HOBBIES_1_022 - HOBBIES_1 (b) HOBBIES_1_025 - HOBBIES_1

Figure 4.36: **M5 Level 10 - Prophet** - MIN-Q1 forecasting example.



(a) FOODS_1_014 - FOODS_1 (b) HOBBIES_1_006 - HOBBIES_1

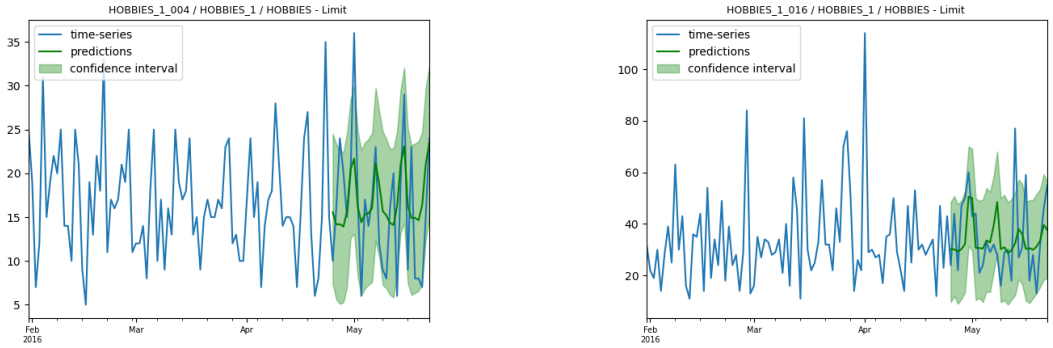
Figure 4.37: **M5 Level 10 - Prophet** - Q1-Q2 forecasting example.



(a) FOODS_1_021 - FOODS_1 (b) HOBBIES_1_029 - HOBBIES_1

Figure 4.38: **M5 Level 10 - Prophet** - Q2-Q3 forecasting example.

The statistics of the errors obtained in the time-series of different magnitude groups are summarized in Table B.8.In the lower magnitude groups of this level - MIN-Q1 and Q1-Q2 - the Prophet



(a) HOBBIES_1_004 - HOBBIES_1

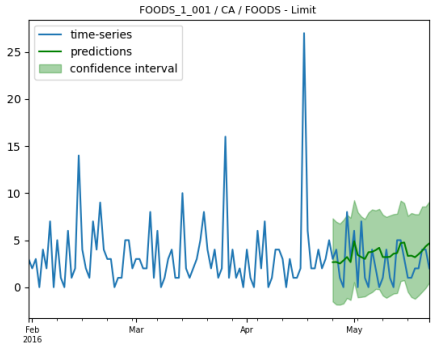
(b) HOBBIES_1_016 - HOBBIES_1

Figure 4.39: **M5 Level 10 - Prophet** - Q3-MAX forecasting example.

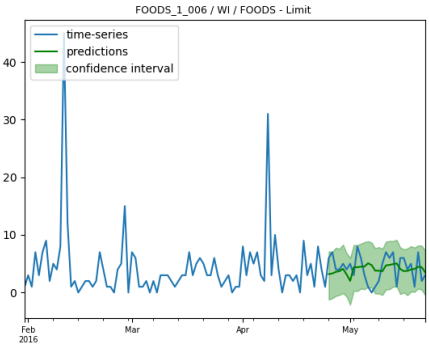
models obtained better MASE results than in the lower magnitude groups of level 9: mean values of 0.765 and 0.754, and standard variations of 0.156 and 0.174, respectively. In the groups with higher magnitude - Q2-Q3 and Q3-MAX - where sales reach much higher values at level 9, we can see that the MASE error worsens at level 10 - average values 0.696 and 4.199, respectively, and standard variation values 0.132 and 12,835. The values for MASE error in the Q3-MAX group are much higher due to the fact that we selected a much smaller sample of time-series, and the few time-series of this magnitude show great intermittence and very sudden variations in sales on the horizon, which are very difficult to predict - as is the case in examples 4.43a and 4.43b. As such, Prophet gets better MASE results than the Decision Trees and DeepAR models in groups MIN-Q1, Q2-Q3, and Q3-MAX. MAPE results become unreliable, having very high values or tending to infinity as a cause of the frequent intermittence of time series at this level.
As with level 9, Prophet models take much longer than Decision Trees and DeepAR models to fit and make predictions, so we selected a much smaller sample of time-series.

4.2.1.3 M5 Dataset - Level 11 of Aggregation

At aggregation level 11, the irregularities and intermittency already noticeable in time-series at level 10 become even more evident. As such, Prophet model forecasts at this level also have wider confidence intervals. The model tries to follow the movements of future sales with some difficulty - 4.40, 4.41, 4.42, and 4.43.

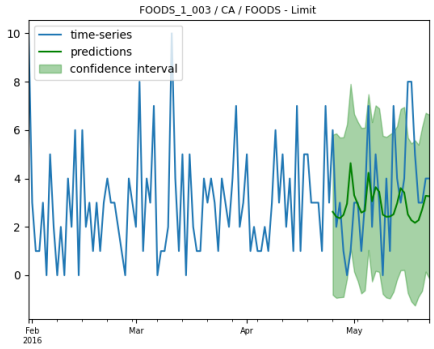


(a) FOODS_1_001 - CA

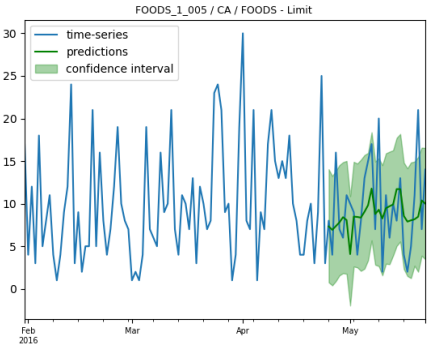


(b) FOODS_1_006 - WI

Figure 4.40: **M5 Level 11 - Prophet** - MIN-Q1 forecasting example.

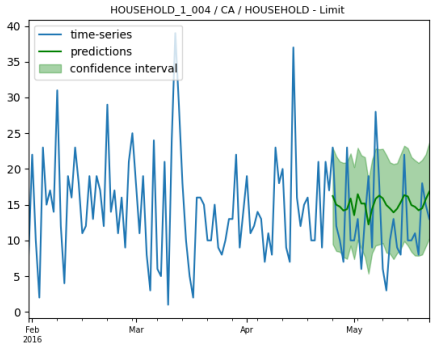


(a) FOODS_1_003 - CA

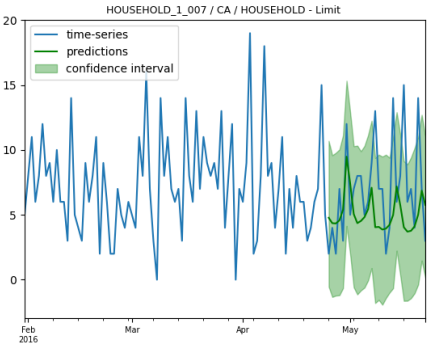


(b) FOODS_1_005 - CA

Figure 4.41: **M5 Level 11 - Prophet** - Q1-Q2 forecasting example.



(a) HOUSEHOLD_1_004 - CA

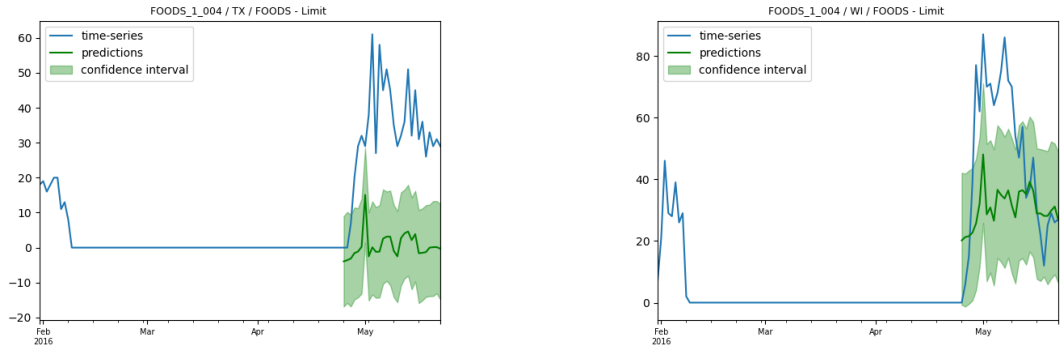


(b) HOUSEHOLD_1_007 - CA

Figure 4.42: **M5 Level 11 - Prophet** - Q2-Q3 forecasting example.

In Table B.9, MASE errors are smaller in groups MIN-Q1 and Q1-Q2 when compared to levels 9 and 10: mean values of 0.626, 0.655, and standard deviation values of 0.213, 0.089, respectively.

In groups Q2-Q3 and Q3-MAX, MASE errors are higher at this level than at the levels 9 and 10 - mean values of 0.811 and 42.141 with standard deviations of 0.211 and 37.025. The values for MASE error in the Q3-MAX group (3 in total) are much higher due to the fact that we selected a much smaller sample of time-series, and the few time-series of this magnitude show great intermittence and very sudden variations in sales on the horizon, which are very difficult to predict - as is the case in examples 4.43a and 4.43b. MAPE errors are unreliable for the same reason as described in level 10.



(a) FOODS_1_004 - TX

(b) FOODS_1_004 - WI

Figure 4.43: **M5 Level 11 - Prophet** - Q3-MAX forecasting example.

At this level of aggregation, Prophet has the best MASE error results compared to decision tree and DeepAR models in the MIN-Q1, Q1-Q2, Q2-Q3 groups - at the cost of much higher tune and fit times. The errors of the Prophet models in the Q3-MAX group are worse than those of the other models.

Chapter 5

Conclusions

During the research done for the development of the State-of-Art, it was possible to conclude that there are several studies on the development of forecasting models to predict the sales of retailers with high accuracy. It was also possible to systematize the advantages and disadvantages of such models. In this way, it was possible to understand which models are the most appropriate to our study, in order to achieve our goals. Classic models, despite having their successes forecasting sales with a linear behavior, are not able to forecast sales with irregular behavior, neither to include input variables that express the effect of demand influencing factors. As we are going to deal with retail products that present irregular sales as a result of the effect of demand influencing factors, we need more complex machine learning models, capable of approximating non-linear sales and able to receive input variables. However, many of these machine learning models are complex and require a lot of time, effort, and expertise: either in the selection and preparation of input variables that represent the demand influencing factors or in the preparation of the features of the model itself. As such, the best option is to use models based on Deep Learning techniques, such as DeepAR, that require minimal manual effort and expertise in preparing parameters or variables, since they can learn from the data. Another conclusion that was possible to obtain in the research of the different types of models, is that despite the advantages and disadvantages of each one, no model is the best in all situations and for all datasets.

From the datasets search described in Section 3.1, there seems to be a lack of datasets for the purpose of demand/sales forecasting and with information on the characteristics/attributes of the products and the stores where they are sold. Many of the datasets that presented product or store attributes were not created for the purpose of forecasting and, therefore, did not have time or sales information. When datasets had product titles, these had little or no useful information. When titles had relevant information such as brand, category, size, weight, and ingredients, the time-series were too short - with less than 5-time points each. Much of this lack of information may be a consequence of retailers' concern with the confidentiality of their data. Furthermore, most datasets

created and made available to promote the development of forecasting models do not include static variables with information on characteristics or attributes of products and stores. Thus, the lack of available data may be one of the main reasons for the lack of studies on forecasting models that include the effect of product and store attributes.

Good preparation and cleaning of the selected datasets were crucial, helping in several subsequent steps of the work:

1. **Downcasting** the data-frames that hold information from different files, as described in Sub-subsection 3.2.1.2, reduces the space occupied by them in memory and makes future manipulations of these data structures faster and more efficient. It accelerates the process of merging information from all files into a single file to be fed to the models. It also accelerates all steps of Feature Engineering, where useful features are created from existing variables of the dataset to be fed as input to the Machine Learning models. It also speeds up the generation of graphics and plots from variables of the dataset to do its Analysis.

2. **Melting** the original format of sales/demand data, as described in 3.2.2.1 and 3.2.1.2, makes it easier to use built-in methods from the Python libraries to restructure the data whenever necessary, as well as to do data analysis. The efficient restructuration of data is important to aggregate sales of different products - by category, department, store, state, etc. - while losing the little information as possible, as well as to frame the data into a new format required by the different models.

3. **Cleaning** consisted of replacing missing data in the numeric and categorical variables of the datasets, as described in Sub-subsection 3.2.1.2. In numerical variables, missing values are replaced by the mean value of these variables in the time-series to which they belong. This substitution is important as many Machine Learning models do not accept null values in numerical variables. In categorical variables, missing values are replaced by a default label that indicates the absence of a value in this variable. This substitution facilitates the encoding of categorical variables into numeric variables during the Feature Engineering phase, which is essential to use Machine Learning models that only accept numeric variables as input.

A meticulous analysis of the datasets is essential to make important decisions, such as:

1. The length of the window to forecast, also known as horizon. For example, the analysis of the M5 dataset in 3.2.2.2 showed that it has a strong weekly seasonality. As such, we set a horizon of 28 days, corresponding to 4 weeks, to forecast.

2. Which dataset variables seem to have an impact on the target variable and should be used as features in forecasting models.

3. The steps to be performed in Feature Engineering in order to create useful features from the selected variables and that can be used as input in the forecasting models. For example,

as already mentioned, the analysis of the m5 dataset shows a strong weekly seasonality. As such, we decided to use lags from the previous 4 weeks - lags 7, 14, 21, and 28 - as input features in Random Forest and Gradient Boost models, as they should have a higher predictive value of future time-series values.

The application of an efficient and reliable validation, as is the case of the Walk-Forward Validation described in 3.18, allowed us to make a robust evaluation of the models in the prediction of different horizons of the time series over time. This validation allowed us to verify that the results of the DeepAR, Random Forest, and Gradient Boost models remain consistent over time, as new sales values are known and used to predict new horizons of all time series - Figures 4.23, 4.22, 4.3, and 4.4. These figures also show that, when sales are stable and the magnitudes of the time-series are high, as is the case with the time-series of the Q3-MAX group, the DeepAR models even tend to learn to better predict the sales of new horizons. as more historical data are added for training over time - Figures 4.23, 4.22.

At aggregation level 9, all models were able to obtain good predictions from the time-series, capturing their behavior over time with relative ease. At this level, each time-series corresponds to the aggregation of sales of products by Department and Store, with their magnitudes becoming much higher and their behavior more regular and with strong seasonality. Within the Global Models, Random Forest obtained the best *MASE* results, while the DeepAR models, using one of the two distributions, obtained the best *MAPE* results in almost all time-series groups. The Random Forest model did not use or attach importance to the remaining features other than the lags at this level of aggregation. Prophet obtained the best *MASE* and *MAPE* results among all models in almost all groups of time-series and does not use static variables on products or store location to help with predictions. The results of the Random Forest and Prophet models show that, when the time-series have a stable behavior and strong seasonality and trends, the past values of the time series, also referred to as lags, are enough information for the models to obtain accurate predictions of future sales. To predict more regular time-series, DeepAR models seem to perform better using a Poisson distribution rather than the Negative Binomial distribution in all groups of time-series (MIN-Q1, Q1-Q2, Q2-Q3, Q3-MAX).

Time-series become more irregular and difficult to predict at levels 10 and 11, where each time-series corresponds to sales of a product - across all stores/states at level 10 and aggregated by State at level 11. At these levels of aggregation, the order of features' importance changed in *Random Forest*: in addition to lags - which are still the most important features - statistical variables such as *cat_id*, *dept_id*, and *store_id*, also began to be used in splitting the trees of decision, to help predict future time series values. This change in *Random Forest* decision trees was more evident at level 11. In general, the Global Models obtained worse *MASE* results at these two levels of aggregation compared to level 9, this difference being notable in the time-series groups with higher magnitudes of the different levels - *Q2-Q3*, *Q3-MAX*. In this levels, the MAPE errors become unstable, the

values tending to infinity being constant as a result of the frequent intermittence of the time series and the presence of null values in them. Under these conditions and analyzing the MASE error, the Gradient Boost models seem to have better results than the Random Forest models when the magnitudes of sales are smaller - groups *MIN-Q1*, *Q1-Q2*, and *Q2-Q3* at level 10, as well as in groups *MIN-Q1* and *Q1-Q2* at level 11.

The results support the conclusion that no model is the best in all situations, especially when sales are unstable, volatile, and with frequent intermittence. This can be seen from the results of Random Forest and Gradient Boost at levels 10 and 11 of aggregation, where the model with the best results varies in the different groups of sales magnitudes.

At aggregation level 9, where time-series are more regular and the results more consistent, almost all Global Models get the best results in time-series groups with higher magnitudes. In these models, *MAPE* and *MASE* errors decrease along the MIN-Q1, Q1-Q2, Q2-Q3 groups, finally reaching the smallest errors in Q3-MAX. This ability of the models to obtain better predictions in time-series whose sales reach greater magnitudes is more evident in DeepAR models, where the errors decrease considerably between groups of successively larger magnitudes. This statement is in agreement with the conclusions of article [23][1] in the analysis of the results of the DeepAR models. For DeepAR models using Negative Binomial and Poisson distributions, from MIN-Q1 to Q3-MAX, the MASE errors decrease from 1.061 and 1.049 to 0.789 and 0.674, while MAPE errors decrease from 26.51% and 27.78% to 10.66% and 9.2%.

The results also show the importance of making a careful selection of the errors used in the evaluation of the models. When the sales behavior of the time series is stable, the MASE and MAPE errors allow us to make a good evaluation of the models' performance. These errors, being scaled and independent of the magnitudes of the time-series, allowed us to detect the tendency that the Global models have to obtain better results in the forecast of time-series whose sales magnitudes are greater. This trend would be impossible to detect using other popular models such as the MAE and RMSE values which, being dependent on the magnitude of the time-series, tend to be higher the greater the magnitude of sales. Furthermore, when sales are irregular, volatile, and intermittent, the MAPE error becomes unreliable, since it tends to infinity whenever a time-series has null values in the target variable. Thus, it becomes necessary to use only the MASE error to compare the performance of different models in different situations.

Although the Prophet model obtains better *MASE* results in almost all conditions, it is important to remember that it needs to fit its parameters for each time-series to be predicted, adjusting to the behavior of the target values overtime of that specific series. Thus, this model requires time to tune the Hyper-parameters and to fit a new model for each time-series. Although this is a strength of the Prophet model, which adapts very well to the behavior of individual time-series, it is impractical

[1] Not available yet, as it is under corporate secrecy

to use it to predict several time-series, as this would require too much training time. This becomes an even greater disadvantage in the retail area, where it is often necessary to forecast the sales of several products or SKUs in different stores, which is reflected in a large number of time-series - as is the case of levels 10, 11, and 12 of the M5 dataset -, which can reach thousands or even millions of time-series. In this case, the best is to opt for a Global Model, such as the Decision Trees or the DeepAR models developed in the study, that can learn a global model from all time-series, in much less training time, and being able to forecast sales for different time-series - for example, for different products in different stores.The DeepAR models were able to fit a global model with a much higher number of time-series in less time compared to the other models.

Using the GluonTS library to implement DeepAR models was quite straight-forward and required minimal manual work. In fact, it already has a DeepAR Estimator implemented, being only necessary to feed the model with all the time series in the requested format, including static and dynamic features, both numerical and categorical. It is also possible to easily adjust the way training is done on time series by assigning the values of the hyper-parameters listed in 3.4.2.1. After that, the model trains and returns predictions in the form of estimates and confidence intervals for the chosen horizon. On the other hand, the development of Random Forest and XGBoost models required more manual work and code to frame the forecasting problem as a supervisioned problem, creating context windows with lag features, selecting the most appropriate lag features for the dataset used in the tests, and adjusting the supervised problem to a task of multi-step forecasting using a recursive approach.

We have to mention that the Prophet models impressed in their ability to adapt to the behavior of time series as pure statistical models, outperforming the remaining global models in the MASE results in almost all magnitude groups of the 3 levels of aggregation. According to the results presented in the Appendix B, the different Global Models, both based on Decision Trees and DeepAR, present more balanced results among themselves in the different levels of aggregation and in the different magnitude groups of each level - there is no clear dominance of one of them in relation to the others, contrary to what happens with Prophet models. However, it is also important to note that the DeepAR models, being faster, were tested on much larger and therefore more representative time-series samples at levels 10 and 11 For example, at level 10 the DeepAR models were tested with 600 time series, and for each one, 4 results were obtained in 4 different horizons (1 per validation iteration), which corresponds to an analysis of 2400 results. These results are more numerous compared to the 240 results of the 60 time series used in the analysis of the decision tree models and than the 60 results of the Prophet models. The same happens at level 12, where the 3600 results from the 900 time series used in the DeepAR models are more representative than the 360 results from the decision tree models, which in turn are more representative than the 45 results from the Prophet models.

When the training and prediction times of the different models were compared at the same level

of aggregation, the DeepAR models using the Negative Binomial distribution were the fastest, achieving results for 70 time series in about 8 and a half minutes, followed by the DeepAR models. using Poisson distribution, able to do the same in just under 10 minutes. In turn, the Gradient Boost and Random Forest models obtained results for the 70 time-series at values close to 14 and 22 minutes, respectively. Finally, the Prophet models took more than 13 and a half hours to get results for all 70 time series, which is a drastic difference in the wait time for results. This problem is even worse in situations of large retailers where it is necessary to predict time series of several SKUs from different stores - reaching the order of thousands or millions - where the Global Models become a more viable option.

That being said, Prophet models become unsustainable in a real scenario at retailers, where it becomes urgent to predict sales of many time-series for different products in different locations. As such, the DeepAR models, despite not having lowered the error values of the Prophet models, deserve to be studied in future works, since they offer a greater possibility of responding to the needs of retailers in the near future: obtaining a model able to forecast the sales of thousands or even millions of time series in the shortest possible time, in a market where the amount of products/SKUs sold is constantly growing.

Appendix A

List of Analysis Graphics

A.1 *Synthetic Dataset Analysis*

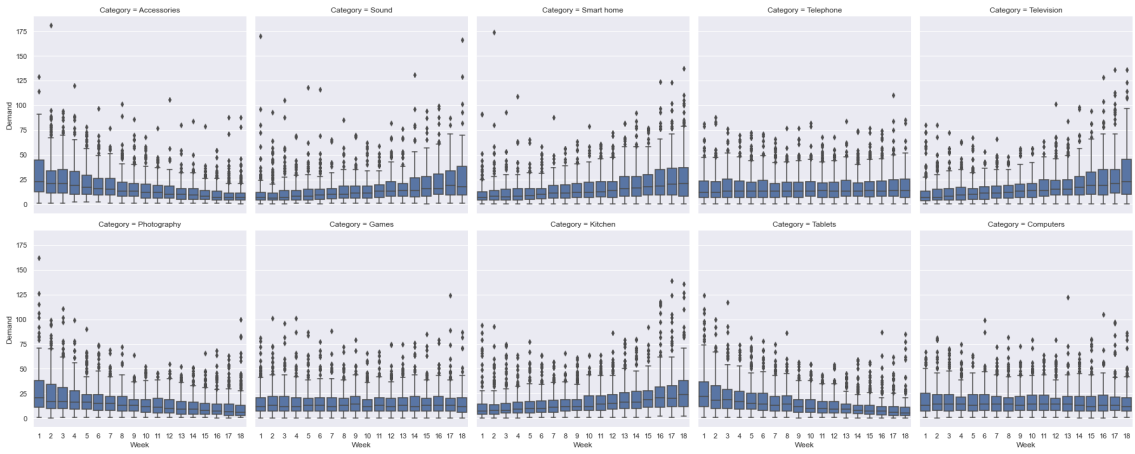


Figure A.1: Boxplot of products' demands over time grouped by categories - Synthetic Dataset.

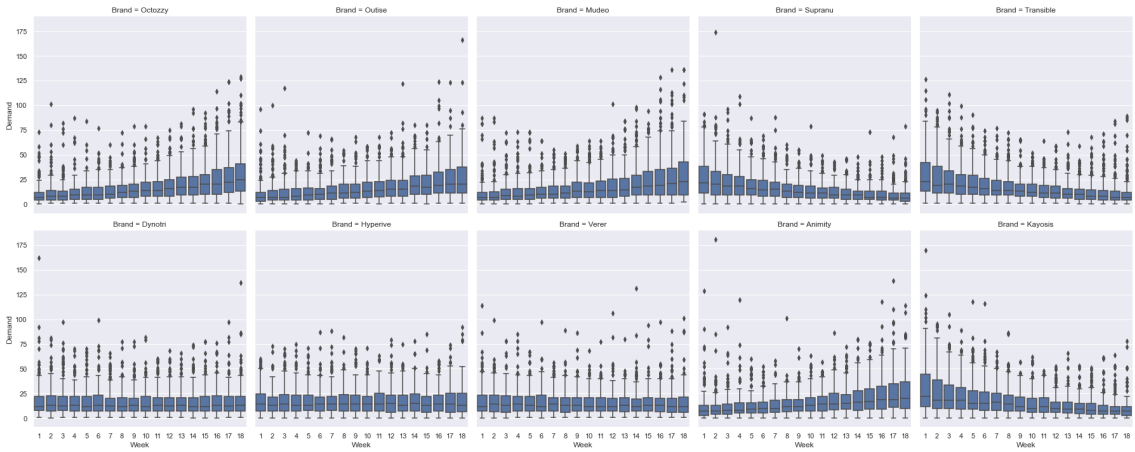


Figure A.2: Synthetic Dataset - Boxplot of products' demands over time grouped by brands.

A.2 *M5 Dataset Analysis*

A.2.1 *Level 9* - Unit sales of all products, aggregated for each Store and Department

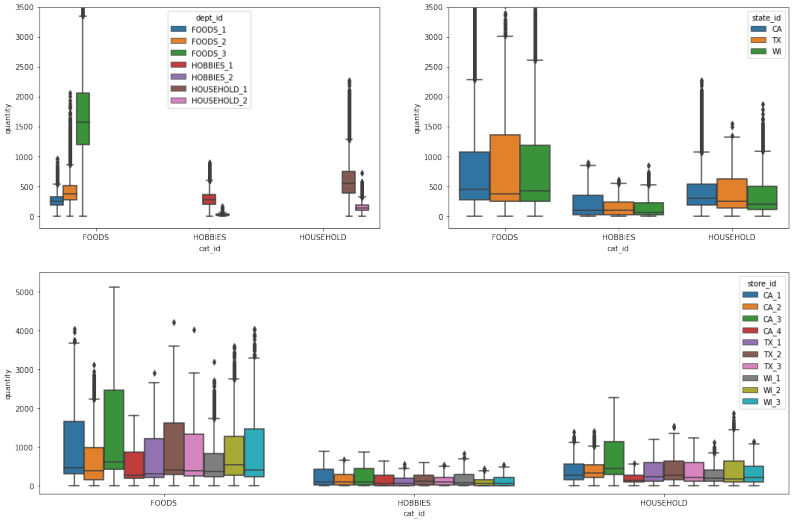


Figure A.3: M5 Level 9 - Distribution of demand for products from different categories by Department, Store, and State.

A.2.2 *Level 10* - Unit sales of product x, aggregated for all stores/states

A.2.3 *Level 11* - Unit sales of product x, aggregated for each State

A.2.4 *M5 Level 12* - Unit sales of product x, aggregated for each Store

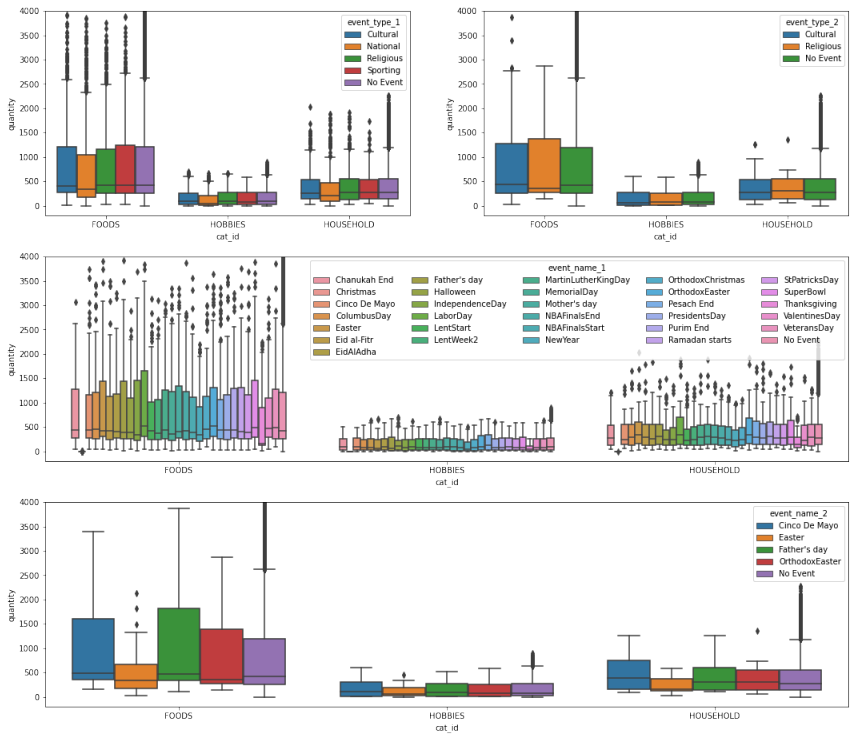


Figure A.4: M5 Level 9 - Distribution of demand for products from different categories by type of Event and name of the Event.

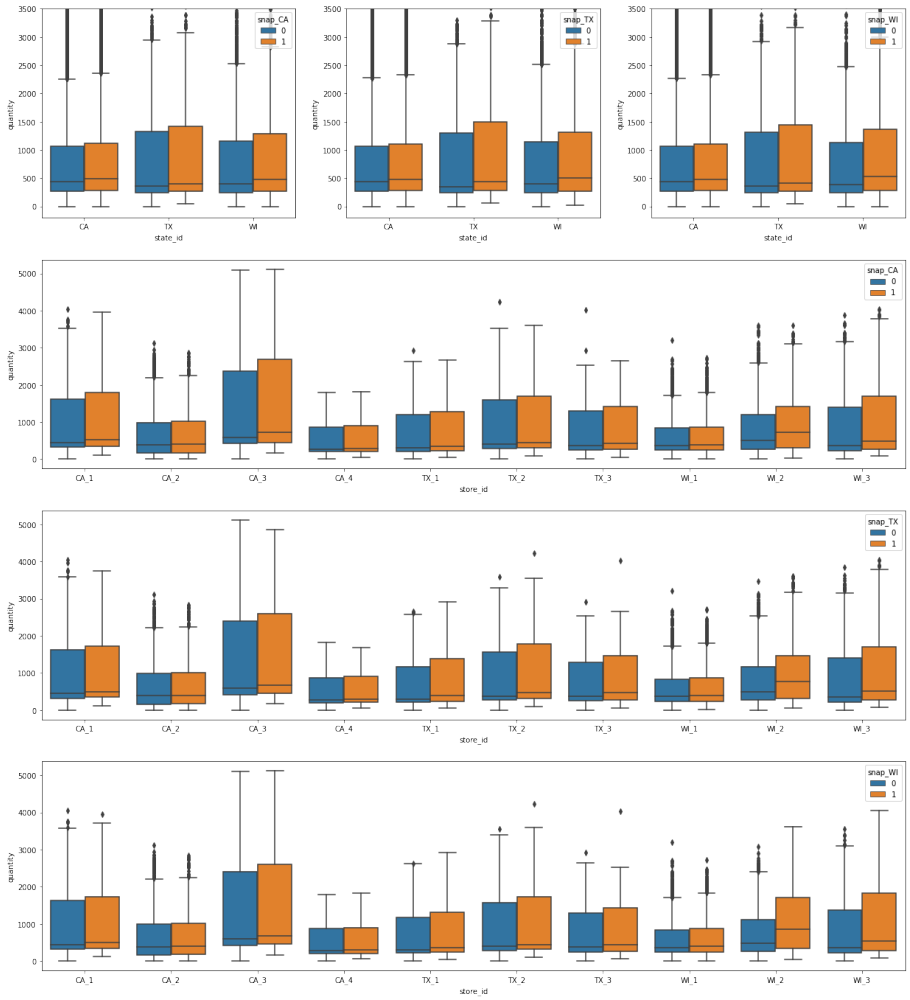


Figure A.5: M5 Level 9 - Distribution of demand for products in the *FOODS* category by snap_CA, snap_TX, and snap_WI.

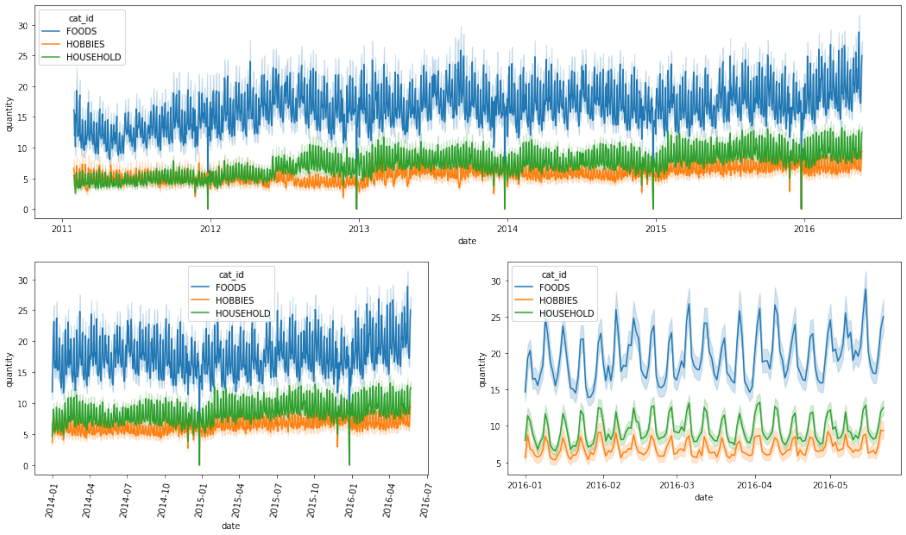


Figure A.6: M5 Level 10 - Expected value and confidence interval of demand for products, by category, over time.

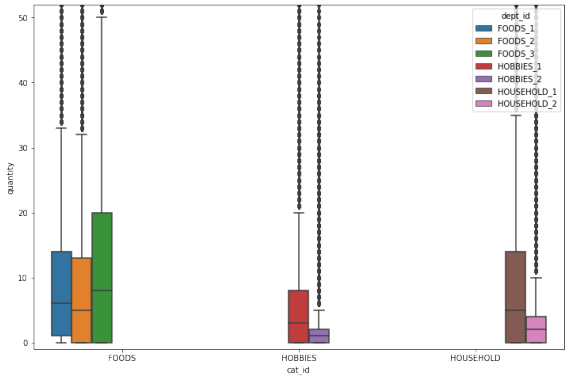


Figure A.7: M5 Level 10 - Distribution of demand for products from different categories by Department.

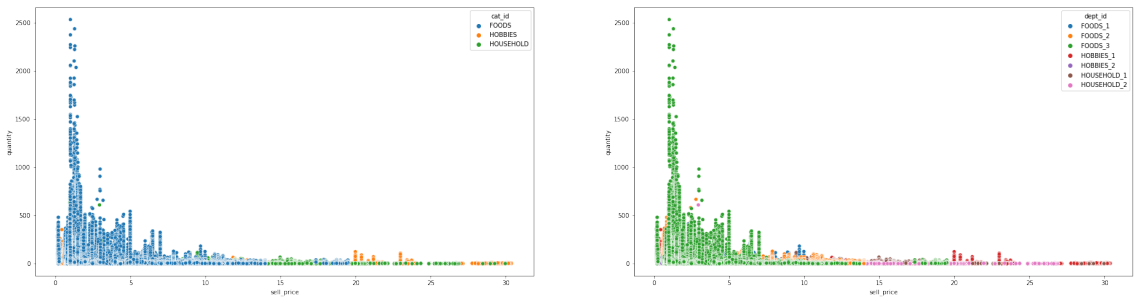


Figure A.8: M5 Level 10 - Prices versus Demand of products on different days.

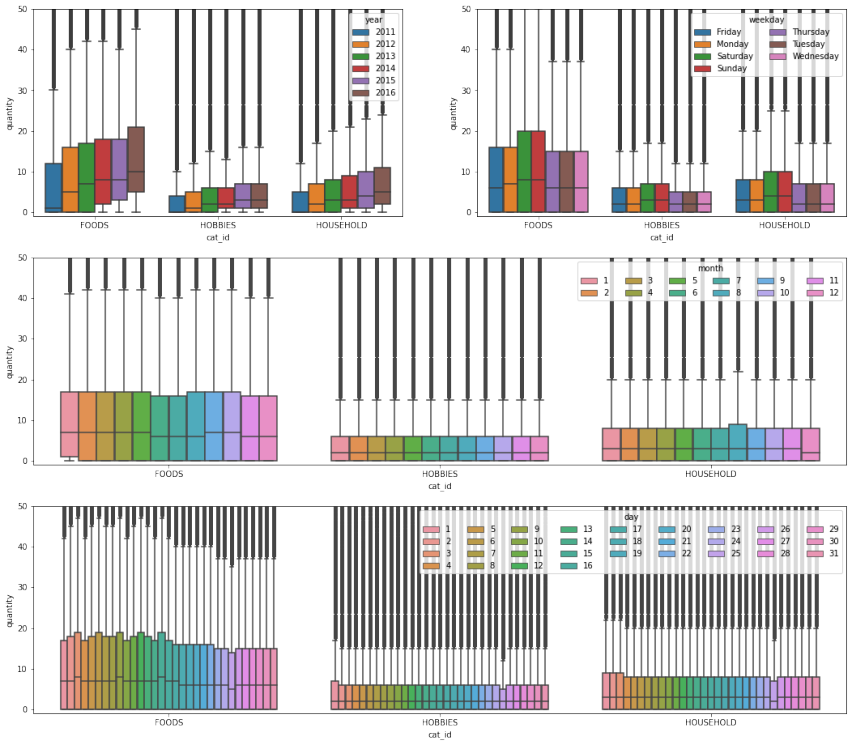


Figure A.9: M5 Level 10 - Distribution of demand for products from different categories by Day of the Week, Day of the Month, Month, and Year.

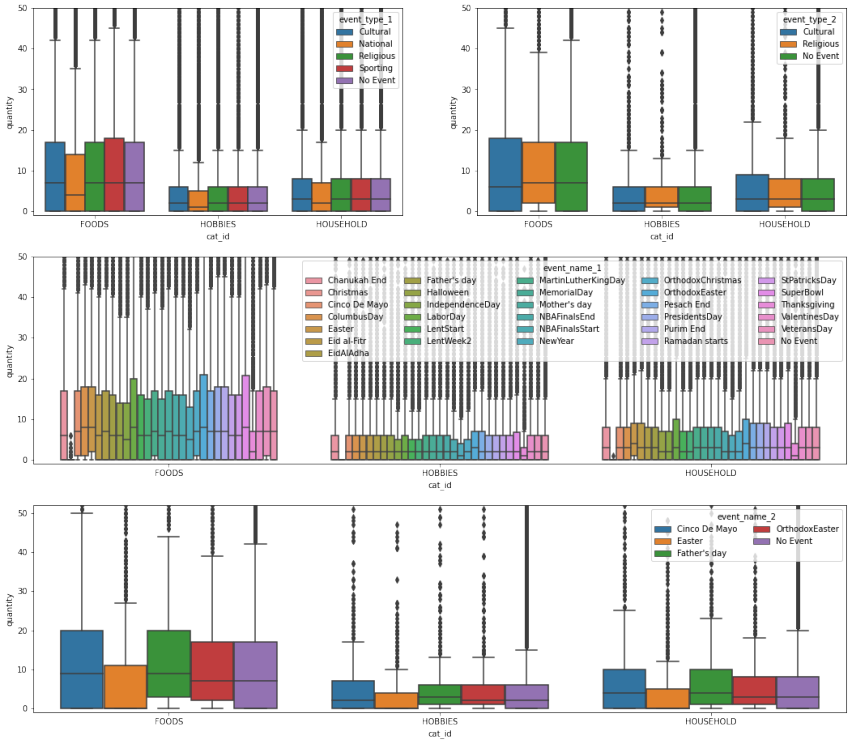


Figure A.10: M5 Level 10 - Distribution of demand for products from different categories by type of Event and name of the Event.

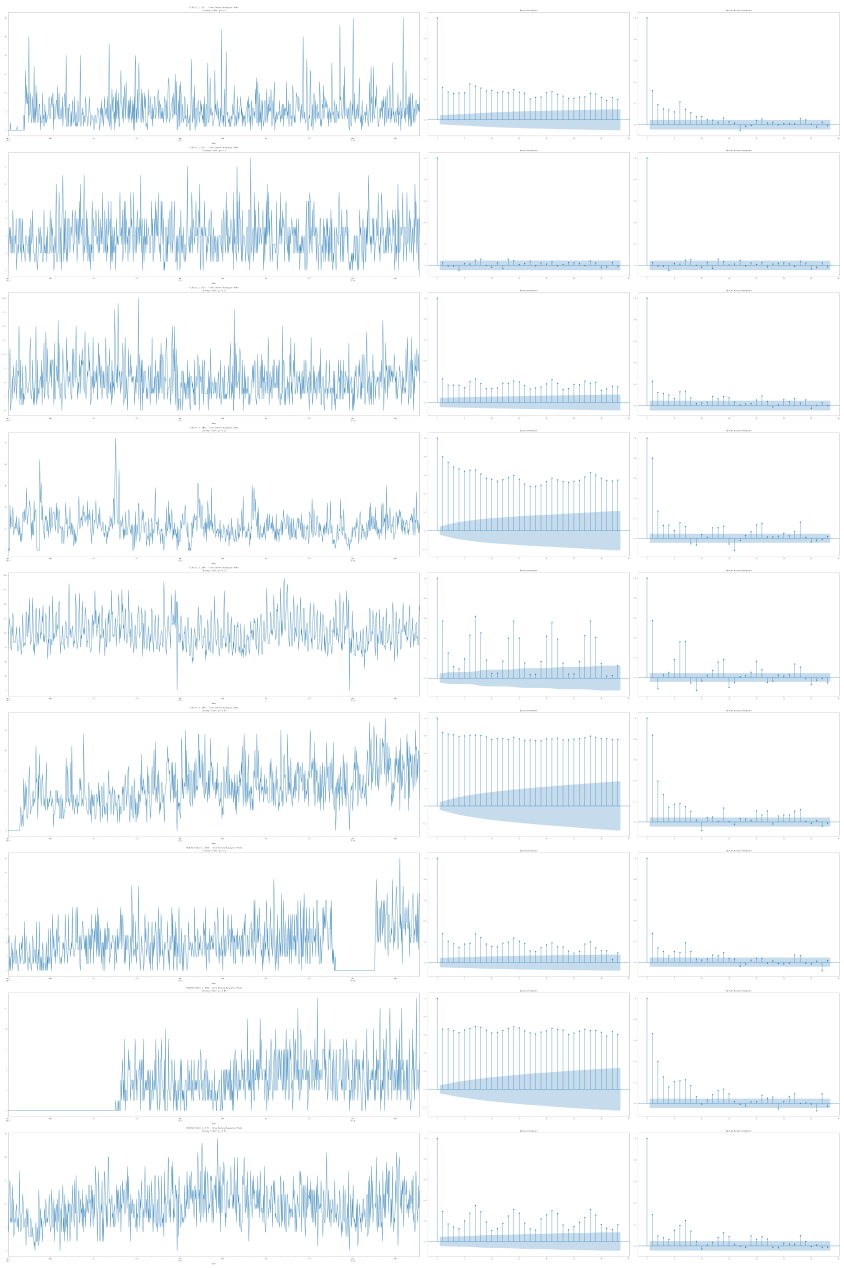


Figure A.11: M5 Level 10 - Examples of dataset Time-Series, with respective partial (PACF) and complete (ACF) auto-correlation values.

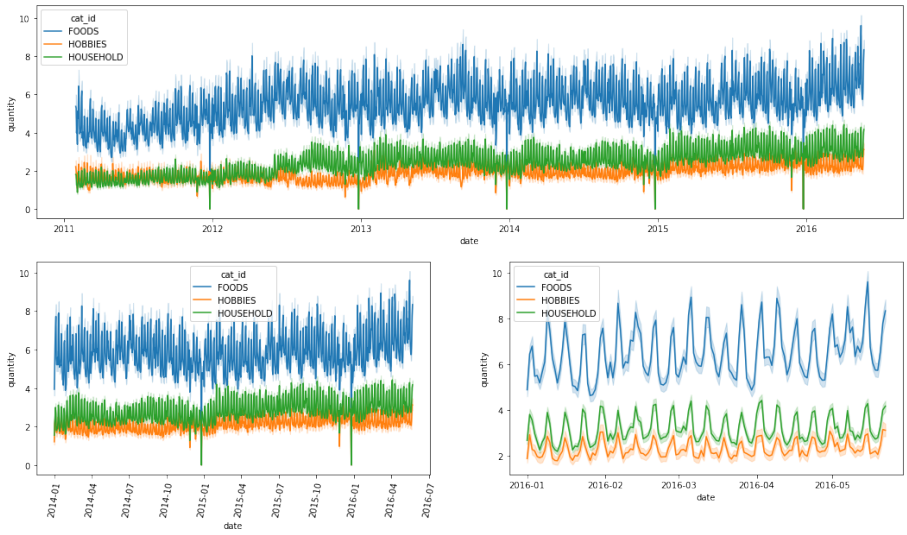


Figure A.12: M5 Level 11 - Expected value and confidence interval of demand for products, by category, over time.

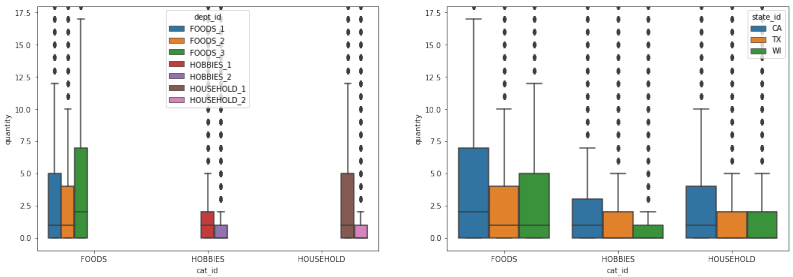


Figure A.13: M5 Level 11 - Distribution of demand for products from different categories by Department, and State.

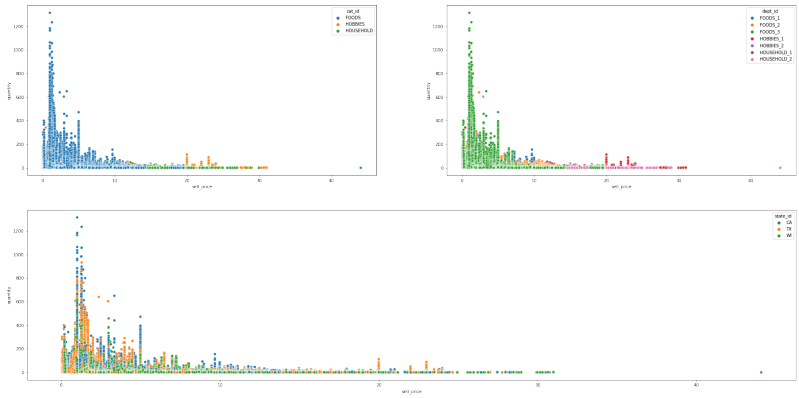


Figure A.14: M5 Level 11 - Prices versus Demand of products on different days.

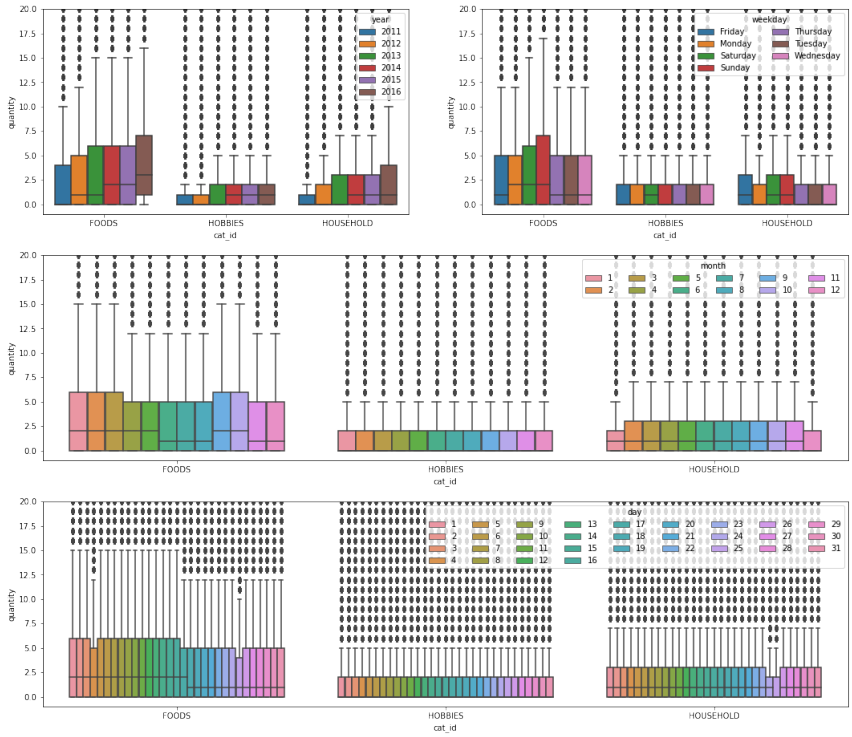


Figure A.15: M5 Level 11 - Distribution of demand for products from different categories by Day of the Week, Day of the Month, Month, and Year.

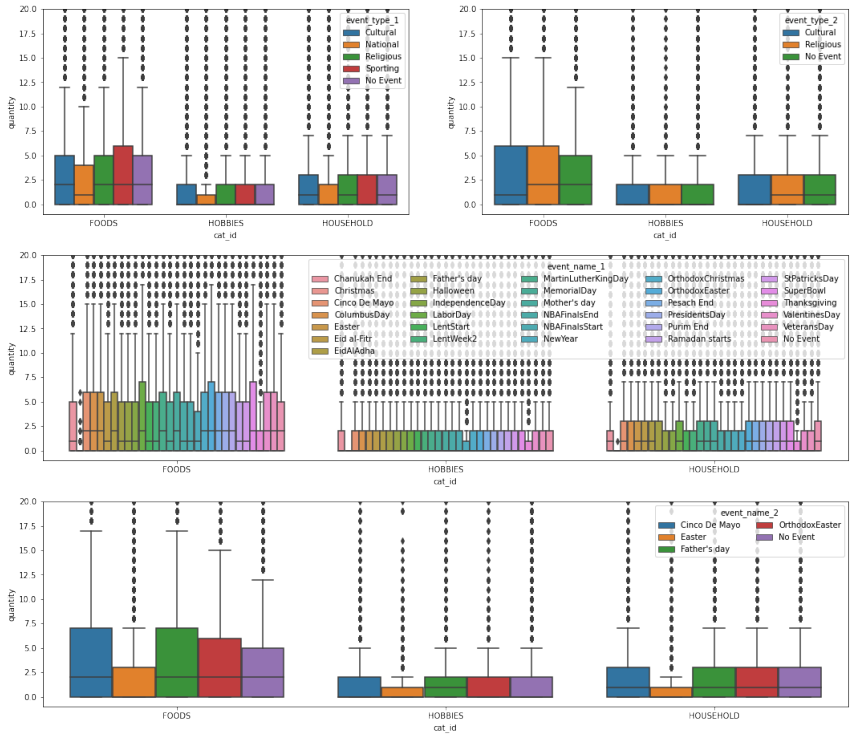


Figure A.16: M5 Level 11 - Distribution of demand for products from different categories by type of Event and name of the Event.

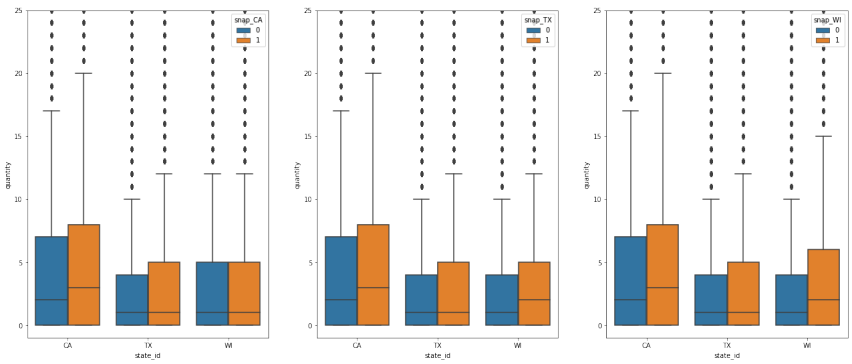


Figure A.17: M5 Level 11 - Distribution of demand for products in the FOODS category by snap_CA, snap_TX, and snap_W

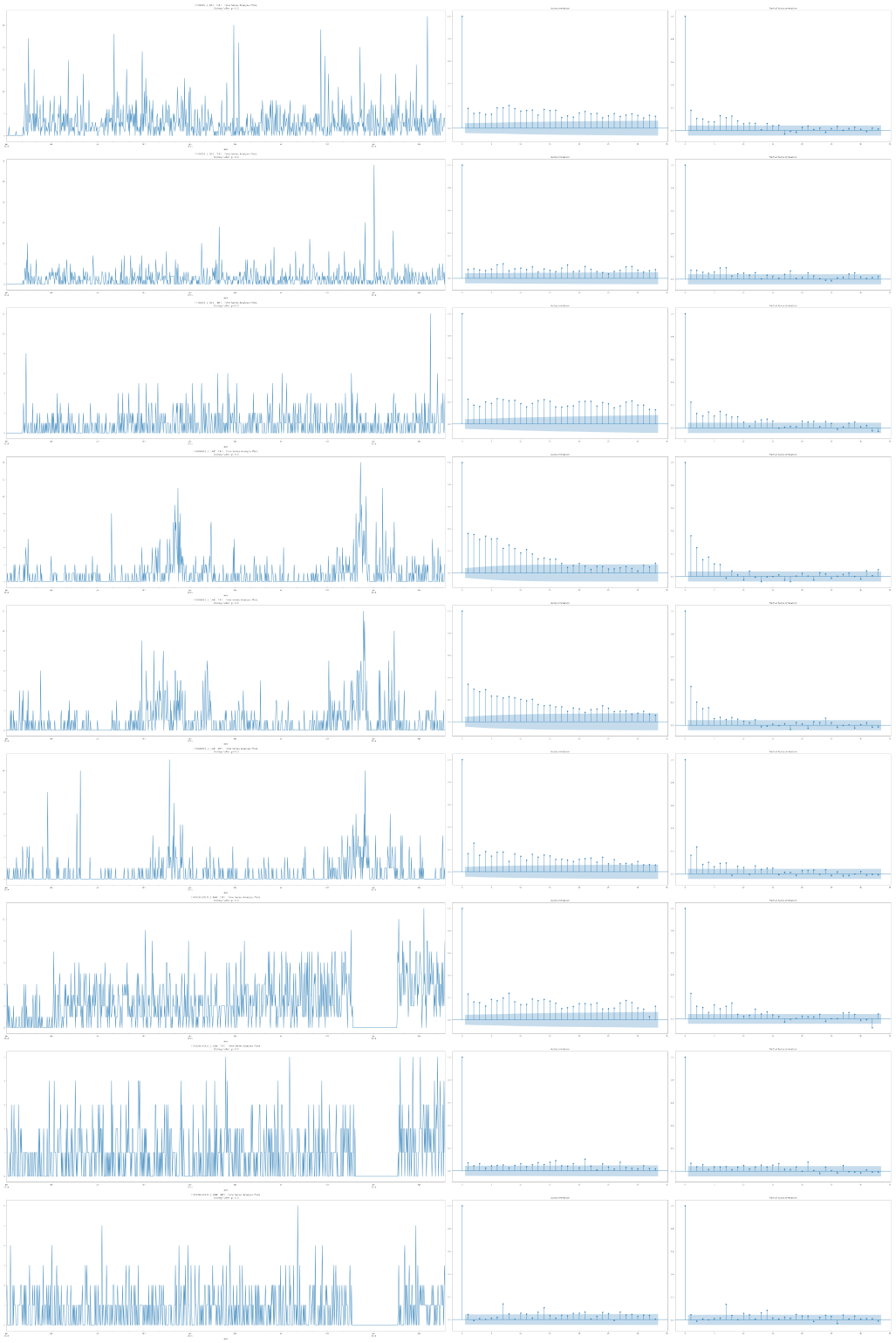


Figure A.18: M5 Level 11 - Examples of dataset Time-Series, with respective partial (PACF) and complete (ACF) auto-correlation values.

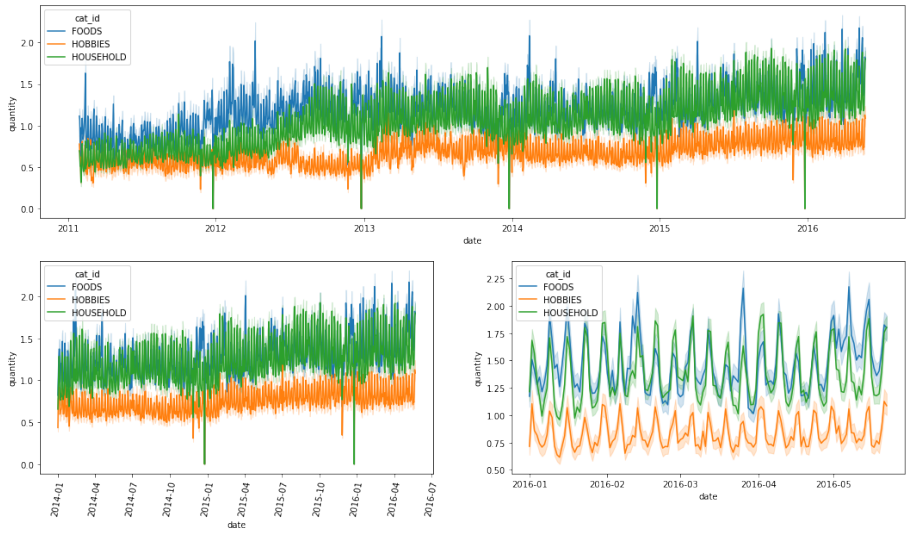


Figure A.19: M5 Level 12 - Expected value and confidence interval of demand for products, by category, over time.

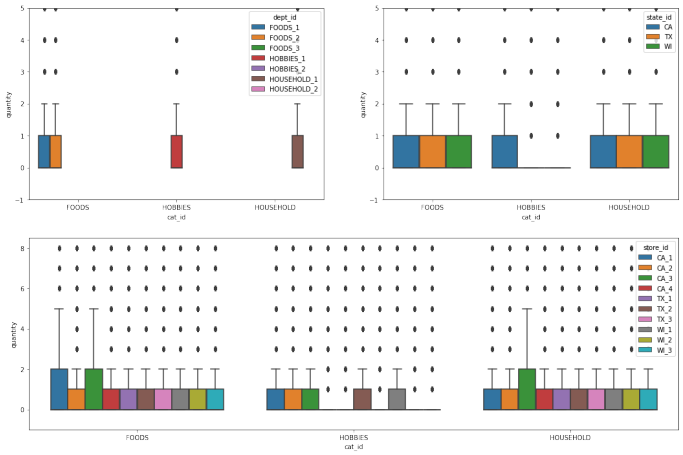


Figure A.20: M5 Level 12 - Distribution of demand for products from different categories by Department, Store, and State.

Figure A.21: M5 Level 12 - Prices versus Demand of products on different days.

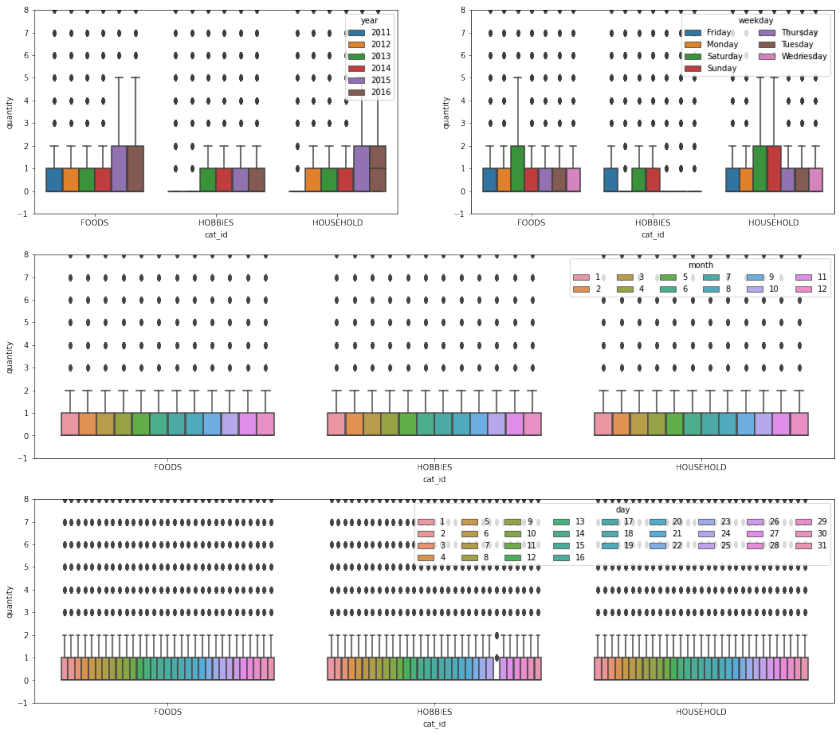


Figure A.22: M5 Level 12 - Distribution of demand for products from different categories by Day of the Week, Day of the Month, Month, and Year.

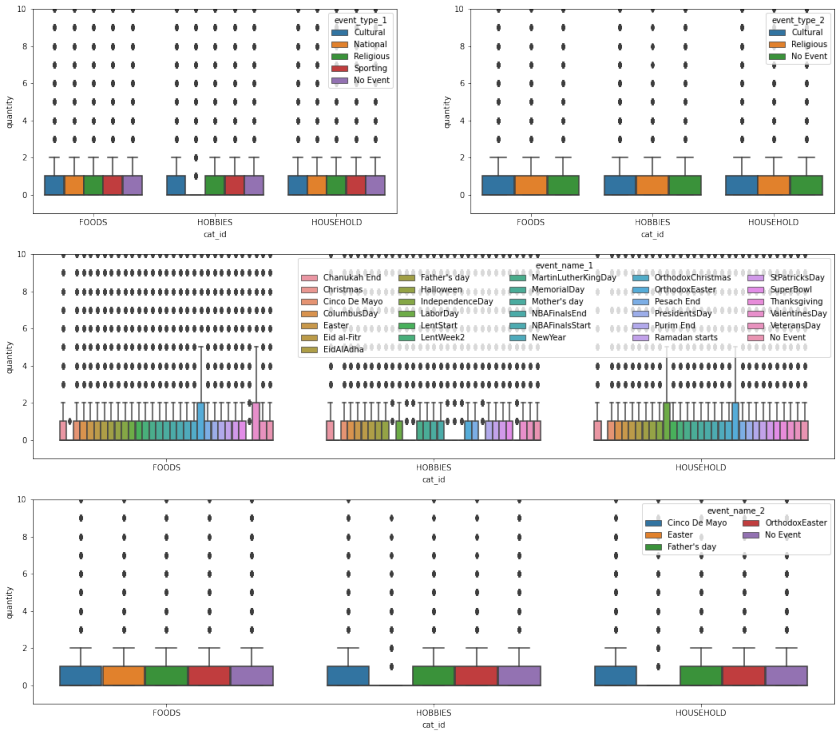


Figure A.23: M5 Level 12 - Distribution of demand for products from different categories by type of Event and name of the Event.

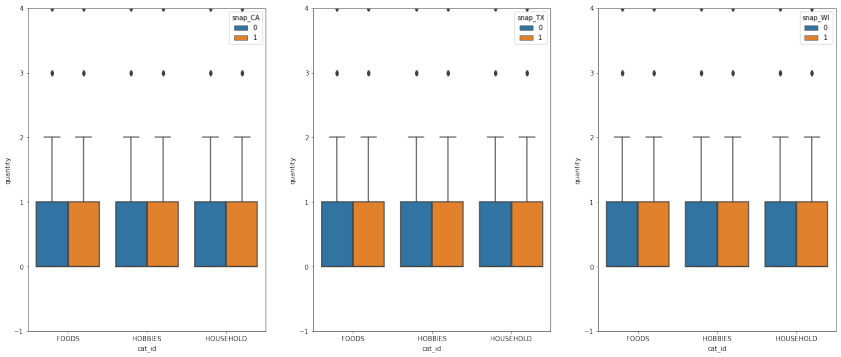


Figure A.24: M5 Level 12 - Distribution of demand for products from different categories by snap_CA, snap_TX, and snap_WI.

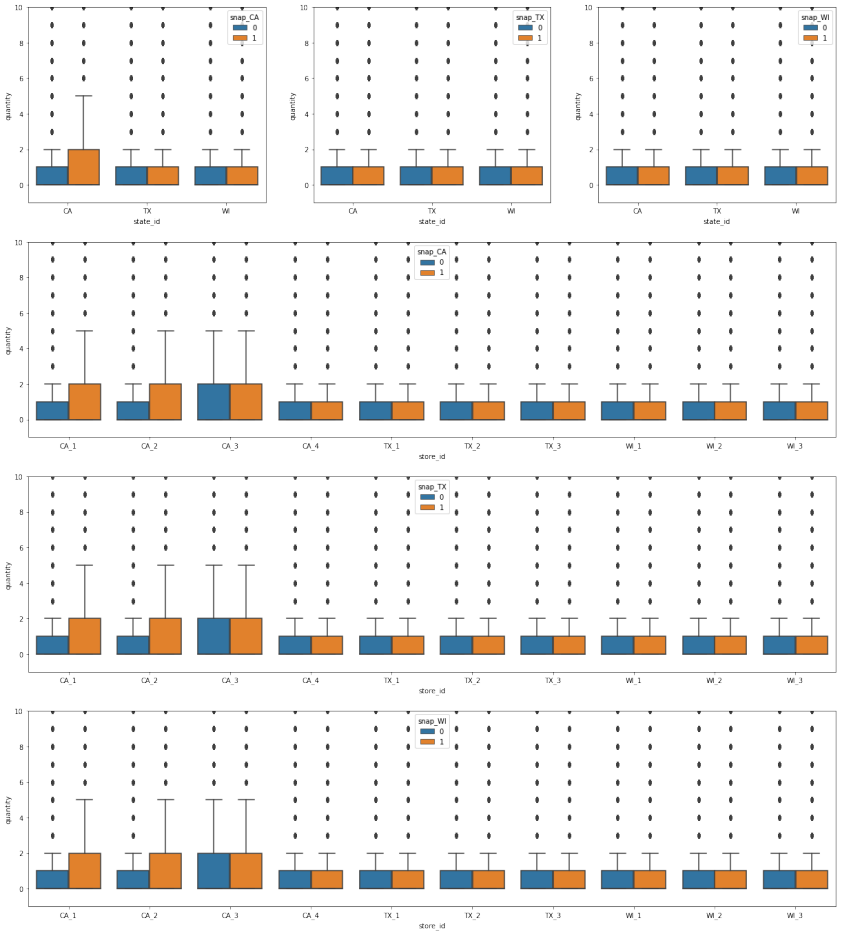


Figure A.25: M5 Level 12 - Distribution of demand for products in the FOODS category by snap_CA, snap_TX, and snap_W.

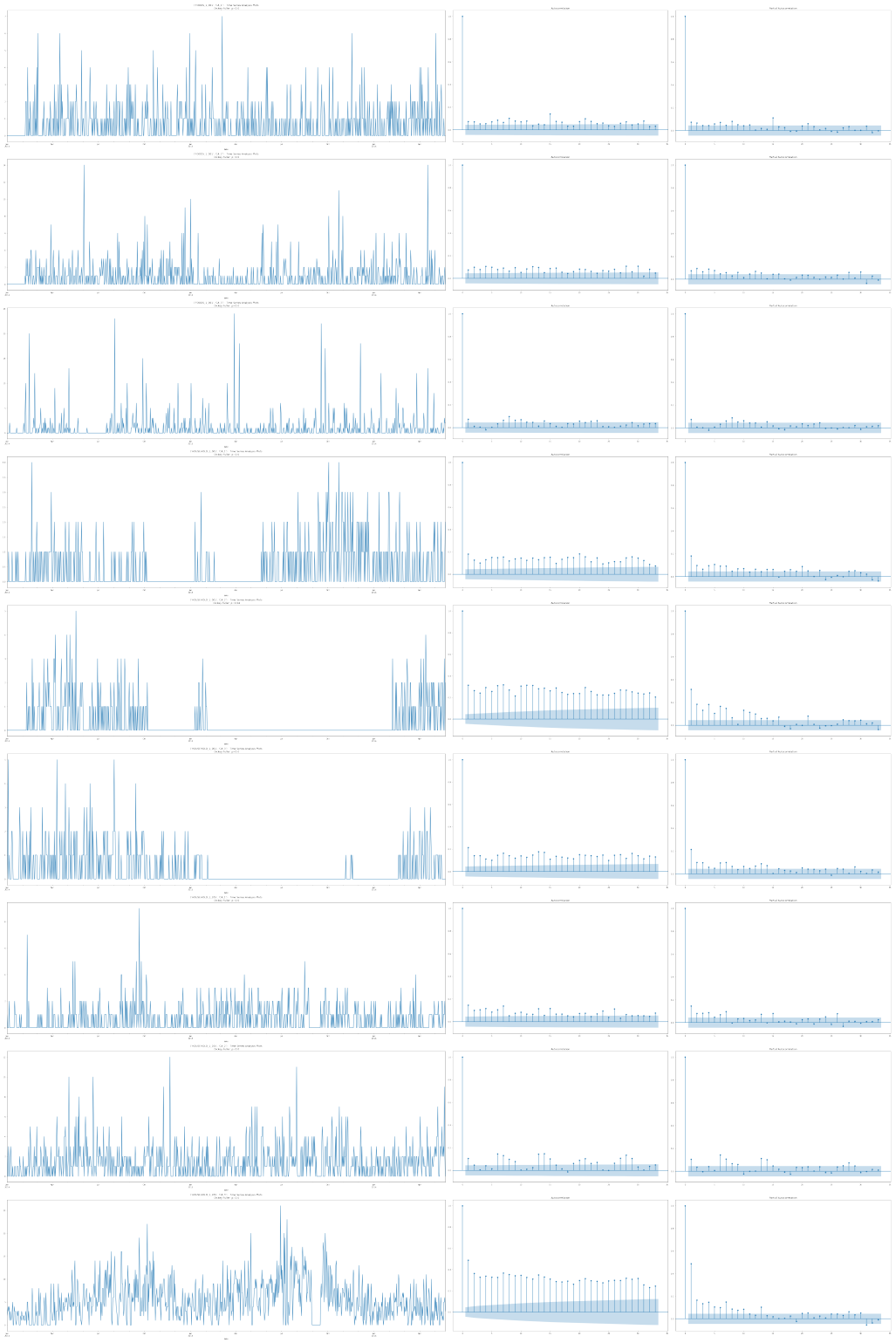


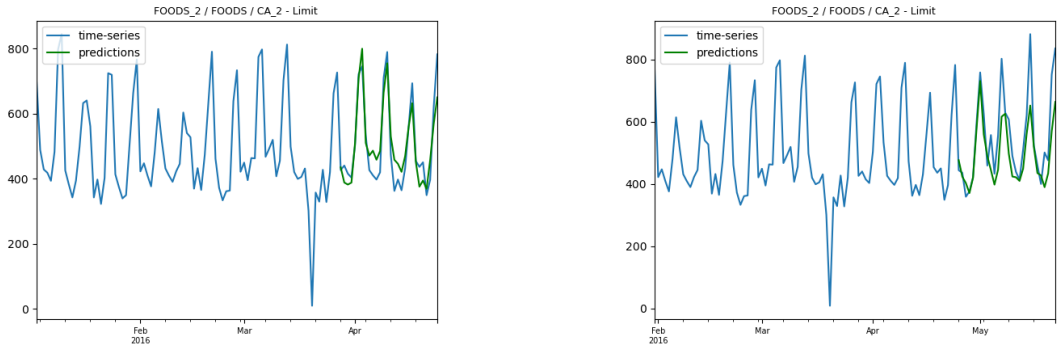
Figure A.26: M5 Level 12 - Examples of dataset Time-Series, with respective partial (PACF) and complete (ACF) auto-correlation values.

Appendix B

Results Plots and Tables

B.1 *Results Tables*

B.2 *Results Plots*



(a) Limit 2016-03-27

(b) Limit 2016-04-24

Figure B.1: **M5 Level 09 - Random Forest** - MIN-Q1 forecastig example.

Group		Random Forest				XGBoost			
		MAE	**RMSE**	**MASE**	**MAPE**	**MAE**	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	9,8675	12,2865	0,8227	0,1237	10,8224	14,2355	0,8642	0,1302
	50%	14,3516	17,4343	1,0001	0,2322	15,2822	18,1597	1,0175	0,2658
	75%	19,7499	22,5209	1,2102	0,3325	19,8702	25,0552	1,2932	0,3607
	Mean	17,6837	22,7334	1,0412	0,2886	18,531	24,0073	1,1146	0,3117
	Std	14,1303	21,8837	0,3289	0,3308	14,6759	22,8186	0,3461	0,3555
Q1-Q2	**25%**	34,6682	44,6466	0,6729	0,1189	34,2154	45,6559	0,6763	0,111
	50%	43,2274	53,7136	0,8753	0,1475	45,0411	59,0344	0,8884	0,1484
	75%	54,6438	68,8161	1,2278	0,1927	59,9353	73,6808	1,4689	0,1911
	Mean	45,6771	57,5045	0,9893	0,1613	49,4563	62,0265	1,0819	0,1749
	Std	16,0438	19,2892	0,3999	0,0646	20,4021	22,6924	0,5277	0,0897
Q2-Q3	**25%**	46,3342	58,2324	0,6014	0,1004	49,3501	62,9511	0,6108	0,1034
	50%	54,9173	72,736	0,6988	0,1199	57,0604	71,9657	0,7062	0,1217
	75%	69,7775	89,2391	0,9485	0,1528	70,4562	84,9194	0,8943	0,1429
	Mean	65,7266	83,1917	0,8056	0,1323	64,3952	81,4167	0,8076	0,1336
	Std	38,2997	45,2695	0,3195	0,046	28,3466	35,841	0,3193	0,0587
Q3-MAX	**25%**	74,9364	99,4664	0,528	0,0726	75,0762	99,079	0,5565	0,0745
	50%	116,492	153,6093	0,6057	0,0846	126,7618	165,2752	0,6167	0,0873
	75%	166,2476	210,5113	0,6875	0,0974	181,8996	229,7764	0,7307	0,0945
	Mean	126,9544	163,1054	0,6176	0,0867	139,5592	179,622	0,6641	0,0924
	Std	62,2247	81,792	0,1573	0,0234	74,4936	95,5819	0,1791	0,028

Table B.1: **M5 Level 9 - Decision Trees -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX - aggregates forecasts of all 4 iterations of the Walk-Forward Validation.

Group		Random Forest				XGBoost			
		MAE	**RMSE**	**MASE**	**MAPE**	**MAE**	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	1,7534	2,1429	0,7251	0,5922	1,6037	2,0627	0,7068	0,5376
	50%	2,1106	2,6546	0,9336	0,9545	2,0188	2,4976	0,8634	0,9405
	75%	3,0416	3,7963	1,0655	6,20532E+14	3,0247	3,8248	1,0657	5,61254E+14
	Mean	2,3451	2,9369	0,9467	4,36079E+14	2,2969	2,904	0,9277	3,97691E+14
	Std	0,802	0,9798	0,2847	6,9047E+14	0,8424	1,0697	0,308	6,41704E+14
Q1-Q2	**25%**	2,7113	3,3474	0,7034	0,4086	2,5213	3,1958	0,6675	0,3652
	50%	3,12	3,8254	0,8361	0,6131	3,0201	3,8302	0,8018	0,5562
	75%	3,8211	4,6749	0,946	1,1758	3,6516	4,5232	0,9444	1,0088
	Mean	3,314	4,1029	0,8587	4,04187E+14	3,1939	3,9934	0,8299	3,5752E+14
	Std	0,8579	1,0026	0,1879	2,15548E+15	0,8635	1,0302	0,2069	1,88967E+15
Q2-Q3	**25%**	3,7056	4,6174	0,8015	0,3485	3,2992	4,1089	0,7318	0,3421
	50%	4,5791	5,8573	0,9583	0,4447	4,3113	5,7037	0,9473	0,4061
	75%	6,5371	8,2787	1,2171	0,6129	7,0037	9,0789	1,1771	0,6004
	Mean	5,3597	6,6969	1,0051	2,66884E+13	5,3962	6,8274	1,0046	2,67538E+13
	Std	2,4367	3,0941	0,2994	1,99718E+14	2,7137	3,419	0,3427	2,00207E+14
Q3-MAX	**25%**	7,9858	9,2818	0,7647	0,2705	7,1132	9,4229	0,7707	0,2886
	50%	11,9862	15,1788	0,9152	0,3989	12,1146	16,1635	0,983	0,3988
	75%	15,875	19,7949	1,1339	0,4938	16,3561	21,85	1,265	0,5443
	Mean	14,3834	17,424	1,1436	5,36907E+15	15,0284	18,2748	1,1921	5,35729E+15
	Std	14,334	15,4465	0,9301	3,0327E+16	14,4307	15,4947	0,9264	3,16629E+16

Table B.2: **M5 Level 10 - Decision Trees -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX - aggregates forecasts of all 4 iterations of the Walk-Forward Validation.

Group		Random Forest				XGBoost			
		MAE	**RMSE**	**MASE**	**MAPE**	**MAE**	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	2,4077	2,8823	0,8102	5,53943E+14	2,1231	2,5039	0,6965	4,73835E+14
	50%	2,8534	3,3091	0,9525	2,18151E+15	2,4555	2,9748	0,7833	2,00785E+15
	75%	3,3636	4,8123	1,1109	3,634E+15	3,2925	4,4837	1,1131	2,86099E+15
	Mean	3,1766	3,9997	1,0325	2,57268E+15	2,9684	3,8346	0,9602	2,17473E+15
	Std	1,1482	1,6552	0,296	2,77891E+15	1,4387	1,9936	0,3873	2,44437E+15
Q1-Q2	**25%**	2,85	3,4002	0,8061	0,7522	2,7483	3,3655	0,7068	0,6312
	50%	4,0737	4,9032	0,9136	1,1715	3,7718	4,5471	0,8714	1,0399
	75%	5,4612	6,5856	1,1146	1,50517E+15	5,0471	6,3736	0,9795	1,13435E+15
	Mean	4,2061	5,1509	0,9638	9,92392E+14	3,814	4,9066	0,8916	7,99636E+14
	Std	1,5234	1,8792	0,2069	1,5915E+15	1,2749	1,854	0,2342	1,32964E+15
Q2-Q3	**25%**	4,8601	5,9767	0,7826	0,6255	4,4027	5,5426	0,7278	0,5954
	50%	6,0922	7,602	0,9647	1,7582	5,7703	7,1012	0,8474	1,2027
	75%	7,4873	9,0119	1,1382	1,90985E+15	6,9603	8,9882	1,218	1,53171E+15
	Mean	6,122	7,6356	1,0378	1,9954E+15	5,8301	7,464	1,0037	1,60401E+15
	Std	1,8654	2,3326	0,3506	3,47314E+15	1,812	2,4171	0,3805	3,14395E+15
Q3-MAX	**25%**	6,7159	8,5309	0,7798	0,3623	8,013	10,3237	0,8521	0,3586
	50%	10,9839	13,1429	1,0117	0,8905	11,2355	14,1325	1,0763	0,8104
	75%	13,5991	17,0716	1,4281	5,15501E+13	13,7511	17,5288	1,5111	6,71965E+15
	Mean	11,7368	13,956	1,3585	8,89636E+15	13,2761	15,8949	1,5586	1,63066E+16
	Std	8,3484	9,325	1,1915	2,41279E+16	9,3076	10,0141	1,2954	4,18696E+16

Table B.3: **M5 Level 11 - Decision Trees -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX - aggregates forecasts of all 4 iterations of the Walk-Forward Validation.

Group		DeepAR - Negative Binomial				DeepAR - Poisson			
		MAE	**RMSE**	**MASE**	**MAPE**	**MAE**	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	9,4911	11,7596	0,8152	0,1336	9,8036	12,0032	0,825	0,1285
	50%	13,5714	16,9326	0,983	0,2411	14,1786	17,5992	0,9828	0,2365
	75%	21,2232	24,8148	1,1097	0,3025	20,7321	25,9289	1,1646	0,3264
	Mean	19,1203	24,2065	1,0605	0,2651	18,3309	23,7181	1,049	0,2778
	Std	18,3671	25,5967	0,3954	0,3136	15,835	24,0127	0,3392	0,3184
Q1-Q2	**25%**	34,1786	44,4482	0,741	0,1189	33,8304	44,5284	0,6971	0,1134
	50%	41,6786	52,4225	0,8803	0,1405	39,3214	51,8302	0,8182	0,1414
	75%	50,5982	63,3898	1,0597	0,1669	48,9911	59,2746	1,1111	0,1675
	Mean	43,5326	55,122	0,9308	0,1448	41,9007	54,1251	0,9012	0,1446
	Std	13,7251	16,8993	0,3009	0,036	11,6409	15,7846	0,2753	0,0415
Q2-Q3	**25%**	47,0625	61,7116	0,6511	0,1071	45,3839	58,7245	0,5833	0,1014
	50%	62,1786	79,023	0,822	0,1297	58,0536	74,7457	0,7311	0,1229
	75%	72,4643	95,0575	0,9678	0,1551	71,3929	91,8031	0,9174	0,1413
	Mean	72,5562	91,88	0,8768	0,1342	65,1024	84,5921	0,7958	0,1285
	Std	45,5037	58,2675	0,3516	0,0373	34,5439	46,1366	0,2754	0,0401
Q3-MAX	**25%**	83	105,8405	0,5766	0,0825	76,2143	103,2672	0,5378	0,077
	50%	139,7679	176,5817	0,7192	0,0961	121,6964	161,2782	0,6262	0,0845
	75%	228,3482	277,7746	0,8839	0,1204	189,4196	241,8206	0,7843	0,0977
	Mean	170,9953	208,6361	0,7894	0,1066	141,7069	181,3334	0,6738	0,092
	Std	110,7601	128,2279	0,2913	0,0358	80,7235	100,2961	0,1978	0,0253

Table B.4: **M5 Level 9 - DeepAR -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX - aggregates forecasts of all 4 iterations of the Walk-Forward Validation.

Group		DeepAR - Negative Binomial				DeepAR - Poisson			
		MAE	**RMSE**	**MASE**	**MAPE**	**MAE**	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	1,7589	2,2467	0,7032	0,4298	1,7857	2,2238	0,7044	0,4376
	50%	2,3393	2,9572	0,8682	0,5291	2,3571	2,944	0,8802	0,5499
	75%	3,2143	4,0514	1,1138	0,6775	3,1696	3,9543	1,1042	0,7202
	Mean	2,632	3,3373	0,9668	0,5796	2,5922	3,2833	0,9501	0,6048
	Std	1,1924	1,4894	0,4137	0,2091	1,1324	1,4195	0,3832	0,2285
Q1-Q2	**25%**	2,2143	2,8223	0,6918	0,3527	2,2857	2,91	0,6985	0,3613
	50%	2,8214	3,5126	0,8507	0,4616	2,8571	3,5715	0,8735	0,4969
	75%	3,6071	4,5201	1,0852	0,6144	3,5714	4,6136	1,0786	0,6665
	Mean	3,2655	4,1125	0,9412	0,5388	3,3175	4,1748	0,9565	0,5661
	Std	1,7905	2,2448	0,4236	0,2951	1,8636	2,315	0,4456	0,2933
Q2-Q3	**25%**	3,1786	3,9857	0,7163	0,3188	3,1964	4,0398	0,7263	0,3302
	50%	4,0357	5,1274	0,9051	0,4053	4,1071	5,1323	0,9002	0,4338
	75%	5,4821	6,9418	1,1101	0,5678	5,5357	7,0909	1,1228	0,6101
	Mean	4,8162	6,0545	1,0004	0,5146	4,7876	6,0389	0,9908	0,5296
	Std	3,2908	3,7777	0,5867	0,403	3,0034	3,5796	0,5108	0,3943
Q3-MAX	**25%**	6,3929	8,2531	0,6768	0,254	6,2857	8,0066	0,7069	0,2577
	50%	9,9643	12,8374	0,8386	0,3238	10,2143	12,8793	0,8697	0,3489
	75%	14,0714	17,8306	1,1466	0,4619	14,2143	17,9215	1,1384	0,5297
	Mean	11,7438	14,6469	1,0097	0,5341	11,8463	14,6426	1,0237	0,5261
	Std	9,8448	10,9265	0,6714	1,2563	9,0568	10,5999	0,6185	1,0025

Table B.5: **M5 Level 10 - DeepAR -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX - aggregates forecasts of all 4 iterations of the Walk-Forward Validation.

Group		DeepAR - Negative Binomial				DeepAR - Poisson			
		MAE	**RMSE**	**MASE**	**MAPE**	**MAE**	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	1,7143	2,1905	0,7023	0,4985	1,7054	2,1782	0,7189	0,5094
	50%	2,2143	2,7493	0,8702	0,6106	2,1786	2,6935	0,867	0,6722
	75%	2,8214	3,5843	1,0968	0,809	2,75	3,5184	1,076	0,8292
	Mean	2,4188	3,1582	0,9394	0,6799	2,3921	3,1056	0,9256	0,7184
	Std	1,2709	1,8903	0,4038	0,2949	1,2179	1,7798	0,378	0,3127
Q1-Q2	**25%**	2,2143	2,8475	0,7096	0,4973	2,2054	2,7487	0,7065	0,5216
	50%	3	3,7776	0,875	0,6391	2,8929	3,7635	0,8659	0,6682
	75%	3,9018	5,0633	1,1241	0,8481	3,9643	5,0203	1,1335	0,9082
	Mean	3,2469	4,1996	0,9537	0,7212	3,2621	4,1973	0,9573	0,7714
	Std	1,4293	1,8913	0,3688	0,3427	1,4507	1,9134	0,369	0,438
Q2-Q3	**25%**	2,8304	3,5694	0,6921	0,439	2,8929	3,6852	0,6945	0,4637
	50%	3,8929	4,9657	0,8584	0,6297	3,9464	4,9107	0,8592	0,6587
	75%	5,4196	7,0951	1,1465	0,9105	5,6161	7,0749	1,0953	0,9801
	Mean	4,5454	5,9015	0,944	0,7466	4,5698	5,8832	0,9528	0,7971
	Std	2,8579	3,6991	0,4036	0,4367	2,7285	3,5529	0,4035	0,4921
Q3-MAX	**25%**	5,5714	6,9739	0,7196	0,3801	5,5714	7,0069	0,7205	0,3885
	50%	7,5357	9,6828	0,8944	0,561	7,6429	9,6753	0,8863	0,5934
	75%	10,2143	13,1517	1,1719	1	10,1429	13,137	1,2049	1,0428
	Mean	8,6997	10,8167	1,0854	0,793	8,7133	10,8215	1,0866	0,8676
	Std	5,5196	5,9484	0,7382	0,8448	5,4182	6,1284	0,7235	0,9719

Table B.6: **M5 Level 11 - DeepAR -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX - aggregates forecasts of all 4 iterations of the Walk-Forward Validation.

Group		Prophet			
		MAE	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	9,6064	12,2759	0,6589	0,1213
	50%	14,1489	18,1305	0,7657	0,1892
	75%	19,9927	24,6819	0,9372	0,2514
	Mean	18,6715	23,2486	0,8254	0,187
	Std	19,2428	22,6796	0,283	0,0762
Q1-Q2	**25%**	36,7981	47,4964	0,6556	0,1148
	50%	48,3638	60,8623	0,8536	0,1443
	75%	55,2194	70,486	1,0147	0,1521
	Mean	48,7233	61,5037	0,9191	0,1394
	Std	18,9469	22,3395	0,3541	0,0373
Q2-Q3	**25%**	44,5039	56,3299	0,5439	0,1103
	50%	58,1665	75,5685	0,6537	0,1174
	75%	75,2239	93,3902	0,6926	0,1319
	Mean	66,5492	83,6199	0,6506	0,1195
	Std	36,5827	46,9451	0,1897	0,0269
Q3-MAX	**25%**	77,3422	102,4022	0,453	0,0716
	50%	133,4425	164,9192	0,5132	0,0826
	75%	177,7949	216,4093	0,7159	0,091
	Mean	129,9924	167,4034	0,5821	0,0827
	Std	61,5061	79,7051	0,1763	0,0149

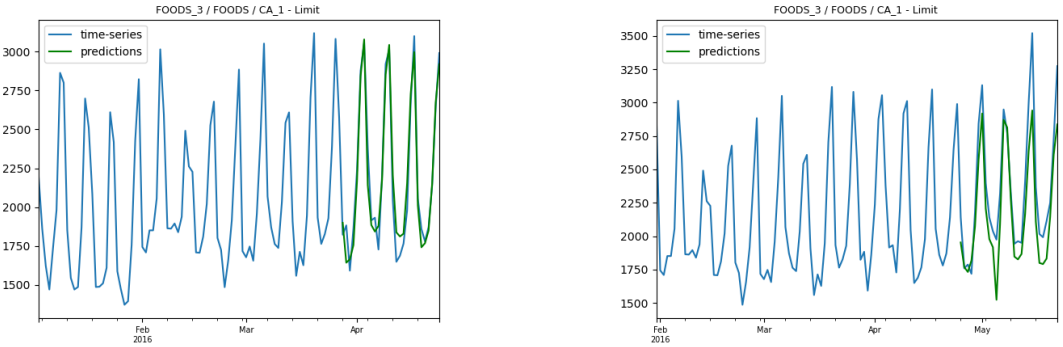
Table B.7: **M5 Level 9 - Prophet -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX

Group		Prophet			
		MAE	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	1,7082	2,2338	0,6531	0,537
	50%	2,0119	2,4659	0,7577	0,7594
	75%	2,8435	3,3342	0,877	6,16794E+14
	Mean	2,3032	2,9359	0,7651	2,77032E+14
	Std	0,8999	1,2325	0,1561	4,00518E+14
Q1-Q2	**25%**	2,5593	3,0929	0,6957	0,4205
	50%	3,1194	3,6972	0,752	0,5295
	75%	3,6361	4,6068	0,8658	0,6314
	Mean	3,1597	3,8424	0,754	5,57642E+13
	Std	0,7976	0,9767	0,1743	2,08651E+14
Q2-Q3	**25%**	3,5843	4,4347	0,6033	0,3418
	50%	3,9963	5,1128	0,6423	0,3777
	75%	4,5045	6,3693	0,81	0,474
	Mean	4,4855	5,7447	0,6964	0,4421
	Std	2,0456	2,5925	0,1316	0,2067
Q3-MAX	**25%**	7,9954	10,9554	0,7157	0,3121
	50%	11,1418	14,2633	0,7987	0,3543
	75%	15,6276	20,0042	0,8839	0,3894
	Mean	17,1969	20,6207	4,1992	2,03454E+14
	Std	22,6302	23,679	12,8354	7,61256E+14

Table B.8: **M5 Level 10 - Prophet -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX

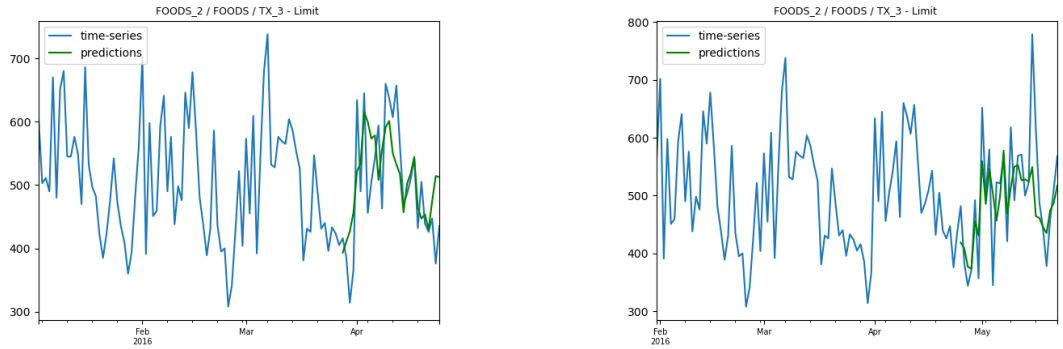
Group		Prophet			
		MAE	**RMSE**	**MASE**	**MAPE**
MIN-Q1	**25%**	1,8345	2,1574	0,504	1,78658E+15
	50%	1,9385	2,3288	0,5253	2,80815E+15
	75%	2,0096	2,3706	0,6978	2,83E+15
	Mean	1,9165	2,2424	0,6262	2,14167E+15
	Std	0,1762	0,2259	0,2126	1,19242E+15
Q1-Q2	**25%**	2,2118	2,9217	0,6144	0,6987
	50%	2,6678	3,4913	0,6674	0,8094
	75%	3,2855	4,1901	0,7023	4,39308E+14
	Mean	2,7756	3,5775	0,6553	2,92872E+14
	Std	1,0778	1,2706	0,0885	5,07269E+14
Q2-Q3	**25%**	4,2916	5,2075	0,6885	0,5422
	50%	5,5458	6,2547	0,6984	0,6246
	75%	5,8804	7,1051	0,8765	8,15999E+14
	Mean	4,9327	6,1235	0,8105	5,43999E+14
	Std	1,6752	1,901	0,2116	9,42234E+14
Q3-MAX	**25%**	19,8501	23,4461	20,9615	1,22093E+15
	50%	21,4582	27,0606	25,9717	1,23359E+15
	75%	26,8244	30,9624	55,2361	2,23973E+15
	Mean	23,9636	27,2522	42,1411	1,89591E+15
	Std	7,304	7,5181	37,0248	1,16916E+15

Table B.9: **M5 Level 11 - Prophet -** Statistics of the errors obtained in the forecasts for the time-series of groups MIN-Q1, Q1-Q2, Q2-Q3, and Q3-MAX



(a) Limit 2016-03-27 (b) Limit 2016-04-24

Figure B.2: **M5 Level 09 - Random Forest** - Q3-MAX forecastig example.



(a) Limit 2016-03-27 (b) Limit 2016-04-24

Figure B.3: **M5 Level 09 - Random Forest** - forecastig volatile sales.

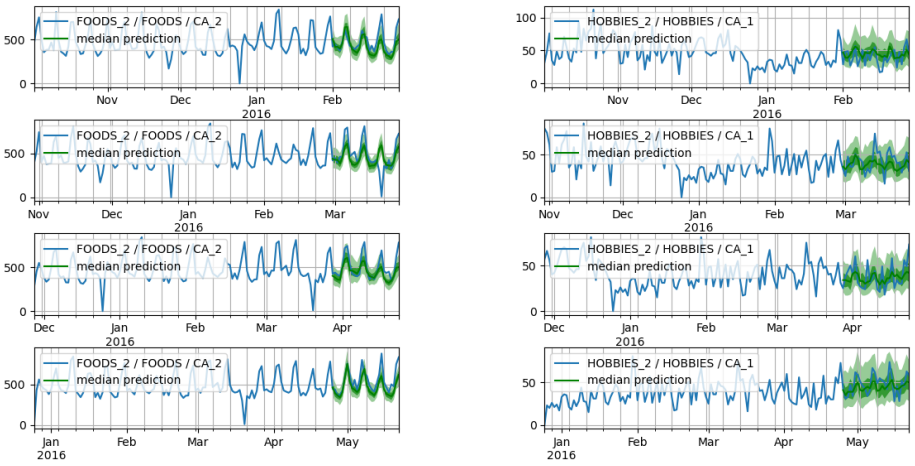


Figure B.4: **M5 Level 9 - DeepAR - Negative Binomial Distribution -** MIN-Q1 forecasting examples

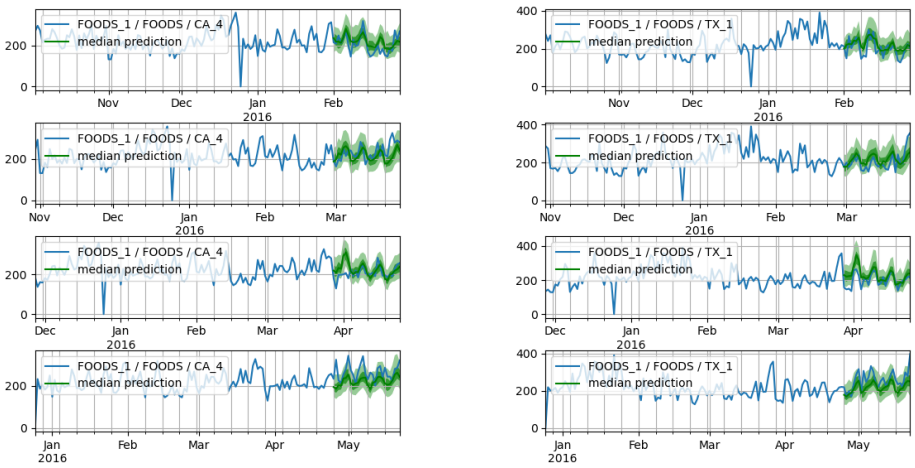


Figure B.5: **M5 Level 9 - DeepAR - Negative Binomial Distribution -** Q1-Q2 forecasting examples

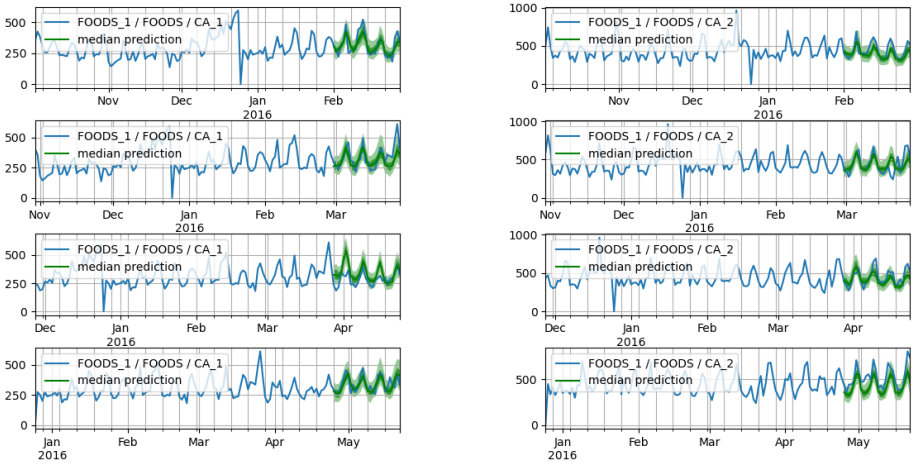


Figure B.6: **M5 Level 9 - DeepAR - Negative Binomial Distribution -** Q2-Q3 forecasting examples

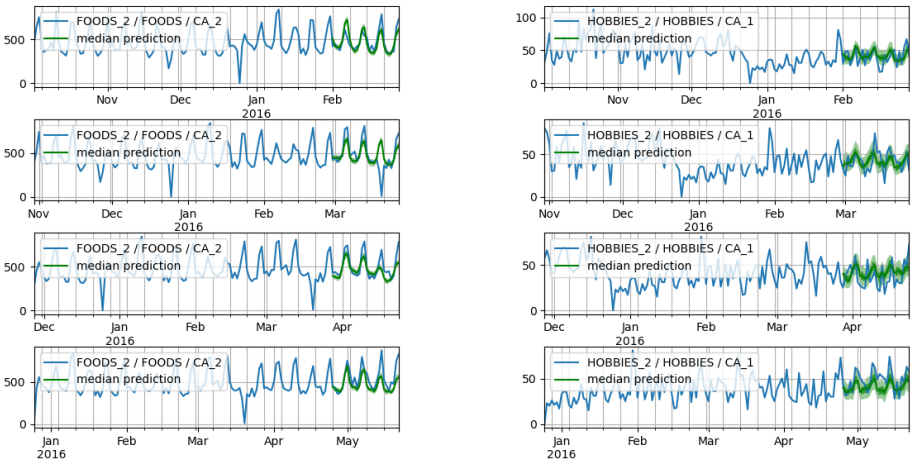


Figure B.7: **M5 Level 9 - DeepAR - Poisson -** MIN-Q1 forecasting examples

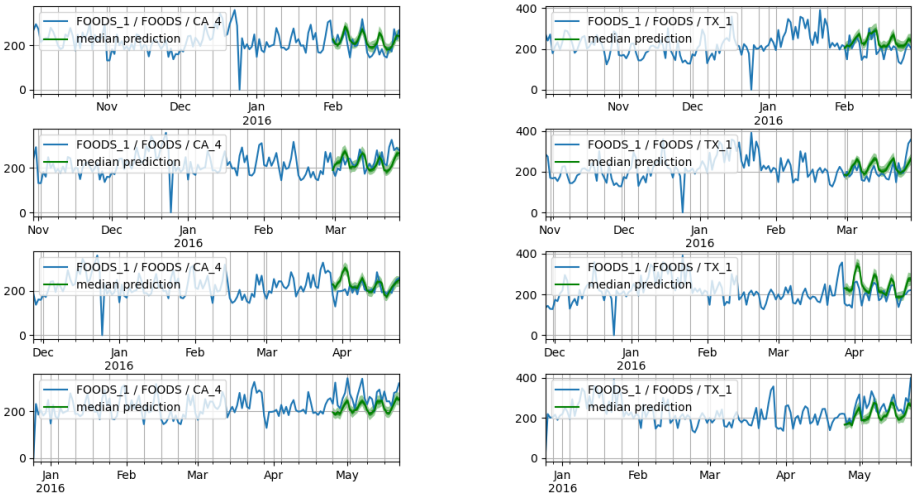


Figure B.8: **M5 Level 9 - DeepAR - Poisson -** Q1-Q2 forecasting examples

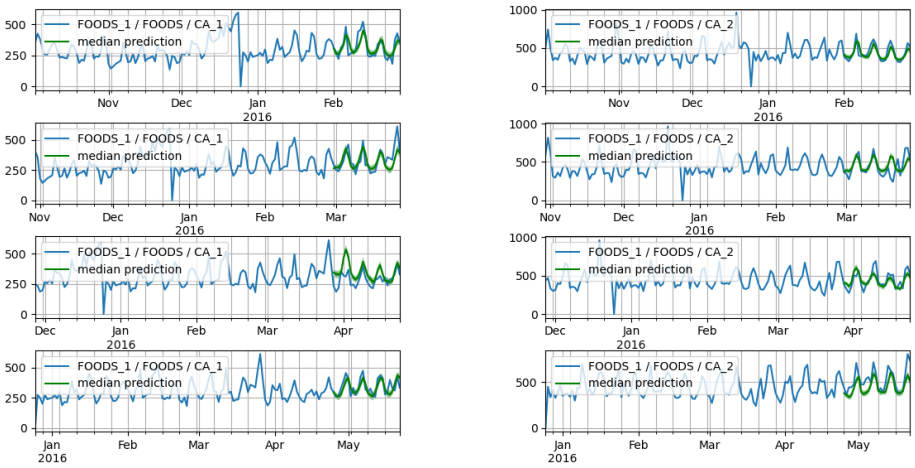


Figure B.9: **M5 Level 9 - DeepAR - Poisson -** Q2-Q3 forecasting examples

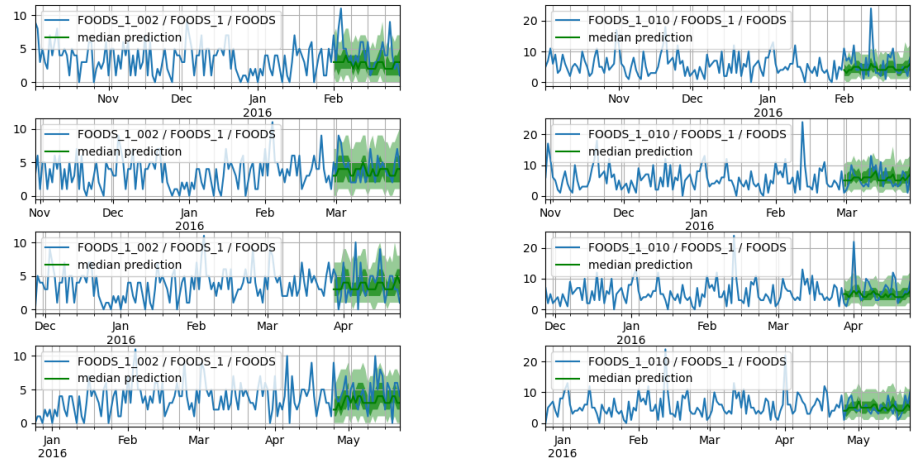


Figure B.10: **M5 Level 10 - DeepAR - Negative Binomial Distribution -** MIN-Q1 forecasting examples

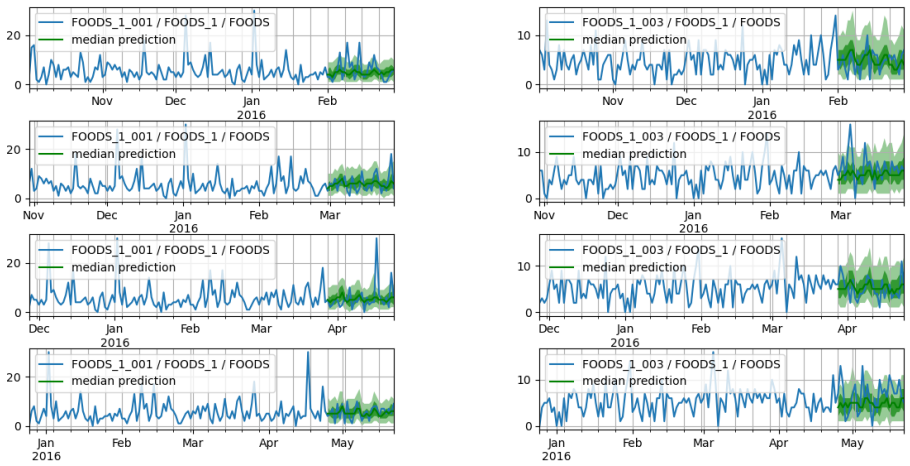


Figure B.11: **M5 Level 10 - DeepAR - Negative Binomial Distribution -** Q1-Q2 forecasting examples

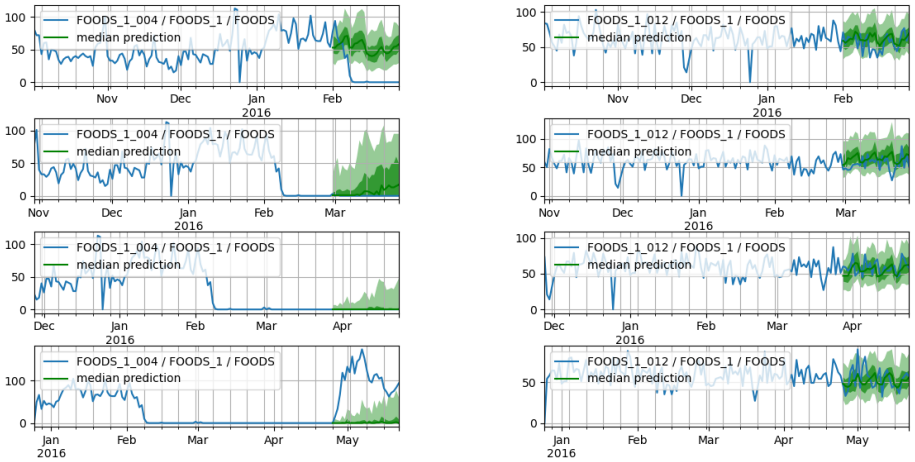


Figure B.12: **M5 Level 10 - DeepAR - Negative Binomial Distribution -** Q3-MAX forecasting examples

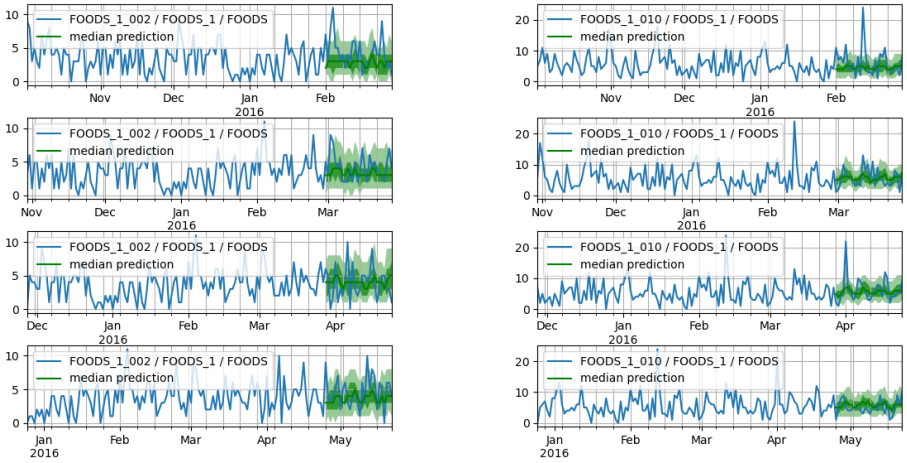


Figure B.13: **M5 Level 10 - DeepAR - Poisson Distribution -** MIN-Q1 forecasting examples

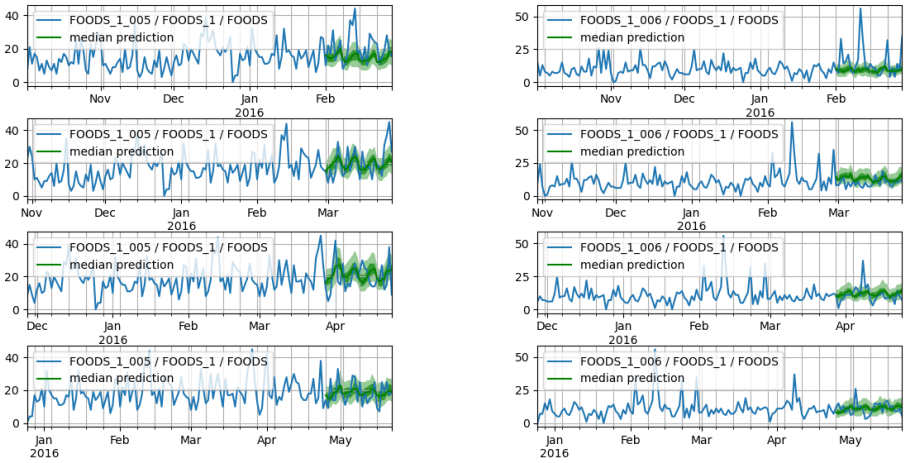


Figure B.14: **M5 Level 10 - DeepAR - Poisson Distribution -** Q2-Q3 forecasting examples

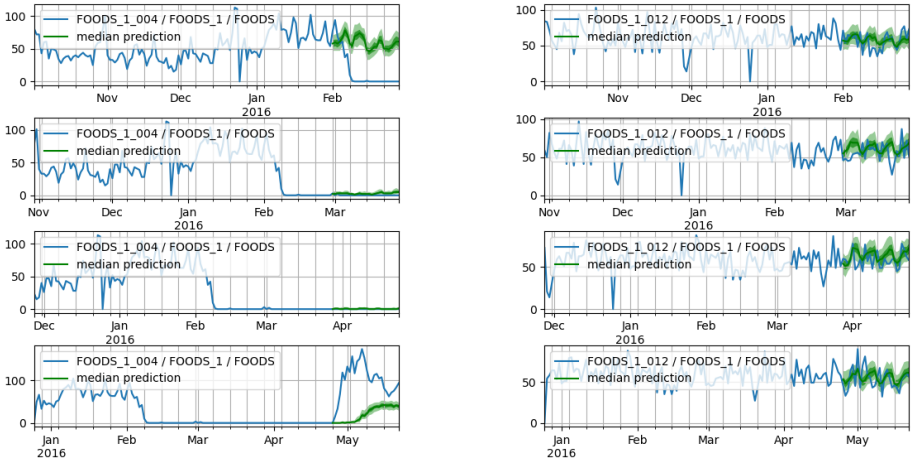


Figure B.15: **M5 Level 10 - DeepAR - Poisson Distribution -** Q3-MAX forecasting examples

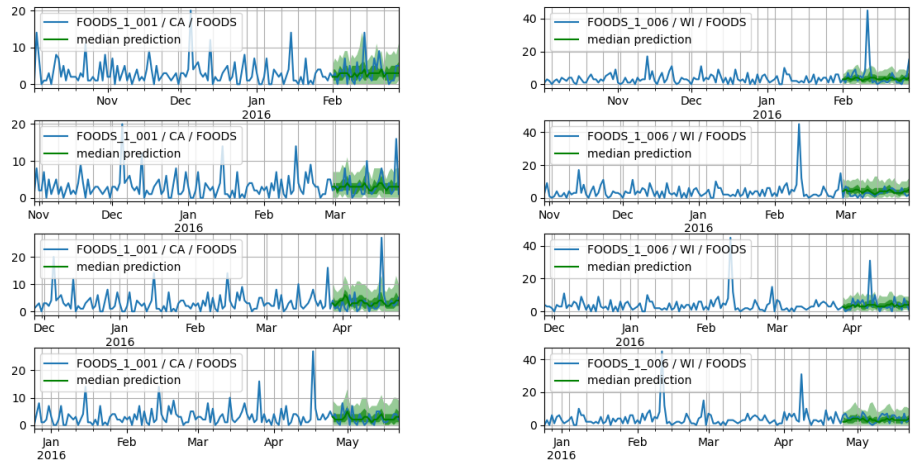


Figure B.16: **M5 Level 11 - DeepAR - Negative Binomial Distribution -** MIN-Q1 forecasting examples

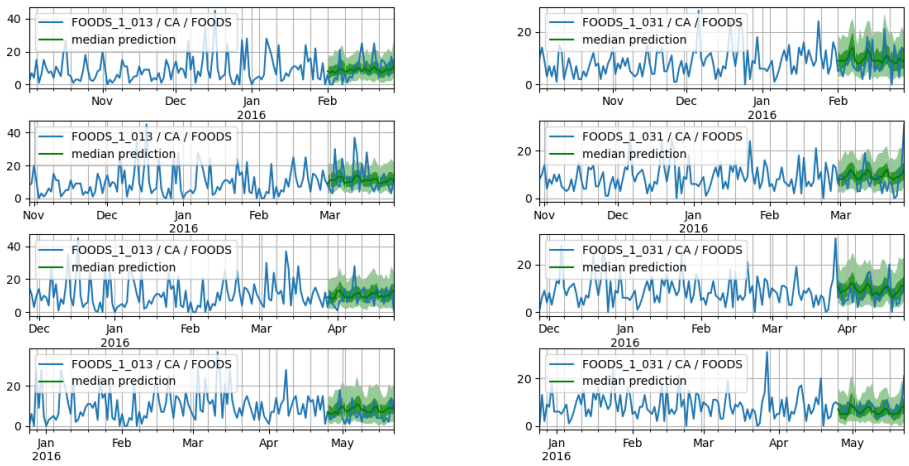


Figure B.17: **M5 Level 11 - DeepAR - Negative Binomial Distribution -** Q2-Q3 forecasting examples

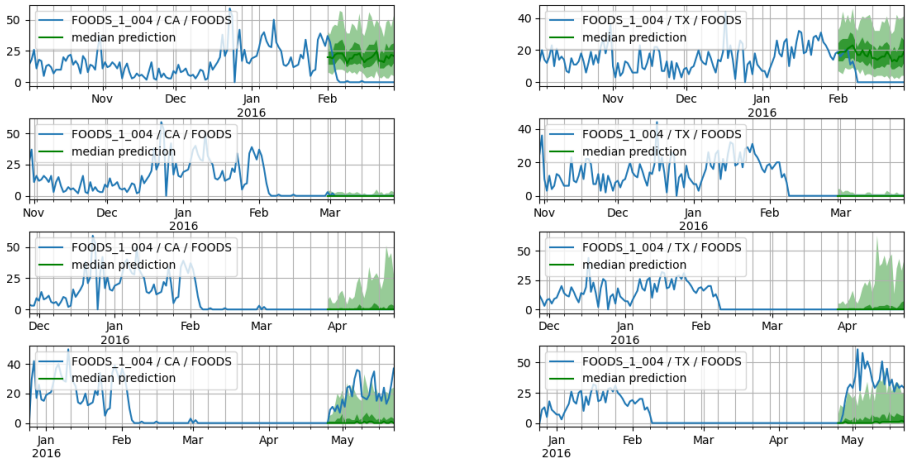


Figure B.18: **M5 Level 11 - DeepAR - Negative Binomial Distribution -** Q3-MAX forecasting examples

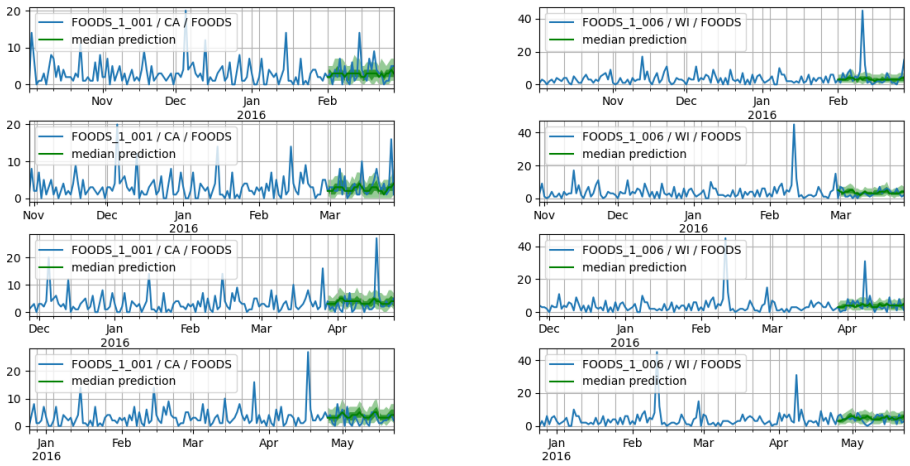


Figure B.19: **M5 Level 11 - DeepAR - Poisson Distribution -** MIN-Q1 forecasting examples

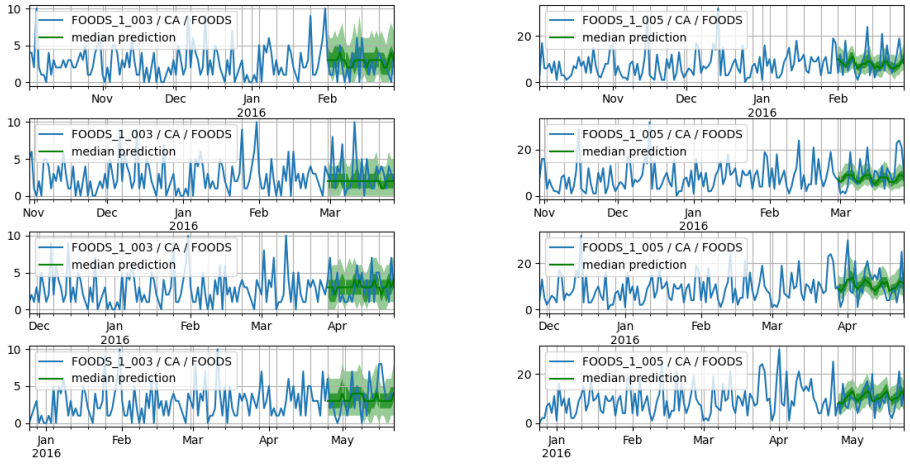


Figure B.20: **M5 Level 11 - DeepAR - Poisson Distribution -** Q1-Q2 forecasting examples

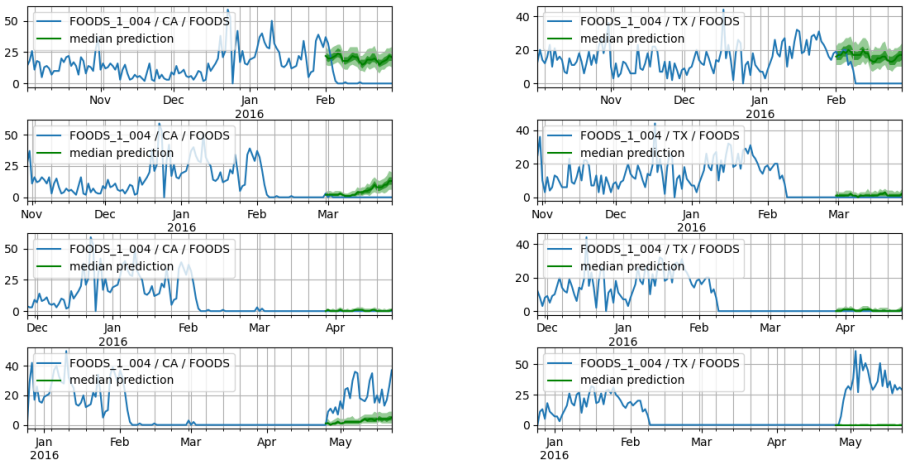
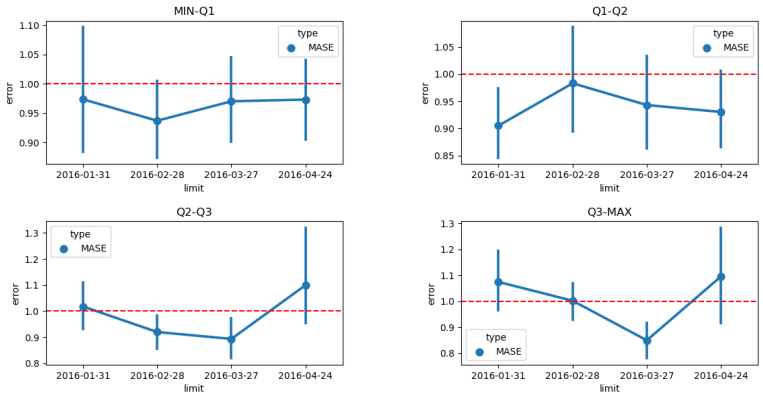
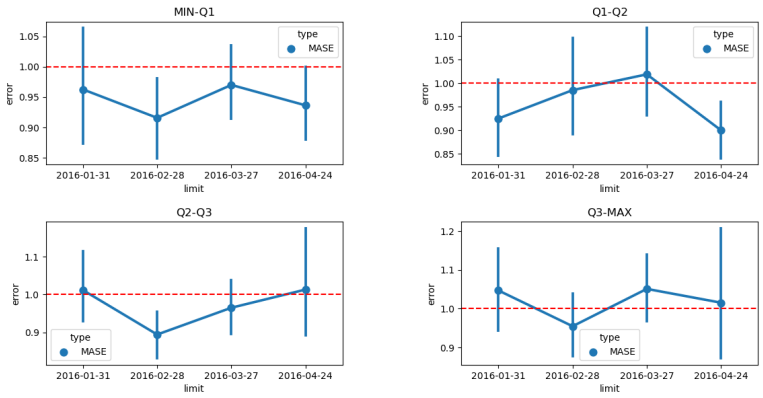


Figure B.21: **M5 Level 11 - DeepAR - Poisson Distribution -** Q3-MAX forecasting examples

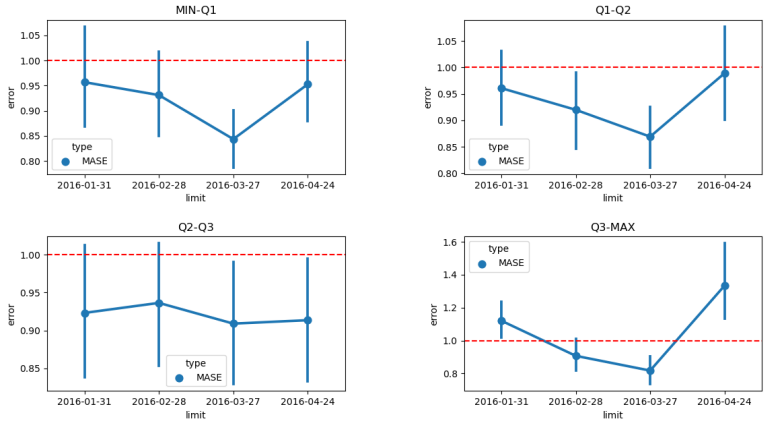


(a) Using the Negative Binomial Distribution.

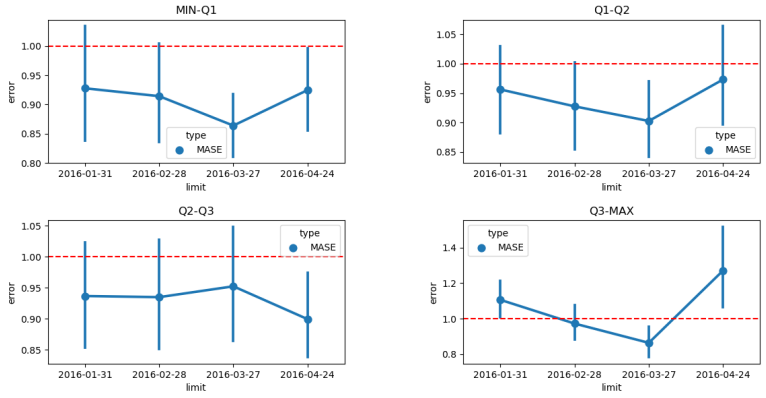


(b) Using the Poisson Distribution.

Figure B.22: M5 Level 10 - DeepAR - Expected value and confidence interval of the MASE errors over training sets with increasing sizes (limit).



(a) Using the Negative Binomial Distribution.



(b) Using the Poisson Distribution.

Figure B.23: **M5 Level 11 - DeepAR** - Expected value and confidence interval of the MASE errors over training sets with increasing sizes (limit).

References

[1] Gluonts - probabilistic time series modeling. gluonts - probabilistic time series modeling - gluonts documentation. Retrieved February 9, 2022, from https://ts.gluon.ai/.

[2] Scikit learn. machine learning in python. Retrieved February 9, 2022, from https://scikit-learn.org/stable/.

[3] Xgboost documentation - xgboost 1.5.2 documentation. Retrieved February 9, 2022, from https://xgboost.readthedocs.io/en/stable/index.html.

[4] Özden Gür Ali, Serpil Sayın, Tom Van Woensel, and Jan Fransoo. Sku demand forecasting in the presence of promotions. *Expert Systems with Applications*, 36(10):12340–12348, 2009.

[5] Ilan Alon, Min Qi, and Robert J Sadowski. Forecasting aggregate retail sales:: a comparison of artificial neural networks and traditional methods. *Journal of retailing and consumer services*, 8(3):147–156, 2001.

[6] Nari Sivanandam Arunraj and Diane Ahrens. A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting. *International Journal of Production Economics*, 170:321–335, 2015.

[7] Nari Sivanandam Arunraj, Diane Ahrens, and Michael Fernandes. Application of sarimax model to forecast daily sales in food retail industry. *International Journal of Operations Research and Information Systems (IJORIS)*, 7(2):1–21, 2016.

[8] Gianluca Bontempi, Souhaib Ben Taieb, and Yann-Aël Le Borgne. Machine learning strategies for time series forecasting. In *European business intelligence summer school*, pages 62–77. Springer, 2012.

[9] Huijing Chen and John E Boylan. Empirical evidence on individual, group and shrinkage seasonal indices. *International Journal of Forecasting*, 24(3):525–534, 2008.

[10] Ching-Wu Chu and Guoqiang Peter Zhang. A comparative study of linear and nonlinear models for aggregate retail sales forecasting. *International Journal of production economics*, 86(3):217–231, 2003.

[11] Claudimar Pereira Da Veiga, Cássia Rita Pereira Da Veiga, Anderson Catapan, Ubiratã Tortato, and Wesley Vieira Da Silva. Demand forecasting in food retail: A comparison between the holt-winters and arima models. *WSEAS transactions on business and economics*, 11(1):608–614, 2014.

[12] Claudimar Pereira da Veiga, Cássia Rita Pereira da Veiga, Weslly Puchalski, Leandro dos Santos Coelho, and Ubiratã Tortato. Demand forecasting based on natural computing approaches applied to the foodstuff retail segment. *Journal of Retailing and Consumer Services*, 31:174–181, 2016.

[13] Gianni Di Pillo, Vittorio Latorre, Stefano Lucidi, and Enrico Procacci. An application of support vector machines to sales forecasting under promotions. *4OR*, 14(3):309–325, 2016.

[14] Gianni Di Pillo, Vittorio Latorre, Stefano Lucidi, Enrico Procacci, et al. An application of learning machines to sales forecasting under promotions. *Control and Management Engineering*, 2013.

[15] Facebook. Prophet - forecasting at scale. Retrieved February 9, 2022, from https://facebook.github.io/prophet/.

[16] Robert Fildes, Shaohui Ma, and Stephan Kolassa. Retail forecasting: Research and practice. *International Journal of Forecasting*, 2019.

[17] Kesten C Green and J Scott Armstrong. Simple versus complex forecasting: The evidence. *Journal of Business Research*, 68(8):1678–1685, 2015.

[18] Rob J Hyndman and Anne B Koehler. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.

[19] Spyros Makridakis, Allan Andersen, Robert Carbone, Robert Fildes, Michele Hibon, Rudolf Lewandowski, Joseph Newton, Emanuel Parzen, and Robert Winkler. The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of forecasting*, 1(2):111–153, 1982.

[20] University of Nicosia. M5 forecasting - accuracy. https://www.kaggle.com/c/m5-forecasting-accuracy/overview.

[21] Jordi Peters. Improving the promotional forecasting accuracy for perishable items at sligro food group bv. *MSc Thesis, Eindhoven University of Technology, Netherlands*, 2012.

[22] Patrícia Ramos, Nicolau Santos, and Rui Rebelo. Performance of state space and arima models for consumer retail sales forecasting. *Robotics and computer-integrated manufacturing*, 34:151–163, 2015.

[23] Mário Santos. Sup.: Alexandra Oliveira; Co-Sup.: Ana Paula Rocha. Analysing the effect of promotions on food products with deeplearning. 2021.

[24] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

[25] Maxim Vladimirovich Shcherbakov, Adriaan Brebels, Nataliya Lvovna Shcherbakova, Anton Pavlovich Tyukov, Timur Alexandrovich Janovsky, Valeriy Anatol'evich Kamaev, et al. A survey of forecast error measures. *World Applied Sciences Journal*, 24(24):171–176, 2013.

[26] Robert van Steenbergen. A synthetic data set with new product demand and characteristics. https://data.mendeley.com/datasets/g3v9xcxjgc/1, Jul 2020.

[27] Souhaib Ben Taieb, Rob J Hyndman, et al. *Recursive and direct multi-step forecasting: the best of both worlds*, volume 19. Citeseer, 2012.

[28] Juan R Trapero, Nikolaos Kourentzes, and Robert Fildes. On the identification of sales forecasting models in the presence of promotions. *Journal of the operational Research Society*, 66(2):299–307, 2015.

[29] Grigorios Tsoumakas. A survey of machine learning techniques for food sales prediction. *Artificial Intelligence Review*, 52(1):441–447, 2019.

[30] RM Van Steenbergen and Martijn RK Mes. Forecasting demand profiles of new products. *Decision support systems*, 139:113401, 2020.