UNIVERSITAT DE BARCELONA

FUNDAMENTAL PRINCIPLES OF DATA SCIENCE MASTER'S THESIS

Open data based electricity load forecasting

Author:
David ÍÑIGUEZ GÓMEZ

Supervisor:
Oriol PUJOL VILA

A thesis submitted in partial fulfillment of the requirements for the degree of MSc in Fundamental Principles of Data Science

in the

Facultat de Matemàtiques i Informàtica

June 30, 2024

UNIVERSITAT DE BARCELONA

Abstract

Facultat de Matemàtiques i Informàtica

MSc

Open data based electricity load forecasting

by David ÍÑIGUEZ GÓMEZ

Electricity is one of the main engines of modern societies. The agents that are involved in the electricity system of a country need to have the best forecasts possible of electricity load in order to ensure that it is correctly supplied, and also to define their action strategies in the market. In this thesis we will focus on the electricity load forecasting for the daily market of the so called Mercado Ibérico de Electricidad (MIBEL), where most of the energy available is auctioned. We studied the State-of-the-Art of the electricity demand approaches, specially for short-term predictions, since we are making one day-ahead estimations. We extracted data from open sources that were later used for designing and testing different types of models. Based on the performance of the different approaches, we selected a model that efficiently combines both time series forecasting and machine learning, obtaining a precision close to the one provided by the system operator, Red Eléctrica. Finally, we analyzed the relevance of each of the variables involved by using the Shapley values and regularization techniques.

Acknowledgements

First of all, I would like to thank my thesis director, Oriol Pujol, for his great advice that has helped me a lot to solve the different difficulties that arose throughout the project. It has been an honor that Oriol has been my director.

I would also like to thank the University of Barcelona for giving me the opportunity to attend this Master, I have learned a lot and after this course I definitely want to become a data scientist for life. Also to each and every one of the Master's professors, I have learned a lot in their lectures and from all the assignments. Their feedback was very useful for me to polish details that will help me be a better professional in the future.

This master's degree would not have been the same without my classmates and all the friends that I have made in it. Friends like Ana, Jon, Sergi, Sergio and Xisco will for sure be for life. But I especially want to thank Jaume because he has been my main support from day one. I am very pleased that you have been my best friend during the master's degree.

Last but not least, I want to thank my family for their unconditional love and support. You have always been there, in the easy moments and in the difficult ones, and I know that the person I am today is thanks to the education you have given me. I can never be grateful enough.

Contents

Chapter 1

Introduction

For centuries, electricity has been the subject of study by many researchers, trying to discover its behavior and properties or how to produce it. Already in ancient Greece they realized that rubbing skins on amber was capable of causing an attraction between both materials. That is, they were already aware of static electricity. In fact, Thales of Miletus described this phenomenon in approximately 600 BC. and the etymology of the word "electricity" itself derives from the Latin *electrum*, which in turn comes from the Greek *élektron*, which means "amber".

Meanwhile, during the 17th century, numerous discoveries related to electricity were developed, such as the invention of an early electrostatic generator, the differentiation between positive and negative currents, and the classification of materials as conductors and insulators. In 1800, already in the Contemporary Age, one of the most significant advances in the study of electricity was carried out by the Italian physicist Alessandro Volta, who discovered that certain chemical reactions could produce electricity and who built the voltaic cell, which produced a constant electrical current, making him the first person to create a constant flow of electrical charge.

Since then, many scientists have developed theories and inventions to try to bring electricity closer to humanity, such as Thomas Alva Edison (inventor of the light bulb), Nikola Tesla, Guiglielmo Marconi, James Watt, André-Marie Ampère or George Ohm.

Today, electricity is an essential part of modern life, as well as an essential aspect of the world's economy. Electricity is used for lighting, heating and cooling, as well as to operate appliances, computers, machinery and devices of all kinds around the world, being a fundamental pillar for contemporary society. Given the importance of electricity in economic and social activities, multiple companies were created to produce electricity and then selling it. This makes it necessary for the purchase and sale of electricity to be carried out in a fair and equal market for all companies, where any of them can offer electricity at a profitable price for them and then to let the law of supply and demand decide if their offer will be taken into account or not.

The fact that electricity is so decisive for the economic development of humanity and that the context in which the purchase and sale takes place is transparent and available to everyone is what has motivated this thesis. Studying the electricity market and the factors that influence it helps to better interpret the state of a country, because as will be seen throughout the project, the economic health of a country and its electricity consumption are highly correlated.

In this project we are going to build a model that forecasts the electricity demand in Spain. In particular, we are going to develop the model for the daily market, which is where the largest volume of energy is traded, and whose price serves as a reference for subsequent markets. We will try different approaches and select the most accurate one, and will also do feature engineering to maximize the accuracy as much as possible. Finally, we will compare our best results with the predictions made by ESIOS, that is, the official demand prediction.

ESIOS is an information system developed by Red Eléctrica. Stands for System Operator's Information System to perform the tasks of information and processes management specifically related to the electricity market. Red Eléctrica as system operator is required to make public the results of the markets or operating system processes. This information is also shown in ESIOS public website. It is worth mentioning that the electricity system is not the same for all areas of Spain. Canary Islands and Balearic Islands have their own systems. In this project, we will focus on the Iberian peninsula electricity system.

Regarding the structure of this report, a contextualization is first carried out, explaining how the electricity market works in Spain, how electricity consumption is calculated, and what factors influence whether it is high or not. Afterwards, the state of the art approaches are discussed, which will be the models on which we will base ourselves to build the final model.

Next, it is explained how the different records that will be used in this project were obtained and how they were organized based on the frequency with which each variable is measured. After that, an exploratory analysis of the demand is carried out, analyzing its dependence on different factors, such as the day of the week or the month of the year in which it is measured or the temperature.

Once all the necessary information has been collected and the dependencies of the electrical demand have been studied, we move on to the experimental section, where two experiments are carried out. The first of them is used to choose the model that will be used later, and in the second one this selected model is fitted on a more refined input. Hyperparameters fine tuning was also performed to maximize the performance of the model.

Finally, attention is paid to the explainability of the models, using Shapley values and L1 regularization, and then comparing both methods. To conclude this project, a review of what was discussed in it is made, and some way to improve the results and future models that could be useful for electricity load forecasting is proposed.

Regarding the results achieved in this project, let us remark that the ESIOS estimates show a percentage error of 1.43%, and the best model developed by us is just below 2%, specifically 1.97%. However, if we remove the outliers we get even closer to the official ESIOS forecast, (1.40% for ESIOS vs. 1.89% for our case). With respect to the model chosen and which was the one with which the commented results were obtained , after having tested linear, machine learning, autoregressive and even some simple deep learning models, finally the option chosen was a combination of time series forecasting with machine learning, implemented in the Skforecast python library.

Chapter 2

State-of-the-Art and background

In this chapter, the background and the state of the art (SOTA) will be explained. In the background section, the functioning of the electricity market and MIBEL (focusing on the MIBEL reports) will be commented, while in the SOTA section different approaches will be explained.

2.1 Background

The cycle of life of electricity is the following:

1. It is produced by different companies in their generation plants (thermal, wind, hydroelectric, solar, etc.).
2. It moves through the transportation and distribution networks, owned by regulated companies.
3. It is sold to end customers through retailers.

This project focuses on the first and third points, i.e. the electricity production and its counterpart the consumption. Regarding the trade of electricity in the first step, it is important to know that there is not only one electricity market, but several interrelated markets that guarantee an efficient use of resources, with very different time horizons. In the long term (months, even years) there are forward markets, that are useful for companies that prefer to reach agreements that are valid for long periods of time, enabling them to plan better their production or consumption.

However, this project is sharpening on short term predictions of the electricity demand. Here there are two main markets, the daily and the intraday market.

The daily market's objective is trading electric energy by presenting buying and selling offers for the 24 hours of the following day. This auction takes place at 12:00 CET. To set the electricity price, each producer offers all its energy available at a profitable price for them (they have to take into account that if they offer their electricity at a high cost, it will probably be dismissed, so their offer has to be balanced with earning as much money as possible). In turn, electricity retailers must request a certain amount of energy to be able to cover the estimated hourly demand that will occur the next day. In the end, the price of electrical energy is that of the last offer from the electricity generating companies with which the demand of the marketers has been covered. That is, the most expensive accepted offer is the one that ends up being established as the price to pay for electricity.

TABLE 2.1: Information about the intraday sessions. D+1 refers to the day after the first session.

Session	Opening time	Closing time	Planning horizon	Time periods
1st	14:00	15:00	24h	1-24 D+1
2nd	17:00	17:50	28h	21-24 and 1-24 D+1
3rd	21:00	21.50	24h	1-24 D+1
4th	1:00	1:50	20h	5-24 D+1
5th	4:00	4:50	17h	8-24 D+1
6th	9:00	9:50	12h	13-24 D+1

Nevertheless, these one-day predictions may make some mistakes and forecast a higher or lower electricity demand than the finally existing. To adjust this, during the day a total of six intraday market sessions take place (they are shown in table 2.1). Intraday markets are an important tool for market agents to adjust, through the presentation of offers for the sale and purchase of energy, their program resulting from the daily market in accordance with the needs they expect in real time. They are carried out once the system operator has made, after the daily market, the necessary adjustments to make the resulting program viable.

2.1.1 MIBEL

The daily market takes place in the context of the MIBEL. MIBEL stands for Mercado Ibérico de Electricidad, which translates to the Iberian Electricity Market. It's a wholesale electricity market that operates in Spain and Portugal, aiming to promote competition and efficiency in the electricity sector of both countries.

Each month, a MIBEL report is published, where detailed information about the operation, performance, and key metrics of the electricity market are provided. These reports typically include data on electricity prices, supply and demand dynamics, generation mix, grid operation and market regulations. They serve as valuable resources for market participants, policymakers, researchers, and the public to understand the functioning and trends within the Iberian electricity market.

For example, the December 2023 MIBEL report summarizes the year in the following items:

- Mibel's electricity demand in the month of December was 24,645 GWh, which represents a variation of 4.6% compared to the same month of the previous year. By markets, the demand increased by 4.1% in the Spanish market and 6.9% in the Portuguese market.
- The economic activity and temperature components had a positive effect on the monthly demand in both Spain and Portugal. The labor component had a negative effect in both countries.
- The monotonous load of the Mibel remained above that of the month of December of the year above 100% of the time. The peak hourly demand for the month was 45,603 MW, a 8.7% higher than that registered in the same month of the previous year.

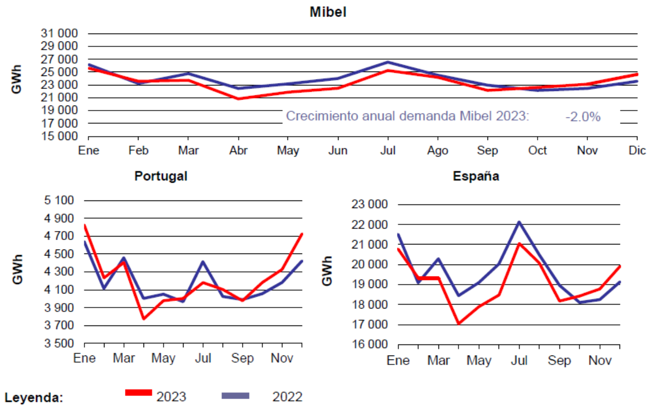

FIGURE 2.1: Electricity demand evolution throughout the year.

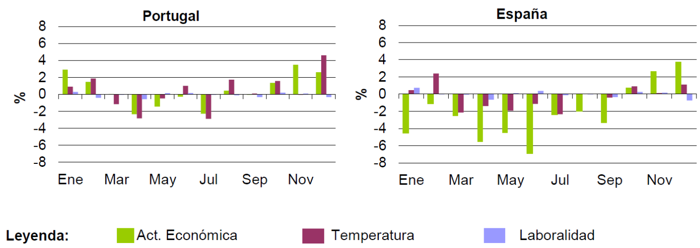

FIGURE 2.2: Influence of economy, holidays and temperature in the demand compared with the same month of the previous year.

The second item is very relevant since it enumerates the factors that have influence in electric load forecasting. Holidays, temperature and economic context are key to predict the daily demand. This three features will be taken into account in the models. However, the electricity price cannot be considered as an input feature, since price depends heavily on demand and, in fact, it is set through the demand-offer mechanism described above.

MIBEL reports also contain very interesting graphics, like the average demand throughout the year (figure 2.1), the influence of economy, holidays and temperature in the demand compared with the same month of the previous year (figure 2.2), or the evolution of demand in the recent years (figure 2.3).

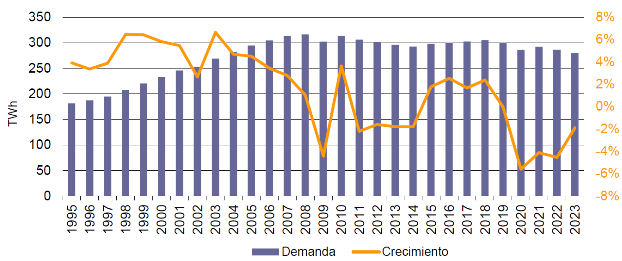

FIGURE 2.3: Total demand evolution and comparison with previous years since 1995.

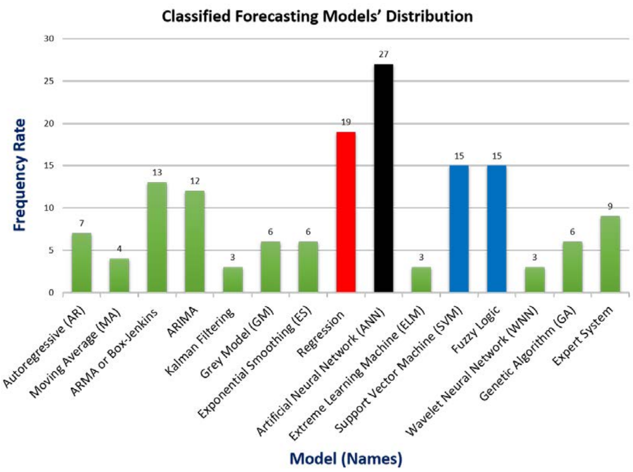

FIGURE 2.4: Distribution of classified forecasting models.

2.2 State of the art approaches

Once the problem to solve has been defined, it is important to know which is the State of the Art of it. The contents of this section are mostly based on the review by (Hammad et al., 2020), which constitutes a very complete and comprehensive review of different approaches to the electricity load forecasting problem. In figure 2.4 it can be seen that there are lots of different ways to solve this problem, and that the most popular ones can be divided in statistical and non statistical approaches. The first ones analyze the structure of time series of the demand and the second ones use more sophisticated algorithms, like neural networks or machine learning.

2.2.1 Statistical approaches

Let us introduce the analytic solutions for the electricity load forecasting problem. As it will be seen in the next section, electricity demand has a lot of short and long term seasonality, so the statistical approaches use this property to try to solve the problem. One common approach involves using typical time series solutions, like autoregressive models or Fourier analysis.

The main idea of autoregressive models (AR) is that the current value of the series can be expressed as a linear combination of past loads, also called lags. A pth-order autoregressive model, AR(p) model is defined as:

$$y_t - \sum_{i=i}^{p} \emptyset_i y_{t-i} = \epsilon_t \tag{2.1}$$

Where $\emptyset_1, \emptyset_2, ..., \emptyset_p$ are the AR coefficients to obtain, and ϵ_t is a random white noise, that is, that has a constant power spectral density.

Another important autoregressive algorithms are the Moving Average (MA) models. It is a linear regression, just as AR, but in this case it regresses the current values against the white noise of one or more past values. Thus, the moving average model of order q "MA(q)" is given by:

$$y_t = \epsilon_t + \sum_{i=i}^{q} \theta_i \epsilon_{t-i} \tag{2.2}$$

Note that since the predictions of the AR and the white noise of MA have certain relationship, it is reasonable to think that there exists a "duality", i.e., invertibility principle between the MA process and the AR(∞) process, that is, the moving average model can be rewritten (inverted) into an autoregressive form (of infinite order).

If we combine AR and MA, we get the Autoregressive Moving Average (ARMA) model. An ARMA (p,q), where p and q refers to the orders of AR and MA, respectively, can be written as follows:

$$y_t - \sum_{i=i}^{p} \varnothing_i y_{t-i} = \epsilon_t + \sum_{i=i}^{q} \theta_i \epsilon_{t-i} \tag{2.3}$$

This model have been a popular choice and extensively applied to load forecasting researches due to their relative simplicity and effectiveness.

However, these three models can only be used when facing stationary time series data. In most cases this does not happens, an so happens with electricity demand. That is why the ARIMA models (ARIMA stands for Autoregressive Integrated Moving Average) were implemented. It includes a differentiating factor, and a lag operator, that operates on an element of a time series to produce the previous element. With all this, the mathematical formulation of the ARIMA(p, d, q) model, using the lag polynomials is given below, where p and q are the AR and MA orders, respectively and d is the number of differences to make the original time series stationary.:

$$\left[1 - \sum_{i=1}^{p} \varnothing_i B^i\right] \left[1 - B\right]^d y_t = \left[1 + \sum_{j=1}^{q} \theta_j B^j\right] \epsilon_t \tag{2.4}$$

This model is the most effective autoregressive model for electricity demand forecasting. One example of its implementation is the one carried out by (Fathin, Widhiyasana, and Syakrani, 2021).

Lastly, to conclude the statistical models, let us introduce the paper presented by (Yukseltan, Yucekaya, and Bilge, 2020), where they use Fourier analysis to forecast electricity demand. Since electricity demand has long and short term seasonality, these frequencies can be used to model the trends of the demand. They build vectors with these frequencies and its variations, and applying matrix multiplications, they get a final prediction. It is relevant that their dataset was not stationary, but had a linear increase, so they added two components to the frequencies vector in order to try to catch this trend (a linear term and an intercept).

2.2.2 Non-statistical approaches

They are mainly solutions that involve either machine learning or deep learning algorithms. Conventional statistical models present difficulties in integrating exogenous variables and capturing complex relationships. In this case, the use of machine learning and artificial intelligence models becomes a more attractive and promising option. As it can be seen in 2.4, some examples widely used are artificial neural networks, support vector machines and fuzzy logic.

Artificial Neural Networks (ANN)

There are different ANN algorithms that can be used for electricity demand forecasting, and the most popular are the feed-forward (FF) neural networks, nonlinear autoregressive with exogenous inputs neural (NARX) neural networks or backpropagation (BP) neural networks. It is worth mentioning that there are two structures of ANN that are valid for the electricity load forecasting problem:

- ANN with one output node. They are used to forecast next hour demand, peak load or total load in a period of time.
- ANN with several output nodes. These are sequences of hourly loads (typically 24, one for each hour of the day). The main objective of these models is to predict the demand distribution for the next day.

As one could expect, the second group has strong robustness and strong learning ability. However, the ANN quickly falls into the local minimum because of the restriction on the generalization ability and cannot make full use of information due to the small sample size.

Support Vector Machines (SVM)

The main objective of the SVMs is to deduct a specific decision rule with a satisfactory generalization ability by choosing some specific subset of training data, called support vectors. In the SVM models, a nonlinear mapping of the input space into a higher dimensional feature space is deployed, and afterward, an optimally separating hyperplane is extracted. Accordingly, the complexity and quality of the SVM solutions do not directly depend on the input space. Initially, the SVMs were developed to deal with pattern classification problems, but later they were extended to be deployed for the regression algorithms as well, named Support Vector Regression (SVR), as it is done in (Fu et al., 2015)

Fuzzy Logic

Fuzzy logic is a generalization of the conventional Boolean theory, but instead of getting a value of 0 or 1 for input, it has associated with it specific qualitative ranges. In other words, for instance, a temperature may be low, medium, or high; however, using fuzzy logic allows outputs to be deduced from noisy or fuzzy inputs and without a need to specify a precise mapping of inputs to outputs. The fuzzy methods are very useful for handling uncertainties and are essential for the knowledge acquisition of human experts. A membership function can be represented for every fuzzy set, where a function for any fuzzy set, or a membership function, exhibits a specific continuous curve that is changing from 0 to 1 or vice versa, while a corresponding transition's region represents a fuzzy boundary of the term. Fuzzy theory is often combined with the other methods to achieve good prediction results.

When combined with ANN, we obtain the so called neuro-fuzzy models, that combine the human-like reasoning style of fuzzy systems with the learning and connectionist structure of neural networks.

2.2.3 Mixing ML and time series approaches

The Skforecast python library (Amat Rodrigo and Escobar Ortiz, 2023) combines both statistical and non-statistical approaches, so it is a useful library for forecasting, since it enables one to use machine learning models combining the lags data (that is, the values of the time series) with exogenous data. Its procedure is the following:

1. Building a matrix with the lags.
2. Adding (or not) exogenous data to the matrix with the lags.
3. Choosing the machine learning models to use.
4. Deciding the type of forecaster in terms of forecasting horizon, recursion...

Figure 2.5 shows a diagram of how the matrix with the lags is built and the model is trained. Once data have been rearranged into the new shape, any regression model can be trained to predict the next value (step) of the series. During model training, every row is considered a separate data instance, where values at lags $1, 2, ...p$ are considered predictors for the target quantity of the time series at time step $p + 1$. By using lags as input features, machine learning models can learn from the past and make predictions about future values.

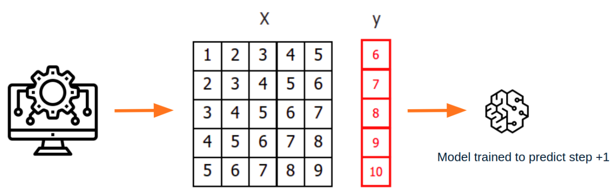

FIGURE 2.5: Diagram of training a machine learning model with time series data.

Once the matrix with the lags is built, one can include extra variables that add extra information about the time series at every moment. These variables are called exogenous variables, and they are really helpful because they provide an essential context that help the machine learning models capture complex relationships and patterns between the different lags and them. Skforecast simply adds these variables in the same matrix that is built with the lags as new columns (see figure 2.6).

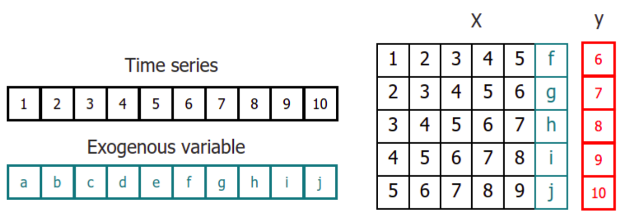

FIGURE 2.6: Time series transformation including an exogenous variable.

When it comes to deciding the machine learning model that will be used here, the models that are compatible with this python library are those who are also compatible with the scikit-learn API (LightGBM, XGBoost, Random Forest, ...), that are algorithms based on decision trees.

After all that, the next step is deciding how we want the future steps of the time series to be predicted. This is done by the forecaster. They differentiate between two types of forecasting: recursive and direct forecasting. The first one computes the new predictions based on the previous ones, and in the second one a model is trained for each step of the forecast horizon, so we have that each prediction is independent. This does not happen in the recursive method, where each prediction is highly dependent on the previous ones. In appendix A one can find some diagrams and of the functioning of these forecasters and a bit more detailed explanation about them.

An implementation of this model can be found at (Amat Rodrigo and Escobar Ortiz, 2024), where they use this library for electricity load forecasting.

2.3 Final conclusion about SOTA

As it can be seen, forecasting the electricity load does not have a unique solution, but just the opposite. There are so many different ways to solve it, so we could not decide a priori which suits the best with the electricity data. As it will be detailed later in the methodology section, a first approach will be made, where most of the models explained here will be considered, and based on the results the best model will be selected, which will be later used for more exhaustive experiments and refining.

Chapter 3

Data and explorative analysis

3.1 Data Extraction

In this part of the report, the process of data collection will be explained, from which web pages they are obtained, and the relevance each category has. All the data that is going to be used in this thesis comes from open sources, so anybody is able to download this information and use them freely. The code is available on Github.

3.1.1 Data collection

The collected data can be divided in 4 different categories:

- Historical records of electricity demand: They will be used to compare our predictions, and also as part of the input of the different models. Real demand and predicted D+1 demand, which is the forecasted demand for the next day were selected. They could be downloaded from ESIOS.
- Cities: They are the final destination of the produced electricity, so knowing the population of each geographical region and the weather in those places is crucial. All province capitals were considered, so the coordinates of all of them were obtained, in order to be able to obtain later the historical data of the weather in those locations. This could be done thanks to the open-meteo API, that given some coordinates, returns different meteorological measures. The temperature, solar radiation, and wind records are the features that will be used in further experiments. The reason these measures were selected will be explained later.
- Economy: It plays an important role on determining the trend in the demand. If the economical context is favorable, then more electricity will be demanded, since there is more business activity. We can say that economy is more important in long term than in short term predictions. However, we downloaded this data because we have the electricity load records of 10 years, which is a long period of time. For this project, CPI, GDP and price historical records were collected. The CPI measures the increase or decrease in prices, and GDP the market value of all the final goods and services produced and rendered in a specific time period by in a country. At the Instituto Nacional de Estadística (INE) one can find information about both GDP and CPI indexes. On the other hand, electricity prices records are available at ESIOS.
- Holiday: Knowing which days of the year are holidays is crucial, because they are special days where there are always less demand than in a work day. In the Social Security database, there is a *.csv* file that contains the most important holidays for each year.

3.1.2 Data Organization in files

When it comes to organizing all these datasets in more comprehensive and intuitive csv files, 4 files were created:

- *demanda_meteo.csv*. It contains both historical records of the weather and electricity demand, as well as the price with a frequency of one hour. In this csv we included the temperature, solar radiation and wind for all capital cities of the provinces, as well as a weighted average based on the population of each city, so we have a single value that represents all 50 provinces.
- *festivos.csv*. Here one can find the holidays of each province, denoted by a binary variable, that is 0 if the day we are studying is a work day in that city and 1 if it is a holiday. Finally, it was added a column that is the weighted average, as it was done with the weather variables, considering the population of each province.
- *IPC.csv*. It includes the CPI of all provinces calculated each month.
- *PIB.csv*. It includes the GDP of Spain every three months.
- *cities.csv*. This file contains information about the 50 province capitals of Spain. In particular, it shows the coordinates of them, and their population throughout the years we are considering. The coordinates of the cities are useful because they will be used as the input of the open-meteo API , and the population information will serve to calculate the different weighted averages explained above.

As it can be seen, the files are basically information related to every province, but the information included in each file is measured with different frequencies. *demanda_meteo.csv* data is obtained every hour, *festivos.csv* contains daily data, *IPC.csv* is calculated each month, *PIB.csv* every three months, and finally *cities.csv* contains static information and data updated every year.

3.2 Demand Exploration

Before starting building different models, let us do some exploration on the real demand, whose prediction is the final objective of this project.

3.2.1 Time evolution

First of all, let us see how the time evolution of the demand looks like. In Figure 3.1, one can appreciate the demand in the first week of 2014, that is, the initial 7 days of the dataset. It is noticeable that for each day two peaks occur in the morning and in the evening, during the hours of greatest business and household activity.

Another plot that is relevant is figure 3.2, that represents the global trend of the demand throughout years. For each day a window is defined, to then compute the mean for this window. We are considering two different windows of 15 days and 180 days at each side, that is, we are averaging 31 and 361 days, respectively. Having a look at the blue curve, it can be appreciated seasonal variations that coincide with the seasons of the year, and a huge decrease in demand when COVID pandemic started. However, before the pandemic the demand trend is ascending, since the

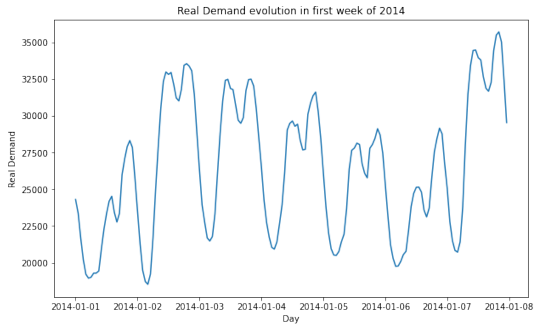

FIGURE 3.1: Real Demand in the first week of 2024.

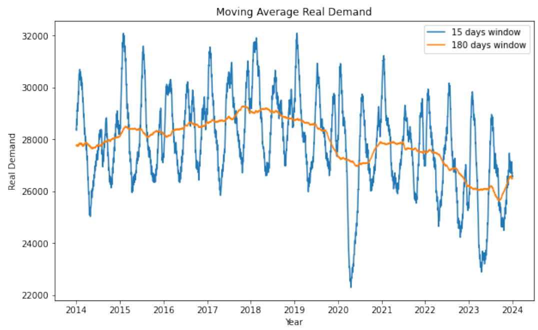

FIGURE 3.2: Moving Average Real Demand.

economy in Spain is recovering from the 2008 crisis. One could expect that after pandemic the demand would come back to pre-COVID values. However, this does not happen, and that is he war in Ukraine caused an exorbitant increase in the price of electricity, causing many companies and households to reduce their electricity consumption (including an increase of self-supply) to be able to meet the payments.

3.2.2 Hourly curve

As it was said in the previous section, the general aspect of the electric demand throughout the day is characterized by two peaks and two valleys. The peaks occur during the hours of greatest business and household activity, that is in the morning and in the evening. On the other hand, at night and at lunch time the demand decreases. However, as one could expect, we do not have the same curve for all days and years, since each of them follows different patterns, depending on the economic context or business activity.

Having a look at figure 3.3, one can appreciate the differences between different days of the week. On Sunday, as it is a holiday, there will be less demand than the other days. On Saturday, we have a small number of companies working all day compared to the midweek, but more than on Sunday, because it is not strictly a holiday. Looking closer to the working days, it is noticeable that Tuesday, Wednesday and Thursday are quite similar, since they are in the middle of the week. Monday are Friday are a bit different, that is because they directly follow and precede weekends, respectively. On Monday morning the demand is lower than other working weekdays because companies are still activating. On the other hand, on Friday we

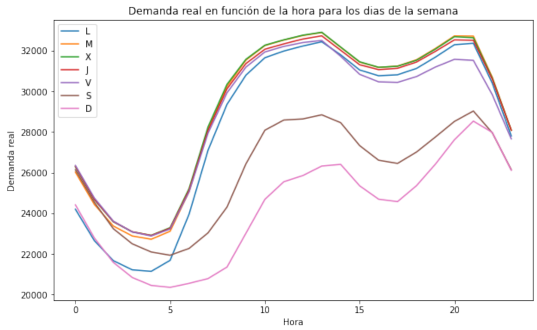

FIGURE 3.3: Hourly Demand based on weekday.

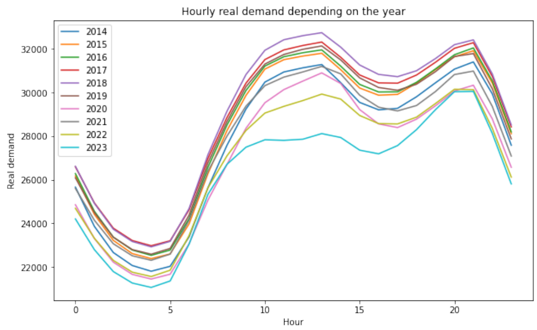

FIGURE 3.4: Hourly Demand based on year.

have the opposite. On the evening, some business close before the end of the day and they start the weekend on Friday afternoon, so there is less activity than usual. We can conclude that routine and festivities greatly influence the daily demand of electricity, especially because of how this affects business activity.

Let us now discuss figure 3.4, where the average hourly demand depending on the year is shown. The economic context is key here to understand the difference between years. This plot is highly related to 3.2, since it visualizes the evolution of the demand considering 6 months windows, that is equivalent to averaging one year. In that plot one can notice that before COVID the demand trend was ascending. This is because economic indexes were better as years past on. However, since COVID demand went down, and one could think that we are still recovering from it. However, having a look at 3.4, it can be seen that that is not the case. Before COVID, the demand is getting higher as long as we are approaching that year, matching with the fact that economy is also getting better and better. However, when COVID pandemic arrived, the national economy almost stopped, and so did the electricity demand. In 2021, the demand was a bit higher than in 2020, this a sign that we are overcoming the situation. But in 2022, the electricity demand was even lower than in 2020. This is because of the huge increase of electricity prices (due to Ukraine war), which forced some companies to buy less electricity. As a solution to it, many of them reduced their activity and others opted by other energy sources like gas or fuel. That is the reason of the low demand in 2023. As it can be seen, the big difference in the demand occurs in the morning, that is when there is more business activity.

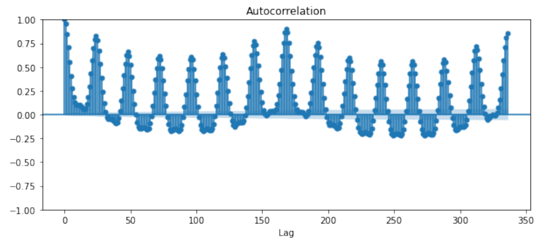

FIGURE 3.5: Autocorrelation plot

3.2.3 Autocorrelation

So far, it has been seen that data has long-term seasonality and short-term stationality, where days have similar behavior. In this section we are going to quantify how similar the demand is with respect to the demand values that precede it. This is determined by the autocorrelation, which measures the linear relationship between an observation and its previous observations at different lags. In figure 3.5 we have the autocorrelation with the previous 336 hours, that is, the last 2 weeks. As it can be appreciated, the lag with better correlation is the one corresponding to exactly the same hour but one week earlier (excluding those lags corresponding to one and two hours before the prediction because in that time the demand does not change that much). This makes sense, since in figure 3.3 the demand is highly dependent on the day of the week, specially on the weekends. This is why this lag is much more important than 24 and 48 hours before. It is noticeable the high decrease in the autocorrelation on the first lags, but once again taking into account figure 3.3, the variation on the demand after 3 hours is important at some points of the day.

3.2.4 Holidays

In figures like 3.3, one can notice that holidays have a great impact on the daily electricity load, since the business activity is highly reduced, making the electricity demand much smaller. To visualize it, a histogram of the energy load for work days and holidays was made. Note that to make the histograms it were only selected those days where it was work or holiday throughout Spain, I did not take into account those that are holidays in some regions but not in others. With all this, we obtain figure 3.6. Notice that there is a notorious difference between the variables. This confirms what was mentioned previously, since clearly when the day is a working day the demand is higher. However, one can think that the demand is similar in some cases, since the graphs are overlapping in some areas. This is because we do not distinguish between day and night. The reason of this overlapping is that in the holidays case those points correspond to the moments with the most energy load (probably in the evening), and to those where the are less electricity demand on work days (in the early morning).

3.2.5 Meteo

Lastly, let us show how does demand vary depending on the weather conditions. The impact of the temperature is high, however, when it comes to forecasting electricity load, previous records of them are also very relevant. Having a look at figure 3.7 (note that the x-axis is the weighted temperature based on population), it is reasonable to think that temperature is a very relevant feature, since we can clearly

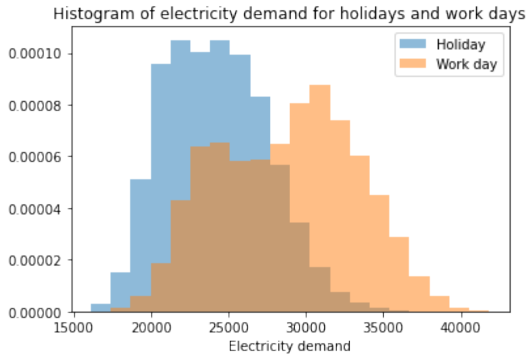

FIGURE 3.6: Histogram of electricity load in work days and when it is a national holiday.

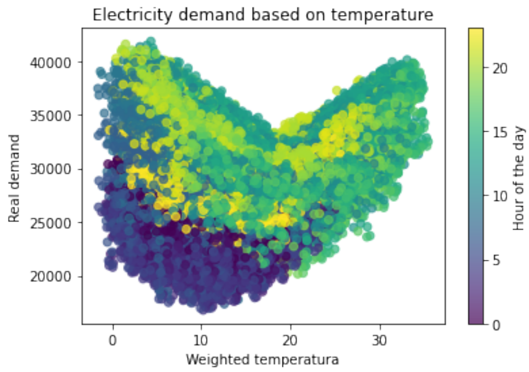

FIGURE 3.7: Scatter plot of the demand based on the temperature for work days

distinguish two straight lines at the top of the plot. This agrees with the idea that in winter and summer we usually have more demand because in those months temperatures are colder and hotter than usual, so people tend to turn on the heating and air conditioning (more information is available in appendix B). This scatter plot confirms that idea, because when the temperature is warm there is less demand. However, it was to be taken into account that this relationship between temperature and demand is not linear at all. In this case, one should try two strategies: use a more sophisticated model, or consider two regions of temperature (cold and hot regions) and fit a linear model considering two temperature features instead of one.

Regarding the radiation and wind data, this information is not as important as the temperature, but they play a crucial role in measuring the electricity self-supply, specially in 2022 and 2023, when the conflict between Russia and Ukraine started. Solar radiation and wind are related with solar and wind energy, respectively. The first one is much more popular than the second one, since it is easier to install solar panels than wind turbines.

Chapter 4

Experimental settings and results

4.1 Methodology and metrics

Once the problem we want to solve has been defined, the state of the art approaches were researched and an exploration to see the trends and how does the data look like was performed, it is time to think about the methodology to follow for the development of the forecasting models and the metrics to use to measure the accuracy of them.

First of all, since we have a lot of different models that are suitable for the electricity load problem, we are going to test most of them. The first experiment will be performed on a simple input that will be explained later. That experiment will help us to decide which model is going to be tested later on a much more complex input. We will also perform hyperparameter tuning on this model, aiming to get even better results.

Note that in all the experiments we are going to split the data in training, validation and test in the same way, in order to be able to compare them fairly. The training set will be always the data corresponding to the years from 2014 to 2021, both included. On the other hand, the validation set will be the year 2022, and finally 2023 will be used for testing. There are some cases where three splits are not required (when we are not performing any hyperparameter tuning nor using a neural network). In that case we will use the validation set (i.e. the year 2022) also for training. By doing this, we ensure that all models and experiments are evaluated on the same set, that is, the year 2023, making it easier to compare the different approaches.

Regarding the metrics we are going to use in this project, these are the root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE) and maximum error (MAX), whose formulas are the following:

$$RMSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 \tag{4.1}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - f(x_i)| \tag{4.2}$$

$$MAPE(\%) = \frac{100}{n}\sum_{i=1}^{n}\frac{|y_i - f(x_i)|}{|y_i|} \tag{4.3}$$

$$MAX = max(|y_i - f(x_i)|) \tag{4.4}$$

Where x_i are the model inputs, y_i the true output values related of the inputs, and $f(x_i)$ are the predictions output obtained by the model. Lastly, n is the number of samples we are going to predict.

4.2 Model selection

As commented in the previous section, we will first perform an experiment where we will test different models, and from it we will decide which model to choose for the second experiment. The models we are going to test are the following:

- Linear model (Linear)
- Extreme Gradient Boosting (XGBoost)
- Autoregressive model (AutoReg)
- Forecaster AutoReg using XGBoost (SkForecast)
- Neural Network with 1 output (NN)
- Neural Network with 24 outputs (NN 24h)

Note that the linear model, XGBoost and the ANN do not need the data to have the structure of a time series, but the AR model and the forecaster do. Before showing the results, let me explain the input features of the model. Note also that the difference between ANN and ANN 24h is the output of the model, because the second model directly predicts the load of the next 24h, while the first one only predicts the electricity demand of one hour.

4.2.1 Input of the experiment

It is worth mentioning that this input will not be used by the AutoReg model and the Forecaster AutoReg model, since they use the demand data as inputs of the model. The features we are including are the following:

- Day of the month.
- Month of the year.
- One-hot encoding of the day of the week. That is, we built 7 variables called 'L', 'M', 'X', 'J', 'V', 'S', 'D', so that the value of the variable is 1 or 0 depending on the weekday (1 if the name and the variable matches with the weekday and 0 if not).
- Holiday. Here we use the weighted average variable that we included in *festivos.csv*.
- CPI and GDP of Spain that day.
- Temperature of tomorrow and 24h before. As we did with the holiday, we use only one value that is the weighted average based on population. We are not considering yet the radiation or wind, because they are much less relevant than temperature. We will use them in the second experiment.

- Electricity demand of 24h before, and also 7 and 14 days before at the same hour. We consider them because as we can see in figure 3.5, these are the most correlated lags.

The last two items are hourly data, and the rest are daily information. This is relevant because for the NN 24h model we will put the daily data only once, and then the temperatures and demands explained above for all 24h of the day making this model much bigger than the other ones. On the other hand, for the autoregressive models (AutoReg and Skforecast) we will use as lags the same hour of the last 14 days. Table 4.1 summarizes the length of of the models input and output. As we can see, NN 24h model is by far the biggest model.

TABLE 4.1: Input and output length of the models we are going to test.

	Linear	**XGBoost**	**AutoReg**	**Skforecast**	**NN**	**NN 24h**
Input length	17	17	14	14	17	132
Output length	1	1	1	1	1	24

4.2.2 Experiment results and discussion

The results of the experiment appear in the table 4.2, where we can see that XGBoost and Skforecast seem to be the best models overall. It is quite surprising that the worst algorithms are both neural networks, performing even worse than the simplest model that is being tested (the linear model). As we can see in table 4.3 (which shows mean μ and standard deviation σ of the errors made when predicting, as well as the percentage of errors found in each interval defined by distances in units of σ to the mean), the NN 24h model is highly biased, since the mean μ is far away from 0, which would mean that the model is unbiased, and the NN model has the greatest standard deviation.

It is relevant the outstanding performance of the linear model as well, even when we have features that do not follow a linear behaviour, like temperatures.

AutoReg has shown good performance, just as expected, since it is one of the most popular time series algorithms, and knowing the short and long term periodic behavior electricity demand has, one would expect this algorithm to have good performance. It is also relevant that this model is the less biased one, since it has mean closest to 0 than any other algorithm.

Finally, let me compare the two best models, that were XGBoost and Skforecast. XGBoost slightly outperformed the second algorithm, but it is a bit biased. One can appreciate in table 4.3, the mean error is around -200, and has the lowest standard deviation. Taking into account that it has much less outliers than the Skforecast model, and that the maximum error of this model, as we can see in table 4.2 is smaller than the second one, we can conclude that XGBoost showed better performance.

However, despite being the model that had the second best performance, for future experiments Skforecast has much more potential for improvement. It performed really well, since it combines time series forecasting and machine learning

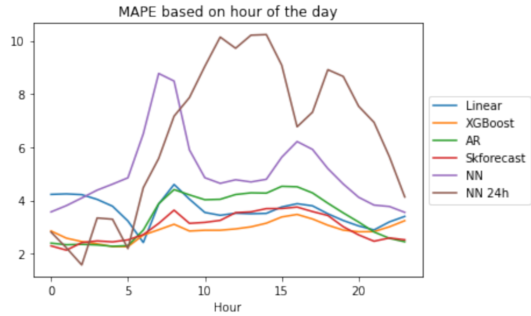

FIGURE 4.1: MAPE based on hour of the day.

(XGBoost in this case), considering also that this model had much less information that the XGBoost algorithm, like temperatures and calendar, which are crucial features. The days where this model had poor performance were holidays during the midweek, where the model did not expect to have such a low demand. If we excluded these days, the metrics would have been much closer to the XGBoost ones.

Note that the error can depend on the hour of the day and day of the week, since as we could see in the data exploration section, electricity demand varies greatly throughout the day or during the week. By having a look at figure 4.1, we can see that in the morning and in the evening we have more MAPE than at night. This makes sense, because at that time we have much more demand variability than in the early morning.

TABLE 4.2: Metrics of all the models tested.

	Linear	**XGBoost**	**AutoReg**	**Skforecast**	**NN**	**NN 24h**
RMSE	1262.7	1021.1	1365.2	1187.2	1384.1	1483.5
MAE	935.8	747.4	888.4	772.8	997.8	1147.8
MAPE (%)	3.62	2.89	3.42	3.00	3.87	4.45
MAX	6947.2	7550.2	12121.9	10948.2	8740.1	7337.2

TABLE 4.3: Distribution of the error for all the models tested.

	Linear	**XGBoost**	**AR**	**Skforecast**	**NN**	**NN 24h**
μ	-72.8	-194.5	10.8	22.5	-128.9	1060.9
σ	1260.6	1002.5	1365.2	1187.0	1378.1	1037.0
$\mu \pm \sigma (\%)$	72.9	74.7	81.8	81.3	75.7	73.3
$\mu \pm 2\sigma (\%)$	94.1	94.5	94.3	95.2	94.0	95.6
$\mu \pm 3\sigma (\%)$	99.1	98.8	97.6	98.0	98.5	98.9
$\mu \pm 4\sigma (\%)$	99.9	99.6	99.1	99.1	99.7	99.6

4.2.3 Selected model

As commented above, in my opinion the model can improve the most when using more precise inputs will be the Forecaster AutoReg using XGBoost (Skforecast) model. We saw that it got similar performance than the best model (XGBoost algorithm) but with less information. If we add these extra variables, like calendar and temperatures to the Skforecast model, we will for sure improve our results much more than with any other tested model.

4.3 Influence of exogenous variables

Once we have selected the most promising model, the Forecaster AutoReg model from the Skforecast library, we will work more deeply with it in order to refine it and optimize the results. This algorithm combines machine learning models with time series. In the previous experiment, we got good results using only previous demand records as inputs of the model. But this library enables us to add extra variables that may help the model capture complex relationships between lags, or even between these variables. This features are called exogenous variables. We will include this variables as inputs in this experiment, we will also perform a hyperparameter tuning process, aiming to improve the results. Finally, we will compare our predictions with ESIOS.

4.3.1 Building the input

In the Skforecast library, they differentiate between time variables (lags) and exogenous variables. Regarding the first ones, we are considering the lags corresponding to the electricity loads from the last 28 days at exactly the same our than the one we are trying to predict. The second type of variables can be summarized in the following list:

- Day of the month.
- Month of the year.
- Week of the year.
- Holiday
- One-hot encoding of the day of the week. That is, we built 6 variables called 'L', 'M', 'X', 'J', 'V', 'S'. Since Sunday is always a holiday, we decided to include it in the holiday feature.
- Temperature.
- Radiation.
- Wind.

Note that the CPI and GDP were removed features compared to the first experiment. This is because their relevance is minimal, since they have very small variations in the period of time used by the model, because they are long term features and we are making short term predictions.

We can divide the exogenous variables in daily and hourly variables. Those that are related to the calendar belong to the first group and the weather variables to the second one.

In this experiment, instead of putting the data without any preprocessing, like we did in the first one, we will apply some preprocessing to the data. The objective of this is to pass to the model instead of the raw data, information about how does the variable vary with respect to its past values. In the case of some of the calendar variables, we are dividing it by the mean of the past 28 days.

$$X_t = \frac{x_t}{\frac{1}{2}\sum_{i=1}^{n} x_{t-i}} \tag{4.5}$$

Where X_t denotes the exogenous variable X that will be the input of the model at time t, x_t the initial variable at time t, and n the number of past values we are taking into account, in this case we have $n = 28$. Note that we only applied this to the one-hot encoding of the day of the week. For that day, we had that $x_t = 1 - F$, where F is the weighted holidays average based on population that was computed at the beginning of the chapter. However, we will not apply this transformation to the holidays feature because we want it to measure the grade of festivity of that day, and with the one-hot encoding we want to see how much of a work day is that day with respect to the previous days, because if we had a lot of holidays in the previous days, the overall demand would be lower than usual.

Regarding the weather features (temperature, radiation and wind), we are going to divide the first two in five geographical areas based on their climate. We do not apply this to the wind because wind is a much more local variable, and as it was commented before, it does not have the same influence as temperature and radiation when predicting electricity load. In figure 4.2 the Iberian peninsula has been painted taking into account the partition that was made, based on the climate:

- Mediterranean: this climate is characterized by hot, dry summers and warm, wet winters. Note that this area was divided in 2 regions in order to have more precise data, since this weather covers the biggest area of these four, and accumulates most of the Spanish inhabitants.
- Oceanic: is found in the northern seaside areas, characterized by mild temperatures during the year, with abundant rainfall distributed throughout the year.
- Arid: this climate is characterized by low annual rainfall, higher temperatures, and typically found in the interior regions.
- Continental: characterized by more significant temperature variations between summer and winter, often found in interior areas away from the moderating influence of the sea.

Now, as we did with the daily features, we are going to pass to the model the variation of the data with respect to its previous values. But in this case we compute it by subtracting the mean. We do this because as we could see in figure 3.7, we have two linear dependencies between the demand and the temperature, so we tried to preserve this relationship.

$$X_t = x_t - \frac{1}{n}\sum_{i=1}^{n} x_{t-i} \tag{4.6}$$

Where, once again, X_t denotes the exogenous variable X that will be the input of the model at time t, x_t the initial variable at time t, and n the number of past values

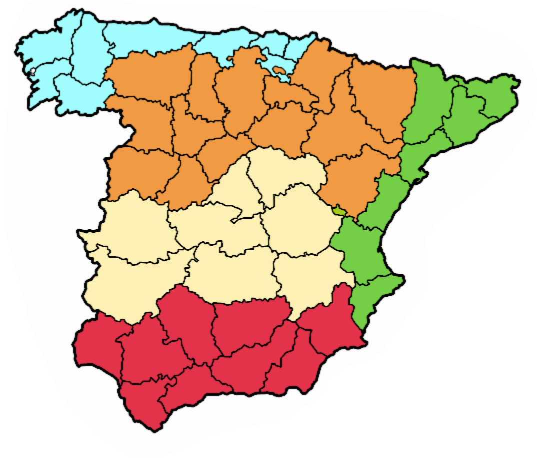

FIGURE 4.2: Colormap of the division based on the climate. Cyan area corresponds to oceanic weather, orange to continental, beige to arid, and green and red to Mediterranean but we are dividing it in two regions.

we are taking into account, in this case we have $n = 28$.

Additionally, we will normalize also all the features that we transformed previously, by bounding them in the $[0, 1]$ interval, where X' will be the input of the skforecast model:

$$X' = \frac{X - min(X)}{max(X) - min(X)} \tag{4.7}$$

It is important to know that the demand is also normalized within the [0,1] interval, but when it comes to computing the metrics, the transformation has to be reverted.

A correlation analysis of the features obtained was also made. It can be found on appendix C.

4.3.2 Experiment results

In appendix D one can find an experiment testing different machine learning algorithms as regressors of the Forecaster AutoReg model on the inout commented above. We get that the best regressor is XGBoost, that matches with the algorithm used in previous experiments.

Now, let us compare the obtained results including these features with a model where we do not include them to see how big the improvement is. These will be compared with ESIOS predictions, to see how far they are from the official predictions. In table 4.4, it can be seen that including the exogenous variables helped the model in predicting the electricity load of the next day, since we got better results in all indexes. It is worth highlighting the huge reduction of the maximum error. As commented before, the fact that the standard model makes that big mistakes in the predictions could be due to that the model does not have the information of when a typical work day is a holiday. Now that the model has that information available, it

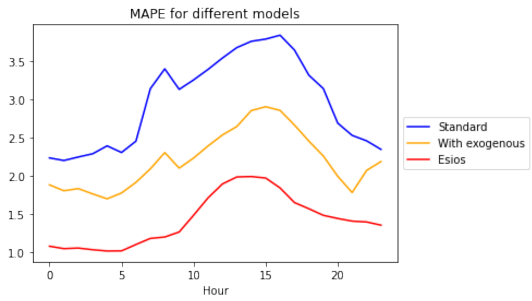

FIGURE 4.3: MAPE based on the hour of the day for the Skforecast model without exogenous variables, the model that includes them, and ESIOS predictions.

does not make those kind of mispredictions.

TABLE 4.4: Metrics of all the models tested and ESIOS.

	Standard	**With exogenous**	**ESIOS**
RMSE	1163.9	793.2	518.0
MAE	765.5	570.9	376.3
MAPE (%)	2.97	2.21	1.43
MAX	10239.0	5903.1	2840.3

Regarding figure 4.3, where we can appreciate the MAPE of each hour of the day, clearly the model with exogenous variables outperformed the standard model. The three curves are practically parallel in both plots, which means that including exogenous features contributed to a better performance at every hour of the day.

Finally, having a look at table 4.5, it is relevant that the standard deviation of the model with exogenous variables is smaller than for the standard model, but keeping the same number of examples withing each confidence interval, which confirms the previous statements. This means that we got a more precise model by adding those features. However, note that we have more outliers than the standard model, this could be an statistical fluctuation, since the difference is only a 0.1%.

TABLE 4.5: Distribution of the error for all the models tested and ESIOS predictions.

	Standard	**With exogenous**	**ESIOS**
μ	8.4	-7.6	-16.9
σ	793.2	710.6	517.7
$\mu \pm \sigma(\%)$	75.8	76.5	75.3
$\mu \pm 2\sigma(\%)$	94.8	94.5	94.2
$\mu \pm 3\sigma(\%)$	98.7	98.5	98.6
$\mu \pm 4\sigma(\%)$	99.6	99.5	99.7

4.4 Fine tuning

We will also perform some hyperparameter tuning of the machine learning algorithm that goes inside the Forecaster AutoReg model from the library Skforecast. In this case, we are using the XGBoost algorithm, as well as we did in the previous experiment. We are performing a fine tuning on the following hyperparameters:

- n_estimators (*#Trees*): The number of decision trees of the XGBoost model. We set the minimum number of estimators to 300, and the maximum to 1500. However, in order to simplify the process a bit, we consider only configurations such that the number of trees are multiples of 20.
- max_depth ($MaxDepth$): Maximum depth of each tree. They vary from 3 to 15.
- learning_rate (LR): This hyperparameter controls how fast the algorithm will drive toward the minimum gradient value. Its values go from 0.0001 to 0.5, following a log-uniform distribution.
- reg_alpha (α): The parameter that controls the relevance of the L1 regularization term. As well as the learning rate, the probability distribution of it is log-uniform, but the range of values goes from 0.001 to 1.
- reg_lambda (λ): The parameter that controls the relevance of the L2 regularization term. The probability distribution is exactly the same as the α parameter.
- subsample ($Subsample$): it indicates the percentage of training examples to use in each tree, so it takes values from 0 to 1. The probability distribution of this feature is uniform.

Note that we are applying a Bayesian search of hyperparameters, that is, the new hyperparameters depend on the previous results and configurations. We do 100 iterations of the process, and we obtain the hyperparameter configuration shown in table 4.6. As we can see, the new configuration is very different compared to the default one, so we should also obtain different results when predicting the test set. At this point, we still do not know if the performance of the tuned model will be better or worse than the original one, but taking into account that we optimized the model hyperparameters, we expect to get a better model.

TABLE 4.6: Default and tuned hyperparameters after Bayesian search

	#Trees	**MaxDepth**	**LR**	α	λ	$Subsample$
Default	100	6	0.3	0	1	1
Tuned	1500	5	0.100227	0.700776	0.001089	0.893955

4.4.1 Results of the tuned model

After the hyperparameter tuning, we will evaluate the obtained model on the test set. We get the results that appear on table 4.7. As we could expect, we get better results than the model with the default configuration, and we come closer to the ESIOS predictions. The remaining difference with the ESIOS results can be partially due, as we can see in the values of the maximum errors committed, to the fact that the tuned

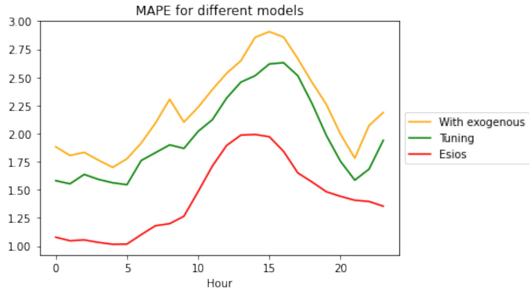

FIGURE 4.4: MAPE based on the hour of the day for the Skforecast model with exogenous variables, the tuned model, and ESIOS predictions.

model has a few very large errors that ESIOS does not commit, making the rest of the metrics of our model a little inflated. This can be seen on table 4.8, where we have a much bigger standard deviation, and a bigger number of examples further than 4σ.

TABLE 4.7: Metrics of the Skforecast model including exogenous variables, after performing hyperparameter tuning and ESIOS predictions.

	With exogenous	**Tuning**	**ESIOS**
RMSE	793.2	710.6	518.0
MAE	570.9	509.0	376.3
MAPE (%)	2.21	1.97	1.43
MAX	5903.1	5471.4	2840.3

TABLE 4.8: Distribution of the error for the Skforecast model including exogenous variables, after performing hyperparameter tuning and ESIOS predictions.

	With exogenous	**Tuning**	**ESIOS**
μ	10.2	-1.7	-16.9
σ	793.2	706.8	517.7
$\mu \pm \sigma(\%)$	75.8	76.6	75.3
$\mu \pm 2\sigma(\%)$	94.7	94.5	94.2
$\mu \pm 3\sigma(\%)$	98.7	98.5	98.6
$\mu \pm 4\sigma(\%)$	99.6	99.4	99.7

Regarding the error based on the hour of the day (figure 4.4) we can see that the tuned model outperformed the default model on every hour, which gives an idea of how important is to select the model hyperparameters that better suit the problem we are trying to solve.

4.5 Conclusions of the experiments performed

In this section, we tried most of the models explained in the State of the Art chapter. To do so, we designed an experiment where different algorithms were tested, starting with the simplest model (a linear regression), and also fitting more complex algorithms like neural networks. In this experiment we got surprising results, highlighting the outstanding performance of the linear model, outperforming the neural network models. A reason of it may be that the configuration of the neural network was not the optimal, but the fact that we got the worst results for this algorithm made me dismiss this kind of models.

We also tried autoregressive algorithms as well, since as it has been commented throughout all the project, electricity load has a seasonal component that can be used when developing models that are capable of detecting these patterns. We got good results with it but had problems when forecasting special days, like holidays that take place on typical work days. However, this features were used by the XGBoost algorithm we implemented, and as a result this model showed a great performance, being the most accurate model in the experiment. Note that SVR was also tested, but with this model showed a poor performance with the default settings. In order to try to improve the quality of its predictions, we performed hyperparameter tuning on this model but the time computation was much higher than other models and the results were still bad.

Finally we chose a model that combines both machine learning and time series forecasting, the Forecaster AutoReg, from the Skforecast library. In the first experiment, we got similar results compared to XGBoost, but with much less information, like the calendar and the weather, so we thought that studying in more detail this kind of models would be the best idea.

And that is what we did on the second experiment, by adding those features and dividing the weather variables in 5 geographical regions based on the climate of these areas, and also made some data preprocessing. We also normalized the exogenous features in the [0,1] interval. After all this, the model was trained, improving a lot the results, confirming that we made the right choice by selecting this model for further experiments.

Then, the most relevant hyperparameters of the model were tuned, aiming to improve the model performance. And so did the model, the results improved. Finally, our best result was a MAPE of 1.97% for our approach, and 1.43% for ESIOS. However, if we do not consider the outliers of the predictions for both algorithms, we got a bit closer, since our model has some predictions that are very far away from the actual value, and ESIOS does not have such big mistakes.

As we could appreciate in tables 4.4 and 4.7, including exogenous as inputs features made the model achieve better results than when it considered only electricity demand from previous days, and that tuning the most important hyperparameters of the machine learning model that we are using is key to maximize its performance. However, we are still a bit far away from the ESIOS predictions. One possible reason is that we have more outliers than the ESIOS model, and that our maximum error is much bigger than ESIOS's. This means that we have a few samples whose error is huge and they inflate the estimation of metrics of the models. If we removed those outliers (the examples that are further than 4σ from the mean), we would have the table 4.9. As expected, we are now closer to the ESIOS predictions, since we removed more outliers that had bigger errors associated to them.

TABLE 4.9: Metrics of the tested models and ESIOS after removing outliers.

	With exogenous	Tuning	ESIOS
RMSE	750.8	663.4	500.3
MAE	556.6	490.1	369.9
MAPE (%)	2.15	1.89	1.40
Number of outliers	37	49	27

It is important to note that another experiment was performed, but this can be found on the appendix E, where another approach for the electricity forecasting problem was tried, by building 24 models, one for each hour of the day, and then combining its predictions. We tested first the same algorithms as the model selection experiment, resulting in that the linear, XGBoost and Forecaster AutoReg are the best performing models. Then, these 3 methods were once again tested on the second, more detailed, input explained before. All models got worse results than in the second experiment, because since these are smaller models, they do not need a more specific input to converge.

Note that all the code needed for the execution and fine tuning of the models is available on Github.

Chapter 5

Explainability

Explainability of a machine learning model refers to the ability to understand, interpret and explain the decisions or predictions made by them. Many times these models function as black boxes, making it difficult to understand why a particular prediction was made. To help explain this model, two methods will be used, Shapley values and L1 regularization. Shapley values are a popular method for explaining machine learning models, as they help to understand how variables and values influence predictions visually and quantitatively. They show how each feature affects each final prediction, the significance of each feature compared to others, and the model's reliance on the interaction between features. On the other hand, L1 regularization adds a penalty term on the loss function, that forces the model to select the most important features. It will be performed feature selection using a linear model (that is easily explainable) and will tune a model using the same method that is used in the second experiment of the experimental section considering only the selected features by the linear model. After that, the Shapley values of the obtained model will be shown, and finally, both explainability methods will be compared.

5.1 Using Shapley values

As it is explained in (Caelen, 2022), the idea of Shapley values come from cooperative game theory, where they give a way to do a fair distribution of payoffs to the players (in our case, the players are the variables). In machine learning terms, this refers to measuring the impact each feature has on the model output, that is, how big or small the variation of the prediction is depending on the value of that feature. In the appendix F.1 one can find the mathematics behind the Shapley values.

5.1.1 Application to the tuned model

As explained above, the Shapley values measure how each feature affects in each prediction, that is, for each feature there are as many Shapley values as samples to predict, so it can be computed the overall relevance of each variable by simply averaging the absolute value of all of them (if the absolute value is not considered, positive and negative cases cancel each other). By doing this, it is obtained figure 5.1. Note that the number that goes with the lag variables represents the number of hours of delay the data have, that is, the feature lag_168 means that it is the demand from 168 hours ago, which is one week. The most relevant features are the lags corresponding to 7, 1, 21, 14, 3 and 28 days, and the exogenous variables of holidays, week, Saturday and Monday. Regarding the time variables, taking into account the autocorrelation plot (figure 3.5), it makes sense that these lags are the most relevant features, since they are the ones with the highest correlation. In relation to the exogenous variables, it is noticeable the influence of the calendar, where holiday is one of the most important features (as it can be appreciated in 3.6). After holidays, the feature with the biggest impact is week, and that is also reasonable, because if we have a look at figure B.3, and knowing that the features month and week are strongly correlated, we can conclude that its relevance was going to be high, too. Then we have Saturday and Monday. Saturday is also a special day of the week, when its demand is lower than in midweek, hence the impact of this feature is high. Monday is a little different from the rest of the midweek days, that is why it is the next exogenous variable in impact, since the demand curves for Tuesday, Wednesday and Thursday are almost the same, and Friday evening is a little different, but not as much as Monday. It is worth mentioning the low impact of the weather features. That could be because we divided the temperature and radiation in 5 features each, so the impact of each of them is smaller than if we considered only one variable for temperature and another one for radiation. The areas with the biggest impact are the most centered ones, since they are more correlated with the other regions. It is also important to note that the output predictions of the demand are normalized in the interval [0,1], and so do the Shapley values.

Let us show another revealing plot. Above, the relevance of each variable has been shown, but let us now see the impact it has on the output depending on the value of each feature, that is, how the predictions are affected when the variable has a high or low value. This is reflected in figure 5.2. It can be noticed, considering 3.5, that for all lags we have that the greater its value the more positive its impact in most of the cases. This makes sense, since the correlation is positive. Regarding the calendar features, note that its relevance is quite close to 0 when the value the of variable is low (that is, when the day of the week does not match the variable name or when it is not a holiday). Knowing that in holidays and Saturdays the demand is always lower than usual (see figure 3.3), when its value is high the impact is negative, since it tends to minimize the demand. With respect to the temperatures, we do not have a clear tendency. One can think that taking into account the figure 3.7, there should not be negative impacts for low temperatures, but these points must belong to predictions in the early morning in winter, when the temperature is very cold and the demand is lower than usual. On the other hand, the positive impacts must be in the hottest days of the year, where the air conditioning is turned on more than normal.

5.2 Using L1 regularization

In this part, L1 regularization will be performed. L1 is well-known for its tendency to force some of the model coefficients to zero, effectively excluding certain features from the model. But it will not be done directly on the Skforecast model, but on a linear model, that is more easily explainable. Then, its results will be directly implemented on another Skforecast model, that will be fine tuned. Finally, Shapley values will be computed with the selected features in order to visualize how they affect the model decision-making. The L1 regularization path can be found in the appendix F.2.

The selected features from the linear model are the following:

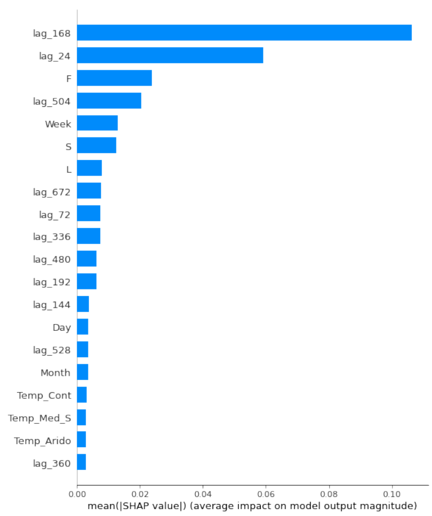

FIGURE 5.1: Mean absolute value of the Shapley values of the most relevant features for the tuned model.

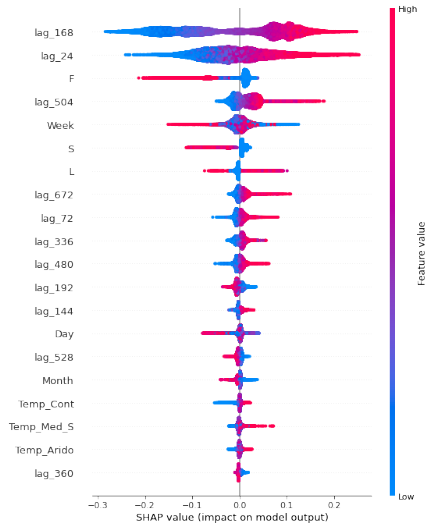

FIGURE 5.2: Impact on model output of the features depending on the value of the feature for the tuned model

- Lags features: all of them except those from 2, 4, 12, 13, 16, 18, 24 and 27 days before.
- Calendar features: all of them except holidays.
- Weather variables: Only temperature in the continental region and radiation in the arid climate area.

Finally, once the feature selection is done, let us train another ForecastAutoReg model the same way it was done on the experimental section. It was trained with the exogenous variables and lags selected, and then a hyperparameter tuning was made in order to maximize its performance. The obtained results are shown in table 5.1, where we compare the results of the tuned model in the experimental section, where all features are included, and then the tuned model after performing feature selection. As we can see, selecting features in this case made the model perform worse. It makes sense, since the linear model dismissed a lot of exogenous features.

TABLE 5.1: Metrics of the Skforecast model including all features and after performing feature selection from a linear model. In both cases a hyperparameter tuning was made.

	Tuning	**Feature selection**
RMSE	710.6	898.1
MAE	509.0	635.3
MAPE (%)	1.97	2.46
MAX	5471.4	5547.0

In order to explain the functioning of the Skforecast model with feature selection, the Shapley values will be once again used. Figure 5.3 shows the average absolute impact of the most important features for this model. It can be seen that in this case the lags features are much more important than the exogenous variables. The electricity load of 1 and 7 days before are the two lags with higher impact. Then have the lags of 21,8,28 and 14 days before the first exogenous variables appear, that are Tuesday, Week and Thursday. It is worth highlighting that the climate variables are not relevant anymore, since only two of them were not removed by the linear model, and they represent only some regions of Spain, so it is reasonable to think that the model is not going to pay too much attention to them.

In figure 5.4, the impact on the model output depending on the value of each variable for the feature selection model is represented. Most of the lags have a positive correlation with the impact on the output variations, that is, the higher the value of one lag, the more positive the impact on the output. It can also be appreciated that most of the calendar features do not have a clear correlation with the output, since the points are symmetric. This is because there is not a high linear correlation (positive or negative) between the electricity load and these features, as seen in the electricity demand exploration section.

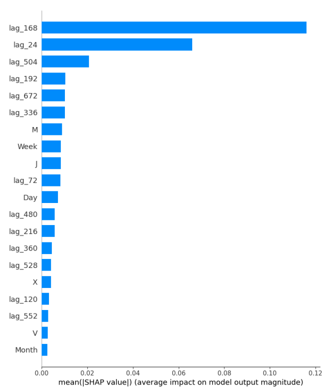

FIGURE 5.3: Mean absolute value of the Shapley values of the most relevant features for the feature selection model

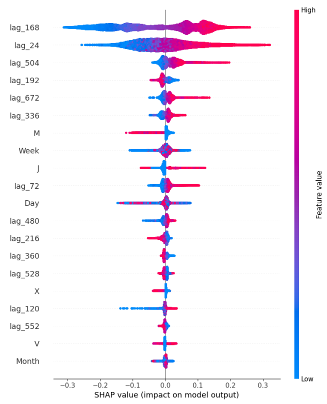

FIGURE 5.4: Impact on model output of the features depending on the value of the feature for the feature selection model

5.3 Comparison between both methods

When it comes to understanding the decision-making of a machine learning model, two of the most popular methods are the Shapley values and L1 regularization. The first one computes the impact of each feature in each prediction (compared to the average value of the output), and the second one adds a penalty term in the loss function that is the objective function that is trying to be minimized.

What is really surprising in the case of the L1 regularization method is that the linear model removed the holidays feature, the most relevant exogenous feature of the tuned model. Note also that the week feature has also much less impact on the feature selection model than in the tuned one. And even more, the most relevant days of the week in the tuned model is Saturday (where demand varies the most compared to the others, excluding Sunday) and Monday (the midweek day that has the most differences with the rest). In the other case, Tuesday and Thursday are the most relevant weekdays, two very similar days in terms of electricity demand. All these are some of the possible reasons for the worse performance of this model compared to the obtained in the experimental section.

Regarding figures 5.2 and 5.4, it is also worth mentioning that in the first plot we have a clearer correlation between the value of the variable the output variations than in the second plot, that is because the model has understood better the role of each feature. This does not happen in the second image, where we have more symmetries, specially in the case of the exogenous variables.

We can conclude this chapter by saying that the most relevant variables and lags are accurately ranked in the case of the tuned model, focusing on the lags of past weeks and the day before, as well as the calendar is also relevant, where the holidays feature is the third variable in impact on the output variation. The climate variables are less relevant than calendar and lags features, which is also reasonable, but they help the model in refining the predictions. However, in the case of the model with the selected features both calendar and climate features lose a lot of relevance. That could be because most of the weather variables and holiday were removed from the linear model, and also the most relevant days of the week are not the most different ones compared to the others. Since the linear model is much simpler than a ForecasterAutoReg, the first one can improve results if some of the features are removed, but the second one had to converge with much less extra information provided by the exogenous features, and that is why the forecaster with the L1 selected features model performed worse.

Note that all the code needed for this analysis is available on Github.

Chapter 6

Conclusion

This project has focused on the electricity load forecasting problem in the daily market that is, on the short-term purchase and sale of electricity. A demand exploration was made, and it was concluded that the calendar and the weather, especially the day of the week, whether a day is a holiday or not and the temperature, are the factors with the greatest impact on the next day electricity load. It was appreciated that economy is not that influential as the previous factors in the case of short-term electricity load forecasting, since the indices that represent the economical development of a country vary less in the short term.

Once the factors that most influence electricity consumption were known, and the state of the art had been studied, it was proposed to test some of the possible prediction models with a simple input, in order to see which ones show better performance. We picked a model that combines both machine learning and time series algorithms, implemented in the python library Skforecast.

Then, we worked on refining the chosen model. To do that, it was decided to include exogenous variables to provide context to the time series values. An input was constructed that includes variables related to the calendar, such as the week of the year, holiday of that day, or the day of the week, as well as the temperature and solar radiation of Spain segmented by climatic zones. All this, plus a hyperparameter tuning, led to a huge improvement on the model performance. The average error obtained was 1.97%, which is reduced to 1.89% if we remove the outliers. To put this result in context, ESIOS predictions get an average error of 1.43%.

Finally, an attempt was made to explain the decision-making process of the tuned model and which variables it pays the most attention to. To do this, two methods were used, Shapley values (which come from game theory and calculate the output variation of each variable on each prediction) and L1 regularization, which allows us to perform feature selection, getting in the first case a reasonable result, and a bit surprising result with the second method.

Looking at future improvements to the model obtained in this project, a proposal would be to combine time series analysis with deep learning structures, instead of machine learning algorithms. This is done in (Carazo and Rodrigo, 2024), where the authors implement recurrent neural networks for electricity load forecasting.

Appendix A

Forecasters implemented in Skforecast library

In the State of the art section, it was commented that there exist different forecasters, depending on the forecasting horizon, recursivity... If the forecasting horizon is just one step ahead, it is called a single-step forecasting. However, when working with time series, it is seldom needed to predict only the next element in the series (t+1). Instead, the most common goal is to predict a whole future interval (t+1, ..., t+n) or a far point in time (t+n). Several strategies allow generating this type of prediction. Those are the recursive and direct forecasting.

- Recursive forecasting: since the value t(n-1) is required to predict t(n), and t(n-1) is unknown, a recursive process is applied in which, each new prediction, is based on the previous one. Figure A.1 is a diagram of this technique.
- Direct forecasting: consists of training a different model for each step of the forecast horizon. For example, to predict the next 5 values of a time series, 5 different models are trained, one for each step. As a result, the predictions are independent of each other. Figure A.2 allows us to visualize the process.

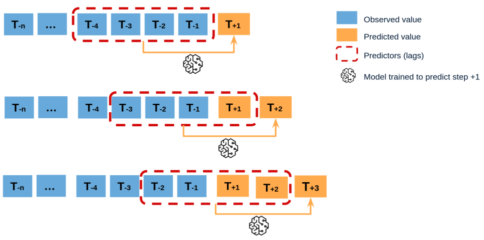

FIGURE A.1: Recursive forecasting scheme

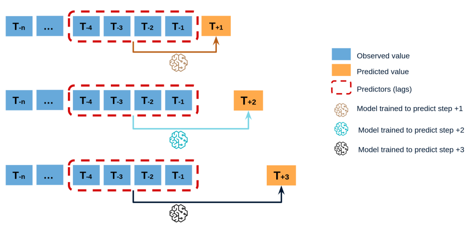

FIGURE A.2: Direct forecasting scheme

Appendix B

Complementary information related to the electricity load exploration

Let us show how the data is distributed depending on the time of day, day of the week or month of the year.To do this, box plots, which are used to show distributions of numeric data values, especially for comparison purposes, are going to be used. They use boxes and lines to depict the distributions of one or more groups of numeric data.

Box limits indicate the range of the central 50% of the data, with a central line marking the median value. Lines extend from each box to capture the range of the remaining data, with dots placed past the line edges to indicate outliers. Construction of a box plot is based around a dataset's quartiles, or the values that divide the dataset into equal fourths. The first quartile (Q1) is greater than 25% of the data and less than the other 75%. The second quartile (Q2) sits in the middle, dividing the data in half. Q2 is also known as the median. The third quartile (Q3) is larger than 75% of the data, and smaller than the remaining 25%. In a box plot, the ends of the box and its center line mark the locations of these three quartiles. The distance between Q3 and Q1 is known as the interquartile range (IQR) and plays a major part in how long the whiskers extending from the box are. Each whisker extends to the furthest data point in each wing that is within 1.5 times the IQR. Any data point further than that distance is considered an outlier, and is marked with a dot.

Figure B.1 represents the distribution of data at different hours of the day. Note that the medians follow the same pattern than 3.1, with two peaks in the morning an in the evening, and two valleys at lunchtime and before the sunshine. But there are a lot of low outliers. These are due to two reasons. One of them, is that these days are holidays (in figure 3.3 we could see that demand on holidays or Sundays are very low compared to midweek), or that they occurred in confinement due to the COVID pandemic, where demand was extremely low (it can be appreciated in figure 3.2). It is worth mentioning that the whiskers are wider in the central hours on the day than in the early hours of the day. This makes sense, since at night climatic or holiday factors influence daily activity much less than during the middle of the day, since at those hours the vast majority of the population is sleeping.

In the same way, if we build a boxplot of the days of the week, we can draw the same conclusions as those in the figure 3.3. The days of least demand are Saturday and Sunday, in which economic activity is lower, and Monday and Friday have a

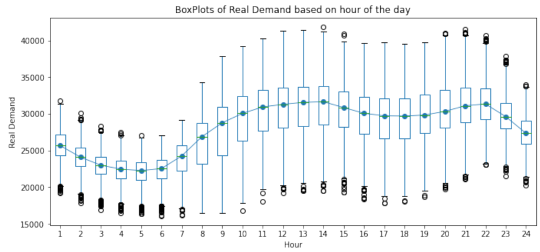

FIGURE B.1: Hourly Demand boxplot

lower demand than Tuesday, Wednesday and Thursday, as mentioned above. Regarding the variability of the data, as expected the width of the whiskers is much smaller on the weekend, since they are days where the routine is clearer. However, on those days we can find outliers, which may be due to high or low temperatures, causing many households to have to turn on the heating or air conditioning more than usual.

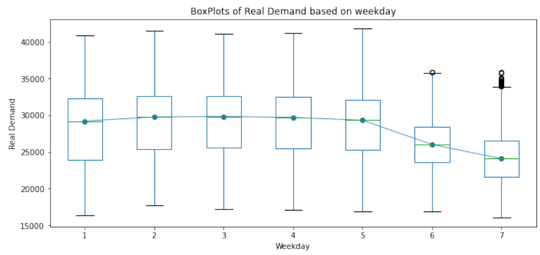

FIGURE B.2: Weekly Demand boxplot

The last boxplot to be discussed is figure B.3, where it can be seen how the demand is distributed in the different months of the year. Notice that in summer and winter we have more demand than in fall and spring. That is because of the weather. In winter and summer we have the coldest and hottest days of the year, which makes people turn on the heating or air conditioning.

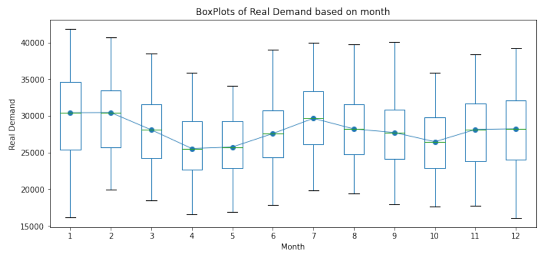

FIGURE B.3: Monthly Demand boxplot

Appendix C

Correlation between the different variables of the second experiment

In this appendix the correlations between the different variables that serve as inputs for the second experiment are shown. Figure C.1 shows those correlations. As we can see, the climatic variables are strongly correlated, which makes sense, since there are usually not totally different temperatures or radiations in each area of Spain. However, it should be noted that the regions between which there is the higher correlation are in the adjacent areas, and in those that are further away that value is smaller, which indicates that there may be specific temperature differences in certain days of the year. On the other hand, it is normal for temperature and radiation to have a certain correlation, since normally temperatures are higher when there is more radiation. As one could expect, the weather variables are also correlated with the month and week of the year, and that these two are almost identical, but I decided to keep them both. Regarding the days of the week, they are a bit negatively correlated, since these variables are part of a one hot encoding, when always one variable is not 0, while the other ones do.

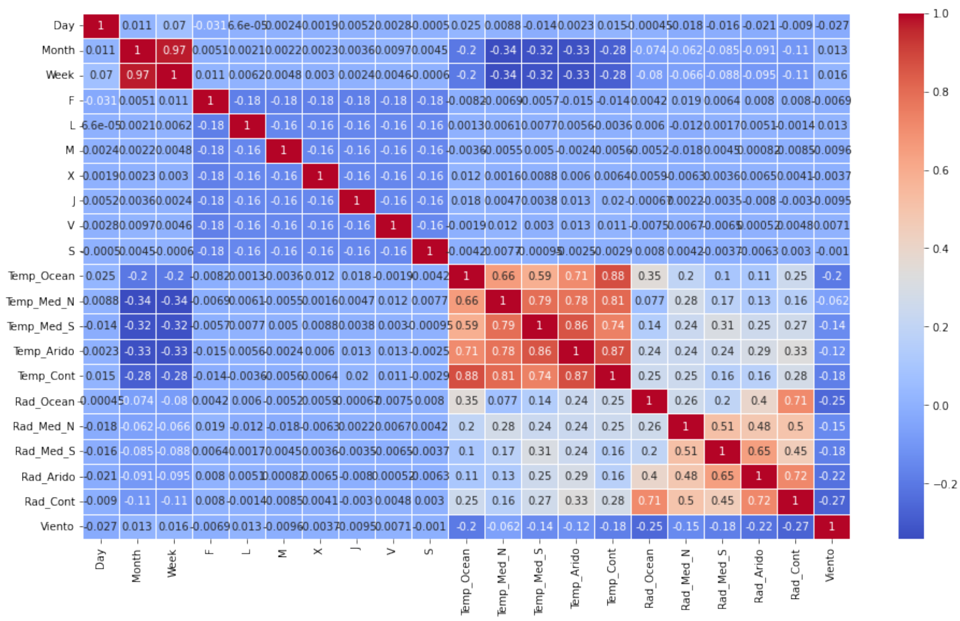

FIGURE C.1: Correlation matrix of the second experiment input variables

Appendix D

Selection of ML algorithm as regressor in ForecasterAutoReg model

As we could see in the State of the Art section, all models compatible with Scikit-learn API can be used in the ForecasterAutoReg models as regressors. So we can decide which regressor will be used in the experiments. In the model selection experiment XGBoost was the used algorithm, but for the second experiment where exogenous variables and a more detailed input were included, we tried different machine learning model in order to decide which one was going to be the regressor of the ForecasterAutoReg model that was going to be tuned.

In the Skforecast website (Amat Rodrigo and Escobar Ortiz, 2023) the developers of the library used tree-based machine learning models, such as Extreme Gradient Boosting (XGBoost), Light Gradient-Boosting Machine (LGBM) or Random Forest Regressor (RF). These will be the tested models. Note that a linear model was also included there, since in the model selection experiment it showed good performance.

In table D.1 the metrics of the experiments are shown. Note that the input of the forecasters is the same input that is used in the second experiment, that is, when the exogenous features were considered. As we can see, XGBoost is the most accurate regressor, and having a look at figureD.1, we can see that this regressor outperforms the others in almost all hours of the day.

To sum up, we decided to keep using XGBoost as machine learning regressor for the upcoming experiments.

TABLE D.1: Metrics of the Skforecast model including exogenous variables, for different regressors of the Scikit-learn API.

ML regressor	Linear	XGBoost	LGBM	RF
RMSE	1125.7	793.2	851.6	919.2
MAE	818.8	570.9	610.2	637.6
MAPE (%)	3.19	2.21	2.37	2.45
MAX	10158.9	5903.1	4878.1	6331.1

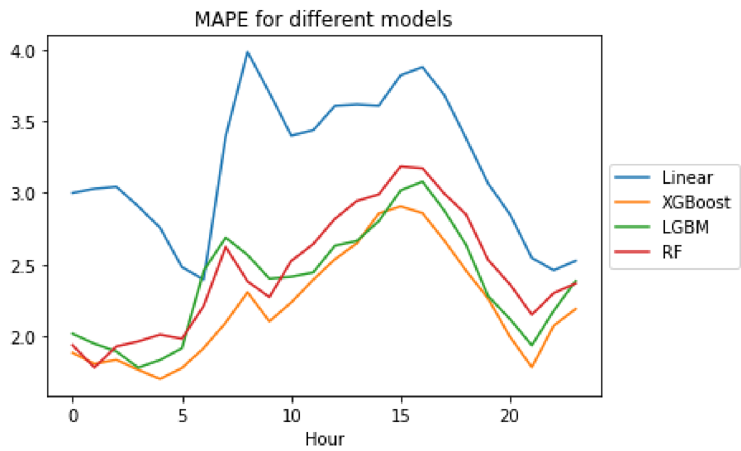

FIGURE D.1: MAPE based on the hour of the day for the different ML algorithms tested on ForecasterAutoReg.

Appendix E

One model for each hour of the day approach

In this project, we also developed another approach for the electricity load forecasting problem, that is building for each algorithm a total of 24 small models, one for each hour of the day. The reason this approach was considered is that the biggest autocorrelation between a demand value and its past lags happens at the same hour of the day but on the previous days. The dataset that has been used throughout the project was partitioned into 24 parts, and each of them will not interact with the other 23 fragments. This is equivalent to a direct forecasting model, in terms of Skforecast forecasters, where one model is built for each step (one model for each hour of the day).

The methodology followed in this approach will be the same as the followed in the main one, that is the following:

1. Perform a first experiment trying different models on a simple input in order to select the most accurate ones for the upcoming experiment.
2. Build a much more detailed input. In this case we will use the same input as the main approach.
3. Design and run a second experiment with the selected models and the new input.
4. Fine tune the model hyperparameters.

The results of the model selection experiments are shown in table E.1. Note that the tested models are the same as the main approach. All models in this approach outperform the "big" models that are shown on the thesis. But this has a simple explanation. Since the models tested here are much smaller, it is normal that when the input is also small, the small models converge better. As well as it happened also in the main approach, XGBoost showed the best performance. It is quite surprising the outstanding results obtained by the linear model, getting even better results than the Skforecast algorithm. In figure E.1 one can appreciate the MAPE based on the hour of the day for the different models tested. As we can see, XGBoost and the linear model have very similar errors, except the morning, that is the time of the day where XGBoost takes advantage of the linear approximation. On the other hand, both time series models have also quite similar behaviour.

Now, if we fit the selected models (linear, XGBoost and Skforecast) with the second, more detailed, input, we have table E.2. Note that in the case of both linear and

TABLE E.1: Metrics of all the models tested.

	Linear	XGBoost	AutoReg	Skforecast	NN
RMSE	945.6	856.2	1263.5	1229.2	1613.6
MAE	688.1	639.5	867.3	854.5	1172.2
MAPE (%)	2.60	2.48	3.35	3.35	4.45
MAX	4266.9	4040.2	6735.0	6677.4	6633.7

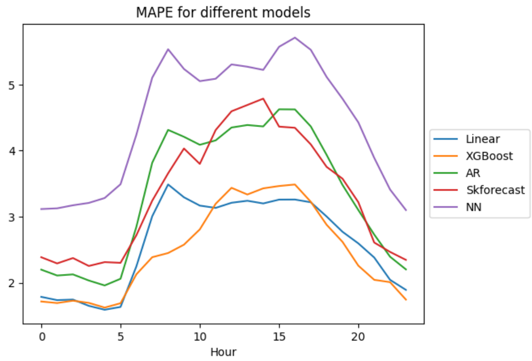

FIGURE E.1: MAPE based on the hour of the day for the different small models tested.

XGBoost models, the results are worse than in the second experiment, even when performing a fine tuning on the XGBoost model, as one could expect taking into account what was commented above. In fact, if we have a look that the explainability chapter, when we were talking about the L1 regularization, we could see that the linear model showed better performance when removing some of the features. On the other hand, the Skforecast model improved the results, since it is a more complex algorithm. But these results are worse than the big model.

We can conclude this appendix by saying that in the case of electricity load forecasting, it is more recommendable to build a big and robust model in case the input is detailed, but if the input is more general and not that specific, one should consider smaller models with faster convergence.

TABLE E.2: Metrics of the selected models from the model selection experiment. Hyperparameter tuning of XGBoost and Skforecast algorithms are also included here.

	Linear	XGBoost	Tuned XGBoost	Skforecast	Tuned Skforecast
RMSE	1208.8	1116.0	1004.6	1106.6	964.9
MAE	821.1	850.7	768.9	834.3	729.7
MAPE (%)	3.19	3.32	3.01	3.24	2.83
MAX	7435.4	4565.6	4031.1	4498.9	4071.1

Appendix F

Model explainability

F.1 The maths behind Shapley values

As it is explained in (Caelen, 2022), the idea of Shapley values come from cooperative game theory, where they give a way to do a fair distribution of payoffs to the players (in our case, the players are the variables). In a cooperative game, "players" have the possibility to forge coalitions to achieve a common goal. One difficulty in the theory of cooperative games is the distribution of benefits among the players. Before defining the Shapley values, let us define the four mathematical axioms that were proposed to formally express the problem: efficiency, symmetry, linearity and dummy player. But first, let us introduce some notations:

- N is the set containing all the players/variables.
- S is a subset of N (i.e. $S \subseteq N$).
- i is an element of N (i.e. $i \in N$.
- v is a value function that maps a subset of players S to a real number, such that $v(S) \equiv$ revenue of coalition S. Note that $v(N)$ represents the total value of the grand coalition.

With these definitions, one can define the marginal contribution of a player i when it joins a subset S as $v(S \cup \{i\}) - v(S)$. Finally, the Shapley value ϕ of a player i given the set N and the value function v is defined by $\phi_i(N, v)$.

Now, once these notations are described, the axioms used by the Shapley value can be formally presented in order to define what is a fair distribution of benefits in a coalition:

- Efficiency: All the revenues $v(N)$ of the grand coalition N are redistributed among all the players (no more, no less).

$$v(N) = \textstyle\sum_{i \in N} \phi_i(N, v)$$

- Symmetry: Players i and j who contribute the same to all coalition subsets S receive the same share.

$$\forall S \subseteq N \backslash \{i, j\} : v(S \cup \{i\}) = v(S \cup \{j\}) \Longrightarrow \phi_i(N, v) = \phi_j(N, v)$$

- Linearity: Let (N, v_1) and (N, v_2) be two coalition games, the values from the games can be combined in an additive way. This axiom states that the Shapley value of a new game that is the sum of two games can be calculated by simply adding the Shapley value of each game.

$$\phi_i(N, v_1) + \phi_i(N, v_2) = \phi_i(N, v_1 + v_2)$$

- Dummy player: Those who do not contribute receive nothing.

$$\forall S \subseteq N \backslash \{i\} : v(S \cup \{i\}) = v(S) \implies \phi_i(N, v) = 0$$

It can be shown that, for a coalition game (N, v), the Shapley value is the unique division of the payoff that divides the total payoff $v(N)$ of the grand coalition N and that satisfies the axioms of Symmetry, Linearity and Dummy player. With all that, one can obtain the the Shapley value of player i given a set of players N and a value function v with the following formula:

$$\phi_i(N, v) = \frac{1}{|N|!} \sum_{S \subseteq N \backslash \{i\}} |S|!(|N| - |S| - 1)![v(S \cup \{i\}) - v(S)] \tag{F.1}$$

The Shapley value of player i is a weighted average of the marginal contributions of i over all subsets S of N. The denominator $|N|!$ before the sum is the number of permutations of the set N where $|N|$ is the total number of players. The weight of each marginal contribution is the factor $|S|!(|N| - |S| - 1)!$, that is the product of the number of permutations of S and the number of permutations of the complement of S and i.

F.2 Regularization path

In this section, we will perform L1 regularization for a linear model in order to select the most relevant features, according to the model's criteria. The input of this model will be the same as the second experiment:

- Previous electricity load records from the 28 last days, measured at exactly the same hour than the one that we are going to determine.
- Day of the month.
- Month of the year.
- Week of the year.
- Holiday
- One-hot encoding of the day of the week. That is, we built 6 variables called 'L', 'M', 'X', 'J', 'V', 'S'. Since Sunday is always a holiday, we decided to include it in the holiday feature.
- Temperature. Divided in 5 geographical regions based on climate.
- Radiation. Divided in the same 5 areas as temperature.
- Wind. Weighted average of all provinces based on population.

There are a total of 49 variables, where 28 correspond to the demand lags, 10 are related to the calendar and 21 have to do with the climate. It is a huge number of variables, and for sure some variables are more important than others. To do it, we will add a L1 regularization term to the loss function of the model we are training. In the case of a linear model, it minimizes the residual sum of squares between the

TABLE F.1: Metrics of the linear model depending on the L1 regularization parameter value.

α	**0**	**0.00001**	**0.0001**	**0.001**	**0.01**	**0.1**
RMSE	1208.8	1208.2	1202.3	1335.9	1916.7	4716.1
MAE	821.1	820.7	817.9	952.1	1509.0	3897.3
MAPE (%)	3.19	3.19	3.18	3.71	6.07	16.49
MAX	7435.4	7452.5	7578.0	7648.5	8239.0	12502.0

actual values in the dataset and those predicted by the linear approximation. The expression of the loss including the regularization term is the following:

$$\mathcal{L}(w, x, y, \alpha) = \sum_{i=i}^{n}(y_i - (\sum_{j=1}^{p} w_j x_{i,j} + b))^2 + \alpha \sum_{j=1}^{p} |w_j| \tag{F.2}$$

Where $w = (w_1, ..., w_p)$ and b are the linear coefficient and intercept we are fitting with this method, respectively, y_i is the actual value and x_i the input associated to it.

The primary consequence of L1 regularization is its tendency to force some of the model coefficients to zero, effectively excluding certain features from the model. By pushing coefficients to zero, this regularization performs feature selection, simplifying the model and enhancing its explainability. As long as the L1 parameter (α) increases, accuracy is traded with model interpretability, that can be appreciated in the table F.1, where the first 3 values of α have similar results, but as long as α increases, the model shows poorer performance.

After that, the feature relevance for $\alpha = 0$ (the unrestricted case), $\alpha = 0.0001$ (the model with the best results), and $\alpha = 0.01$ (when the regularization is so strong the model performance gives preference to it instead of converging to an accurate solution) will be shown. To get the feature relevance, one has to simply multiply the fitted coefficient and the standard deviation of the feature, because if the variance of a variable is high then it has more impact on the output variations.

Note that in figures F.1, F.2 and F.3 some labels of the x-axis may be a bit confusing, let us clarify them:

- The first 28 values are the lags in terms of days. They are shown on the upper plot.
- On the lower plot, the variables starting with 'T' are the temperature features. 'TO' = Temperature of Oceanic climate area, 'TN' = Temperature of north Mediterranean climate area, 'TS' = Temperature of south Mediterranean area, 'TA' = Temperature of Arid climate area, 'TC' = Temperature of Continental climate area.
- On the lower plot, as well as with the temperatures, the variables starting with 'R' are the temperature features. 'RO' = Radiation of Oceanic climate area, 'RN' = Radiation of north Mediterranean climate area, 'RS' = Radiation of south Mediterranean area, 'RA' = Radiation of Arid climate area, 'RC' = Radiation of Continental climate area.
- On the lower plot, finally, 'Wi' refers to the wind.

Note also that each point size is directly correlated with the importance of the feature, such that the size of the point is proportional to the square root of the relevance of the variable. With this, it is easier to detect the most important features. In figure F.1 we have the first case, where any feature has been eliminated yet. The most relevant variables are the previous demand records and the calendar features. Having a look at figure F.2, where $\alpha = 0.0001$, notice that some variables have been forced to 0. They are mainly the weather features, since they are divided in 5 regions, making it tougher for the model to select all them, and also did some lags. Finally, in figure F.3 only 5 features left to be eliminated. This features are lags 1, 7, 21, 28 and week. In this case, the regularization term is much more important for the model than the least squares term, so the model uses as less number of variables as possible, but trying to get as many correct predictions as possible. However, as it can be seen in table F.1, the model performance is already much worse than less restrictive algorithms.

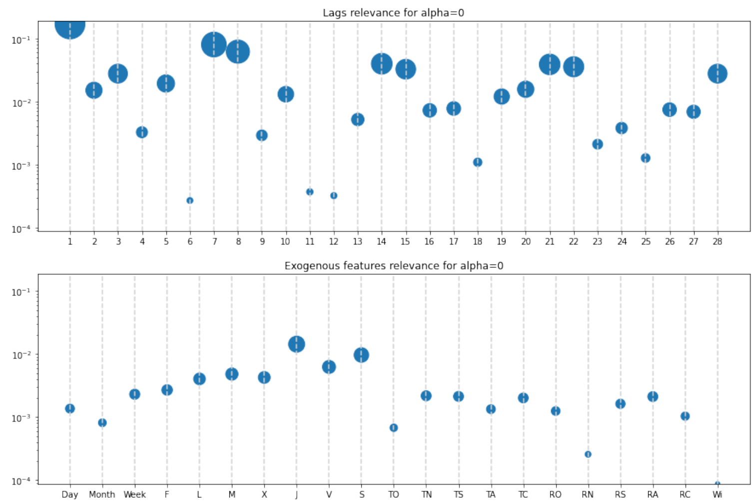

FIGURE F.1: Model coefficients for $\alpha = 0$

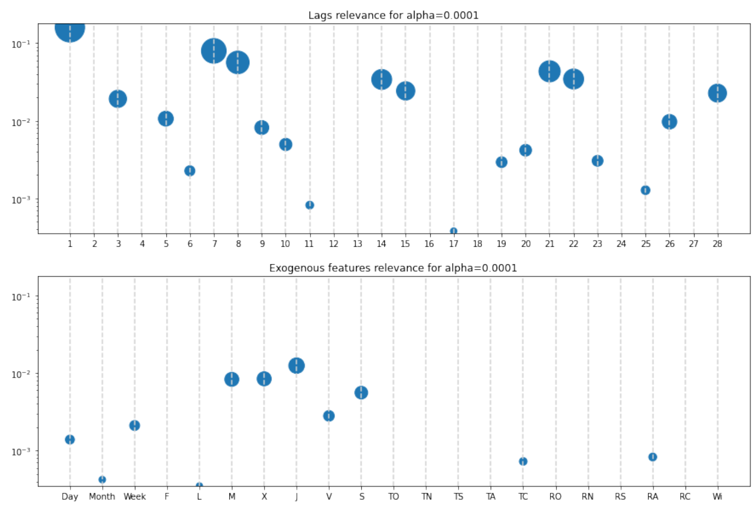

FIGURE F.2: Model coefficients for $\alpha = 0.0001$

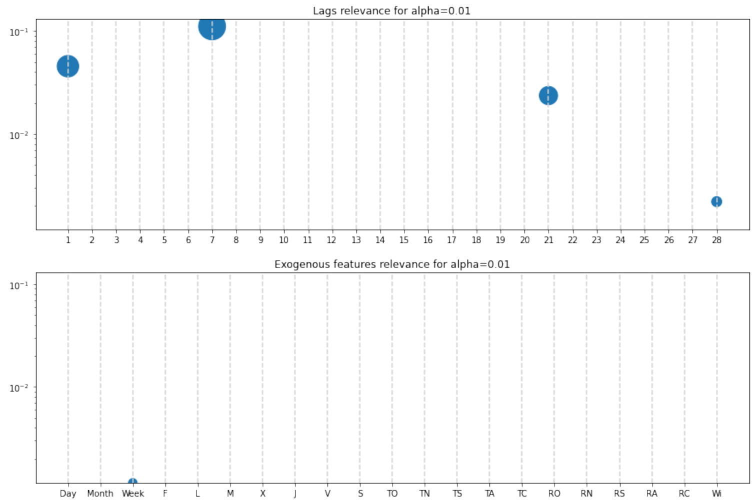

FIGURE F.3: Model coefficients for $\alpha = 0.01$

We can say, after seeing figures F.1, F.2 and F.3, that in cases where there are a lot of features and the model is simple, it is recommendable to perform an L1 regularization in order to select the most relevant features. Sometimes if one directly removes some variables maybe the results improve, because the algorithm pays attention to the important features, enabling it to find a more accurate solution. On the other hand, being too restrictive can cause more features than necessary to be removed, resulting in a sub optimal model that converges to sub optimal solutions because of having a smaller input than the optimal one.

Bibliography

Aisyah, Siti and Arionmaro Asi Simaremare (2021). “Correlation between Weather Variables and Electricity Demand”. In: *IOP Conference Series: Earth and Environmental Science*. Vol. 927. 1. IOP Publishing, p. 012015.

Amat Rodrigo, Joaquin and Javier Escobar Ortiz (Sept. 2023). *skforecast*. Version 0.10.0. DOI: 10.5281/zenodo.8382788. URL: https://skforecast.org/.

— (2024). *Predicción (forecasting) de la demanda energética con machine learning*. https://www.cienciadedatos.net/documentos/py29-forecasting-demanda-energia-electrica-python.html.

Caelen, Olivier (2022). *What is the Shapley value ?* https://medium.com/the-modern-scientist/what-is-the-shapley-value-8ca624274d5a.

Carazo, Fernando and Joaquín Amat Rodrigo (2024). *Deep Learning for time series prediction: Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM)*. https://www.cienciadedatos.net/documentos/py54-forecasting-con-deep-learning.html.

Fathin, Muhammad Ridwan, Yudi Widhiyasana, and Nurjannah Syakrani (2021). “Model for Predicting Electrical Energy Consumption Using ARIMA Method”. In: *2nd International Seminar of Science and Applied Technology (ISSAT 2021)*. Atlantis Press, pp. 298–303.

Fu, Yangyang et al. (2015). “Using support vector machine to predict next day electricity load of public buildings with sub-metering devices”. In: *Procedia Engineering* 121, pp. 1016–1022.

Hammad, Mahmoud A et al. (2020). “Methods and models for electric load forecasting: a comprehensive review”. In: *Logist. Sustain. Transp* 11.1, pp. 51–76.

ORT, Universidad (2024). *Qué es la electricidad y quién la descubrió*. https://fi.ort.edu.uy/blog/que-es-la-electricidad-y-quien-la-descubrio.

Sundararajan, Mukund and Amir Najmi (2020). “The many Shapley values for model explanation”. In: *International conference on machine learning*. PMLR, pp. 9269–9278.

Yukseltan, Ergun, Ahmet Yucekaya, and Ayse Humeyra Bilge (2020). “Hourly electricity demand forecasting using Fourier analysis with feedback”. In: *Energy Strategy Reviews* 31, p. 100524.