Probabilistic Forecasting of Electricity Demand in Colombia

Jennifer Mosquera Cabra

Universidad Nacional de Colombia
Facultad de Ciencias
Departamento de Estadística
Medellín, Colombia
2024

Probabilistic Forecasting of Electricity Demand in Colombia

Jennifer Mosquera Cabra

Trabajo de grado como requisito parcial para optar al título de:
Magister en Estadística

Director:
Profesor Titular, Departamento de Estadística, Víctor Ignacio López Ríos, Ph.D.

Codirector: Profesor Titular, Departamento de Matemáticas y Estadística, Universidad de Antioquia, Santiago Gallón Gómez, Ph.D.

Universidad Nacional de Colombia
Facultad de Ciencias
Departamento de Estadística
Medellín, Colombia
2024

For the defense of love, for continuing to believe in the genuineness of the little things, and for the animals of the world.

Jennifer Mosquera Cabra

Acknowledgments

I would like to thank my advisors for the time they have dedicated to help me and in transmitting their knowledge to me. I would also like to thank my life partner, Adiel Ignacio Restrepo Gil who without his statistical recommendations would not have been the same.

Abstract

New approaches have emerged in the field of uncertainty measurement, offering ways to estimate models and their corresponding confidence levels for point predictions. Our first purpose is to compare the predictive capabilities of some models built for forecasting daily electricity demand in Colombia. Initially, we employ generalized linear models, followed by Machine Learning models such as ensemble learning models, support vector machines (SVM), and finally deep learning models. The goal is to determine which model demonstrates superior predictive accuracy in forecasting daily electricity demand in Colombia. In order to evaluate their performance, we mainly use Mean Absolute Percentage Error (MAPE) as a comprehensive measure, which allows us to evaluate their effectiveness in capturing the actual demand values. And also take into account the mean absolute error (MAE) and the root mean squared error (RMSE).

Next, we turn our attention on the creation of prediction intervals to handle the uncertainty in our forecasts. We use techniques like Bootstrapping to figure out these intervals. We also incorporate conformal prediction to improve the reliability of our intervals. Our prediction intervals are evaluated primarily based on their coverage percentage. This will allow us to see how frequently our prediction intervals correspond to the actual demand from this data. Through this combination of methods, our goal is to establish a robust and user-friendly framework for forecasting daily electricity demand in Colombia.

The results of this development suggest that (1) for the daily energy demand of Colombia, with the variables obtained at a daily frequency, a simple model such as a regularized model works better than an advanced and much more complex model such as a deep learning model. (2) Regarding feature selection concerns, the most important variables are the energy demand lags and demand structure variables for the Lasso model, which works as a feature selection method, due to its regularization nature. This confirms that the inclusion of lags or having an autocorrelated structure is important in this type of problem. Finally, for the forecast intervals, in which we used two methods, the first and most common was the bootstrap method and the second, whose development is more recent, is the conformal Prediction. The construction of our prediction intervals allowed us to give a 99 % confidence level to the point prediction and not just rely on the comparison between the actual and predicted values.

keywords: electricity demand, forecasting, prediction intervals, uncertainty quantification, Bootstrapping, conformal prediction, coverage percentage, time series modeling.

Pronóstico probabilístico de la demanda de electricidad en Colombia

Resumen

Han surgido nuevos enfoques en el campo de la medición de la incertidumbre, que ofrecen formas de estimar modelos y sus correspondientes niveles de confianza para predicciones puntuales. Nuestro primer propósito es comparar las capacidades predictivas de algunos modelos construidos para pronosticar la demanda diaria de electricidad en Colombia. Inicialmente, empleamos modelos lineales generalizados, seguidos de modelos de Machine Learning tales como modelos de aprendizaje ensemble, máquinas de vectores soporte (SVM), y finalmente modelos de aprendizaje profundo. El objetivo es determinar qué modelo demuestra una precisión predictiva superior en el pronóstico de la demanda diaria de electricidad en Colombia. Para evaluar su desempeño se utiliza principalmente el Error Porcentual Absoluto Medio (MAPE) como medida integral, que permite evaluar su efectividad para capturar los valores reales de demanda. También tenemos en cuenta el error medio absoluto (MAE) y el error cuadrático medio (RMSE).

A continuación, centramos nuestra atención en la creación de intervalos de predicción para manejar la incertidumbre de nuestras previsiones. Utilizamos técnicas como Bootstrapping para calcular estos intervalos. También incorporamos la predicción conforme para mejorar la fiabilidad de nuestros intervalos. Nuestros intervalos de predicción se evalúan principalmente en función de su porcentaje de cobertura. Esto nos permitirá ver con qué frecuencia nuestros intervalos de predicción se corresponden con la demanda real a partir de estos datos. Mediante esta combinación de métodos, nuestro objetivo es establecer un marco robusto y fácil de usar para la predicción de la demanda diaria de electricidad en Colombia

Los resultados de este desarrollo sugieren que (1) para la demanda diaria de energía de Colombia, con las variables obtenidas a una frecuencia diaria, un modelo simple como un modelo regularizado funciona mejor que un modelo avanzado y mucho más complejo como un modelo de aprendizaje profundo. (2) En cuanto a las preocupaciones de selección de características, las variables más importantes son los rezagos de demanda de energía y las variables de estructura de demanda para el modelo Lasso, que funciona como método de selección de características, debido a su naturaleza de regularización. Esto confirma que la inclusión de retardos o tener una estructura autocorrelacionada es importante en este tipo de problemas. Por último, para los intervalos de predicción, en los que utilizamos dos métodos, el primero y más común fue el método bootstrap y el segundo, cuyo desarrollo es más reciente, es la Predicción conforme. La construcción de nuestros intervalos de predicción nos permitió dar un nivel de confianza del 99 % a la predicción puntual y no basarnos únicamente en la comparación entre los valores reales y los predichos.

Palabras clave: demanda de electricidad, previsión, intervalos de predicción, cuantificación de la incertidumbre, Bootstrapping, predicción conforme, porcentaje de cobertura, modelización de series temporales.

Acronyms

ARIMA	Autoregressive Integrated Moving Average
B-SCN	Boosting Stochastic Configuration Network
CQF	Conditional Quantile Function
CP	Conformal Prediction
DL	Deep Learning
ELM	Extreme Learning Machine
IVMD	Variational Mode Decomposition
LSTM	Long Short-Term Memory Neural Networks
MAPE	Mean Absolute Percentage Error
MAE	Mean Absolute Error
ML	Machine Learning
MLP	Multi-Layer Perceptron
NN	Neural Network
OLS	Ordinary Least Squares
PIs	Prediction Intervals
QR	Quantile Regression
QRA	Quantile Regression Average
RNN	Recurrent Neural Networks
SARIMA	Seasonal autoregressive integrated moving average
SVR	Support Vector Regression
ReLU	Rectified Lineal Unit

Content

1. Introduction

In order to understand and predict energy demand, numerous efforts have been made to develop effective models that can capture the complexity of some factors like climate, social dynamics, and economic influences. It is essential to consider how frequently we create these models since it impacts how we manage and select the data (Nowotarski and Weron, 2018). Researchers have proposed different types of models, from traditional to advanced methods, all aimed at capturing complex relationships or patterns. The goal is to better understand how different variables interact within the context of energy demand and how different models could have a better predictive performance.

To forecast electricity demand, a variety of models are employed, and these can be classified according to two groups. Firstly, under the category of Statistical Models or traditional models, we find time series models such as autoregressive integrated moving average (ARIMA), seasonal autoregressive integrated moving average (SARIMA), unobservable Components models, additive component models, multivariate regression models designed for both short- and long-term forecasting and holt winters multiplicative (or Additive) models. Lastly, in the context of Deep Learning (DL) models, this includes neural networks, which exhibit the capacity to capture various facets of the data, revealing intricate relationships (Marino et al., 2021).

Among the several methods developed to model or closely estimate energy demand, a new area of research has emerged in the domain of probabilistic predictions. Probabilistic forecasting, unlike single-point forecasting, involves many methods that allow for predicting the likely range of outcomes instead of a single specific value in the future. This forecasting technique offers more comprehensive information by showing a spectrum of potential values within which the actual value could occur, enabling the calculation of prediction intervals. Notably, one of the seminal works in probabilistic prediction pertains to the estimation of electricity prices, as exemplified by Zhao et al. (2008). This study used the support vector machines (SVM) methodology, allowing the computation of Prediction Intervals (PIs) to capture forecast uncertainty. Similarly, Huurman et al. (2012) introduced the concept of generalized autoregressive conditional heteroskedasticity (GARCH) time-varying volatility models, providing a framework to quantify uncertainty through density forecasts. These models contribute to a comprehensive understanding of the intricate dynamics in electricity price prediction.

This project makes the following contributions: firstly, it provides a comprehensive examination of energy demand in a daily frequency in Colombia. This inclusive approach offers detailed insights into daily patterns and nationwide trends, fostering a more holistic understanding of demand dynamics. Secondly, the project distinguishes itself by identifying the influencing factors affecting demand behavior, covering both accessible and modifiable elements. This identification is pivotal for enhancing forecast accuracy and decision-making processes. Additionally, the project conducts a thorough review and practical application of methods for constructing probabilistic forecasts, offering valuable insights into the associated uncertainties of energy demand. This contribution improves our understanding of energy demand forecasting, establishing a solid foundation for future research and strategic decision making.

In the subsequent chapters, the project develops systematically, starting with the theoretical framework. The initial section explores the background of energy demand forecasting and conducts a comprehensive review of probabilistic forecasting. Following this, the third chapter provides a detailed description of the dataset, presenting relevant variables and offering a thorough descriptive analysis. Subsequently, the fourth chapter introduces the trained models, categorized into statistical models including generalized linear models, also categorized into machine learning models as ensemble methods, support vector machines, and deep learning models. And then we present the model selection process, incorporating backtesting and presenting model results.

The fifth chapter delves into the critical aspect of prediction intervals. This section investigates the construction of prediction intervals through bootstrap and conformal prediction methods. An evaluation of these prediction intervals is performed, focusing on the presented results. In the final chapter, concluding remarks summarize key findings, providing insights derived from the conducted analyses.

2. Theoretical framework

2.1. Energy demand forecast background

In their article, Fabbiani et al. (2021) conducted a comparison of diverse approaches for predicting gas demand, subsequently categorizing them into three main classifications. In the category of linear models, they employed techniques such as ridge regression, lasso, support vector regression (SVR), and elastic net. For non-linear modeling, random forest and neural networks were employed, whereas nearest neighbor and gaussian processes were applied to non-parametric models. Expanding on those separate models, the researchers created four combined predictors: The study considers simple average, weighted average, average of subsets and SVR aggregation. Grouping predictors has been more accurate than predicting with individual predictors. This suggests major improvement can be attained by careful grouping of predictors for forecasting of the gas useage.

In their study, Li et al. (2020) introduced a detailed multistage approach covering several forecasting time periods and yearly cycle configurations. They used sample entropy to measure data complexity, thus improving their comprehension of the inherent attributes of electricity demand. Secondly, the authors incorporated variational mode decomposition as an instrument for reducing noise, allowing them to study high variance and the inclusion of relevant variables. In this study, different methodologies are included such as SVR, multivariate adaptive regression splines (MARS), ARIMA, autoregressive-based time-varying models, variational mode decomposition, fast fourier transform, and intelligent optimization algorithms.

The investigation by Jiménez et al. (2019) involved a comprehensive statistical analysis of time series data concerning national electricity demand and consumption in the caribbean region. They used multivariate statistical techniques in their research, focusing on factor analysis. This analytical approach was employed to reduce the dimensionality of the dataset by identifying and associating variables exhibiting high levels of correlation. As a result of this analysis, the researchers established a neural network model. The research employed a variety of methodological sources. These methodologies included principal components analysis, exponential smoothing, artificial neural networks (ANN), support vector machines (SVM), fuzzy logic, adaptive neuro fuzzy inference system, hybrid models, autoregressive moving average models, and grey Markov.

The deep learning-based framework, as presented by Bedi and Toshniwal (2019), significantly improves electricity demand forecasting by effectively exploring long-term historical dependencies. This approach employs cluster analysis to segment consumption data based on seasonal patterns and characterizes load trends to gain deeper insights into influencing factors. The use of a multi-input multi-output long short-term memory (LSTM) network model enhances the precision of predictions. This framework demonstrated its superiority over several state-of-the-art prediction models, including ANN, recurrent neural networks (RNN), and SVR models when it was applied to electricity consumption data from Union Territory Chandigarh, India.

In Al-Musaylh et al. (2018) was presented a study focusing on short-term electricity demand forecasting. The study's main objective is to establish accurate and reliable models for electricity demand forecasting, with a specific emphasis on the utility of statistical models. These models, including ARIMA and multivariate adaptive regression splines (MARS), which offer valuable insights for renewable and conventional energy engineers, electricity providers, end-users, and government entities. And also include the SVR model, the study employs data from Queensland, Australia, sourced from the Australian Energy Market Operator (AEMO) database, and predictor variables that include lagged combinations of electricity demand data.

The study conducted by Son et al. (2020) emphasizes the importance of precise electric power load forecasting, essential for ensuring the stability of power supply and optimizing operations for power generation and distribution companies. The research introduces Deep Learning (DL) techniques, specifically deep neural network (DNN) and LSTM, exploring their potential applications in load forecasting. By comparing the performance of DNN and LSTM models in load forecasting, the study evaluates the quality of the predictions using some metrics, including mean absolute error (MAE), root mean squared error (RMSE), coefficient of variation RMSE (CVRMSE), mean absolute percentage error (MAPE), determination coefficient, and computation time. Furthermore, the research explores an analysis of power consumption patterns among different companies and their impact on load forecasting. In conclusion, the study proposes a multivariate model for load forecasting, using the capabilities of DNN and LSTM.

The main objective of the study by Ullah et al. (2022) is to enhance the accuracy of short-term power load forecasting through the application of DL techniques. The authors introduce a novel method that combines deep convolutional LSTM and stacked gated recurrent unit (GRU) models, enabling the capture of temporal dependencies and patterns within energy load data. Notably, this proposed approach demonstrates superior performance compared to existing methods, as indicated by some prediction error metrics such as MAE, mean squared error (MSE), and the determination coefficient.

2.2. Probabilistic forecasting's review

Probabilistic forecasting, in contrast to point forecasting, encompasses a set of techniques that facilitate predicting the expected distribution of outcomes rather than a single future value. This forecasting approach offers more comprehensive information by outlining the range of probable values within which the true value may lie, enabling the estimation of prediction intervals (Amat Rodrigo and Escobar Ortiz, 2023).

The article by Nowotarski and Weron (2018) emphasizes the significance of probabilistic forecasting within the energy industry. It provides valuable insights into the effective usage of probabilistic forecasting, offering guidelines and recommendations. The study also offers an up-to-date overview of recent advancements in electricity price forecasting. Many models are explored, including autoregressive (AR), ARIMA, generalized autoregressive conditional heteroskedasticity (GARCH), SVR, ANN, and LSTM models. These models are examined in the context of critical variables such as historical electricity prices, weather data, demand data, generation capacity data, fuel prices, economic indicators, regulatory policies, and market structure and competition.

In their research, Nowotarski and Weron (2018) expressed the actual dependent variable at time t as $y_t = \widehat{y_t} + \varepsilon_t$ where $\widehat{y_t}$ is the point forecast at time t and ε_t is the associated error term. This framework moved beyond conventional point forecasting by extending its scope to construct PIs. The objective of these PIs, set at a $(1 - \alpha)$ confidence level, was to define upper and lower bounds for y_t. These bounds were established based on the $\frac{\alpha}{2}$ and $(1 - \frac{\alpha}{2})$-th quantiles of the cumulative distribution function (CDF) of ε_t. The resulting PIs were denoted as $\left[\widehat{L}_t, \widehat{U}_t\right]$, where $\widehat{L}_t$ and $\widehat{U}_t$ are the lower and upper bounds, respectively.

The concept of probabilistic forecasting, as articulated by Gneiting and Katzfuss (2014), centers on the establishment of probability distributions to describe future quantities. Moreover, the research of Weron and Misiorek (2008) and Maciejowska et al. (2016) emphasize the significance of point forecasts and the concurrent error distribution. The research conducted by Jensen et al. (2024) presented a comprehensive framework for constructing distribution-free valid PIs. This methodology is particularly adapted to address the challenges posed by non-stationary and heteroscedastic time series data. A noteworthy aspect of this approach is its applicability across some forecasting models, including DL structures trained on extensive data squences.

Guo et al. (2018) present in their study a comprehensive approach to short-term electricity load forecasting and probability density forecasting. Their main objectives include identifying key factors that influence accurate forecasting, assessing the DL model's performance using some metrics, and comparing the proposed method with established forecasting techniques. The models developed in their research encompass a deep feedforward network and a probability density forecasting method based on deep learning, quantile regression, and kernel density estimation. Additionally, their study incorporates random forest and gradient boosting ML models. The considered variables span a range of factors, including date-related elements such as monthly, daily, and seasonal data, air-quality-related factors (e.g., PM2.5, SO2, CO), weather-related components (including rainfall levels and daily temperatures), and economic factors such as the consumer price index (CPI).

In the effort to mitigate the small-sample bias inherent in least-squares estimation, Clements and Kim (2007) considered in their study three distinct methods. Firstly, introduced a bootstrap bias-corrected ordinary least squares (OLS) estimator. Additionally, the researchers explored the application of the Roy–Fuller estimator and the Andrews–Chen estimator as alternatives to the conventional OLS method. Their findings underscore the efficacy of bootstrap PIs based on the Roy–Fuller estimator, which consistently outperformed the other techniques.

Masarotto (1990) embarked on the calculation of bootstrap PIs for stationary autoregression models, with a specific focus on the concept of memory. This concept concerns whether there exists a subtle effect that maintains a lasting influence on the future predictions of a time series over an extended temporal horizon. In scenarios involving time series models driven by non-parametric noise, the determination of the lower and upper bounds for distribution-based PIs involves the computation of quantiles derived from the kernel estimator of the Probability Density Function (PDF) of ε_t. This approach aligns with the methodologies established by Weron and Misiorek (2008), Nowotarski and Weron (2013), Misiorek et al. (2006), and Panagiotelis and Smith (2008).

Khosravi et al. (2013) adopted a two-step process: firstly, employing a Neural Network (NN) for point forecasting with k-fold cross-validation, and secondly, implementing the bootstrap method to construct 90 % PIs. Efron and Tibshirani (1993) highlighted several advantages of the bootstrap method when compared to classical techniques for calculating PIs in the context of stationary processes. Classical methods often hinge on the assumption of a normal or known error distribution, which may not hold true in practical scenarios. Bootstrap PIs, on the other hand, circumvent this limitation by generating pseudo-data sets through repeated resampling of residuals, using the estimated model.

It is important to mention that the accuracy of bootstrap Prediction Intervals (PIs) can be negatively influenced by the small-sample bias linked with OLS estimation. This is especially notable when working with limited data and highly persistent or long-memory processes, as noted by Clements and Kim (2007). To address this concern, Kilian (1998) introduced the concept of "Bootstrap-after-bootstrap", emphasizing a bias correction approach. One of the key advantages of bootstrap PIs over distribution-based PIs is their ability to account for historical forecast errors and parameter uncertainty simultaneously. Bootstrap intervals do not rely on assumptions about the distribution of the time series and encompass the variability stemming from parameter estimation, as emphasized by Pascual et al. (2005). This feature enhances the robustness and versatility of bootstrap PIs in practical forecasting applications.

In their study, Gao et al. (2022) addressed the dual objectives of point forecasting and uncertainty analysis within the domain of electricity demand prediction. Their approach incorporated several key elements, starting with the use of Pearson's correlation coefficient which is used in the study to perform feature dimensionality reduction. Through Pearson's coefficient, input characteristics that significantly affect electricity demand, such as socioeconomic and meteorological factors, are filtered and reduced in order to improve forecasting efficiency. The researchers introduced the variational mode decomposition (IVMD), an approach optimized by the sparrow search algorithm (SSA), to decompose electricity demand series into distinct subsequences. This segmentation of data facilitated a more granular analysis and forecasting of demand dynamics. Furthermore, a hybrid forecasting model, denoted as IELM-Adaboost, was devised by synergistically combining the sparrow search algorithm (SSA), ELM, and the adaptive boosting algorithm (Adaboost).

In their study, Ning et al. (2022) introduced a probabilistic forecasting approach based on the boosting stochastic configuration network (B-SCN). Their methodology encompassed a comprehensive correlation analysis involving multidimensional input parameters. To enhance model stability and construction, they introduced an adaptive B-SCN network architecture. The creation of confidence intervals was facilitated through the application of the gaussian process. Their research yielded notable outcomes, indicating that the proposed boosting-SCN prediction model outperforms both the SCN model and other conventional forecasting models in terms of forecasting accuracy. In addition, their findings highlight the effectiveness of probabilistic forecasting in capturing uncertainties in electrical data.

3. Data description

3.1. Variables

In this project, the variables were categorized according to two perspectives. The first category focused on factors such as demand behavior and temporal patterns. Simultaneously, the second category honed in on weather data, encompassing meteorological parameters. This classification set the stage for a comprehensive analysis, facilitating a deeper understanding of the intricate dynamics influencing energy demand. The electricity demand and calendar data were sourced from XM, which is the company specialized in the management of real-time systems, wholesale energy market administration and the development of energy and information solutions and services in Colombia[1]. This data is collected on a daily basis, the variables were derived, including demand trends, variations in the series, and a COVID-19 variable indicating the period when the presence of COVID-19 had a significant impact. Additionally, weather variables were obtained from IDEAM, the Institute of Hydrology, Meteorology, and Environmental Studies, a government entity in Colombia under the Ministry of Environment and Sustainable Development[2]. The date range comprising the entire dataset is from december 1, 2017 through september 30, 2023.

3.2. Descriptive analysis

In Figure **3-1**, the time series of national electricity demand, also known as the Sistema Interconectado Nacional (SIN), is presented. Two significant aspects of the series behavior can be inferred. Firstly, there is a noticeable level change in the trend of the series after the onset of the COVID-19 pandemic. Additionally, a pronounced decrease in demand during the COVID-19 period is evident. The series also exhibits a discernible seasonal pattern characterized by a weekly seasonal pattern. More details of the structure of the electricity demand series will be given below.

[1]See: `https://www.xm.com.co/consumo/informes-demanda/indicadores-de-pronosticos-oficiales-de-demanda`
[2]See: `http://www.ideam.gov.co/`

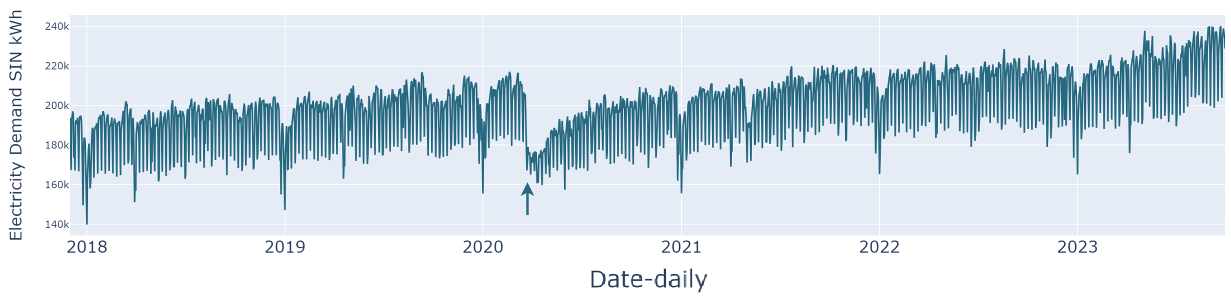


Figure 3-1.: Electricity demand from december 1, 2017 through september 30, 2023 SIN kWh.

When visually analysing the energy demand series, it can be observed that there is a point in time where the variability of the data changes significantly. In this period there was low volatility or low variance. This change may indicate the presence of a level change in the series, which could imply that the variance is not constant over time. This translates into the need to use some transformation to stabilize variance in a parametric model.

The Figures **3-2**, **3-3** illustrate the seasonality pattern of energy demand. The analysis suggests that the months characterized by higher demand are january, august and september, while wednesdays, fridays and thursdays exhibit the highest daily demand. It is noteworthy that sunday registers the lowest demand, potentially attributed to its non-working or non-industrial nature. This observation underscores the importance of considering both temporal and weekly patterns in demand, shedding light on potential influences such as workdays and industrial activities.

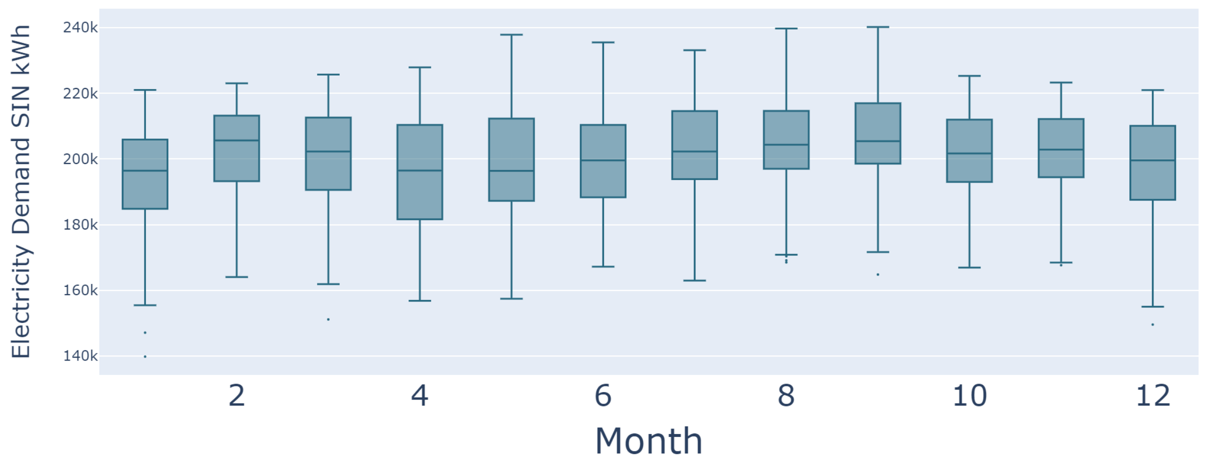


Figure 3-2.: Boxplot figure of electricity demand from december 1, 2017 through september 30, 2023 SIN kWh, on a monthly basis.

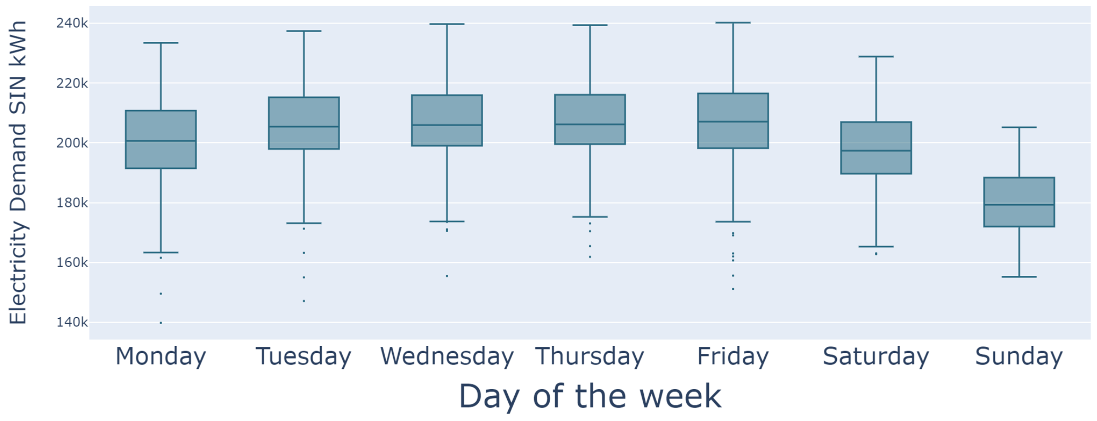


Figure 3-3.: Boxplot figure of electricity demand from december 1, 2017 through september 30, 2023 SIN kWh, for weekdays.

Figure **3-4** presents the autocorrelation function (ACF), a key tool in the statistical analysis of time series. The peaks highlighted in the ACF point to remarkable correlations at regular intervals, revealing weekly seasonal pattern in demand. This observation suggests the existence of periodicity, specifically, a possible weekly repetition in the data. The slow decline in ACF provides further evidence of a significant seasonal trend or pattern in the time series studied. Which implies that the strength of time dependencies does not go fast to zero, which is characteristic of non-stationary processes.

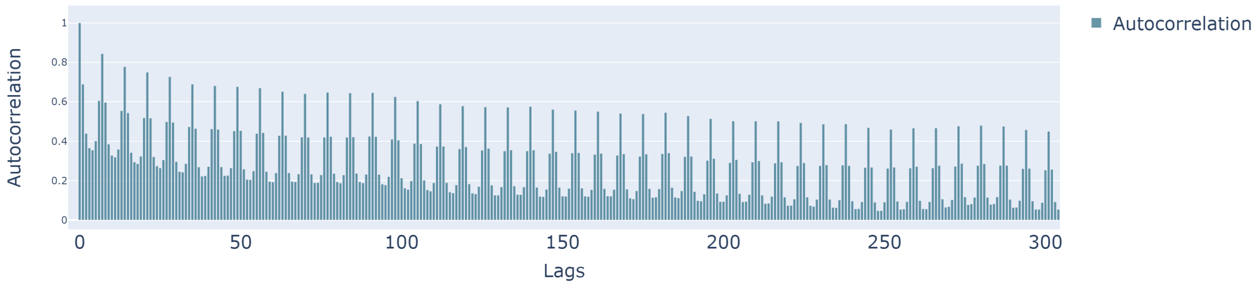


Figure 3-4.: Autocorrelation function (ACF) of electricity demand SIN kWh.

The Figure **3-5** shows the energy demand on the days of the week. From this figure you can see the behavior of the demand on these days and it is evident that the days with the highest demand are the days of the work week and also just after a monday holiday. The day type variable [3] has forty-one categories as: 1st may, 2nd january, 20th july, 24th december, 25th december, 31st december, 7th august, 8th december, sundays before a holiday monday, sundays before a holiday, monday in january, sunday, sunday of december vacation, sunday

[3]See: A.1 for the nomenclature of the categories of the day type variable.

of january vacation, easter sunday, easter thursday, thursday, thursday holidays in december, thursday holidays in january, holiday mondays, holiday mondays in january, monday, monday of december vacation, monday of january vacation, tuesdays after a holiday monday, tuesdays , tuesdays of december vacation, tuesdays of january vacation, wednesday, wednesday of december vacation, wednesday of january vacation, easter wednesday, saturdays before a holiday monday, saturdays before a holiday monday in january, saturdays, saturdays of december vacation, saturdays of january vacation, easter saturday, friday, friday of december vacation, friday of january vacation and easter friday.

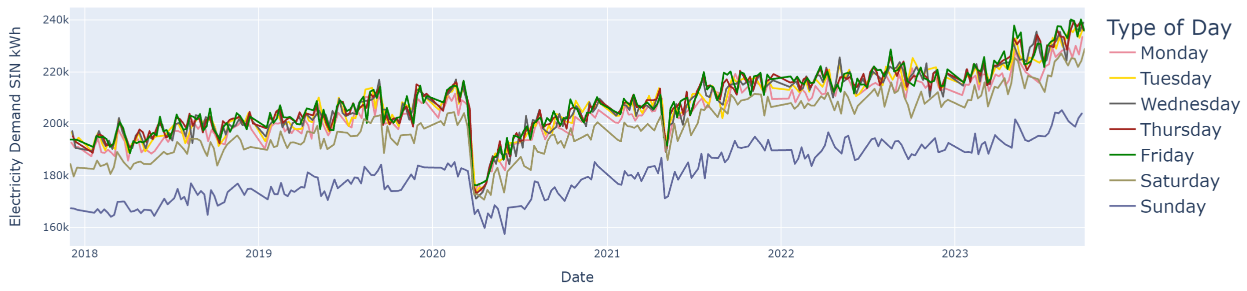


Figure 3-5.: Electricity demand from december 1, 2017 through september 30, 2023 SIN kWh by Type of weekdays.

The Figure **3-6** shows the days on which dates are culturally celebrated in Colombia, such as: 8, 24 and 31 december, the first two days of the year, the first of may which is International labour day, 20 july which is Colombia's independence Day and 7 august which is the battle of Boyacá. From the figure it is not possible to infer a common trend between these days, but it could influence demand.

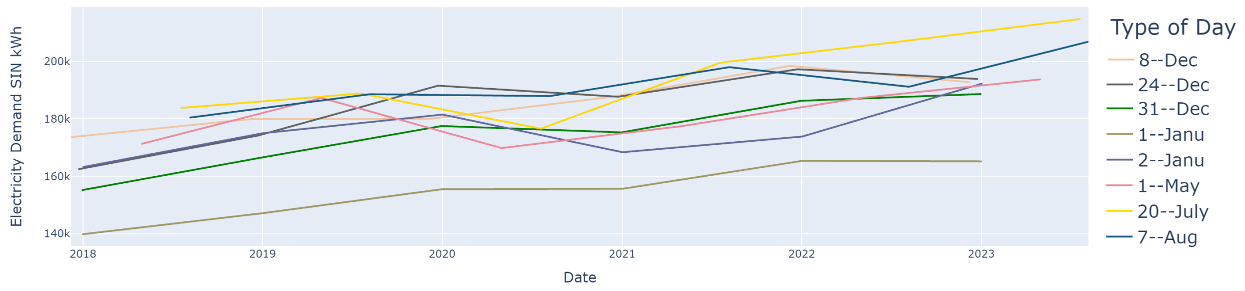


Figure 3-6.: Electricity demand from december 1, 2017 through september 30, 2023 SIN kWh by Type of cultural days in Colombia.

The Figure **3-7** shows the days that are holidays during holy week, allowing to indicate this period of the year. The Figure **3-8** reflects the behaviour of energy demand on holidays and also on days close to a holiday. It can be seen that the days that are close to a holiday, such as the tuesday before a monday holiday, the saturday before a monday holiday, the sunday before a holiday and the sunday before a monday holiday in january have a similar behaviour, which could indicate that this variable could capture some fluctuation in demand.

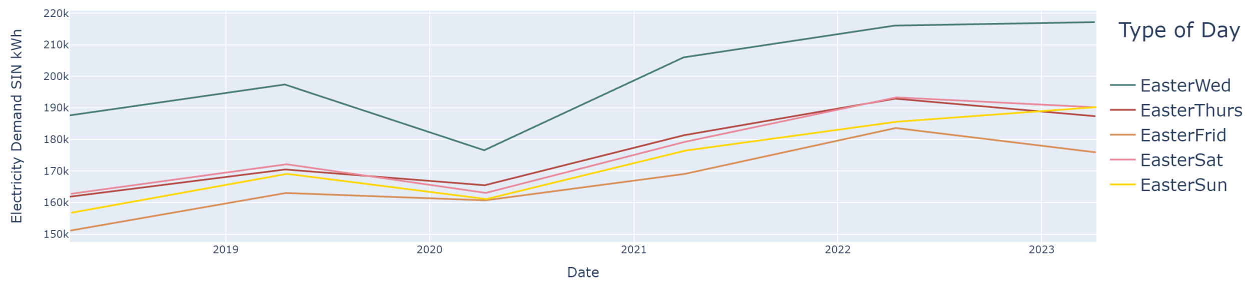


Figure 3-7.: Electricity demand from December 1, 2017 through september 30, 2023 SIN kWh by Type of holy week days.

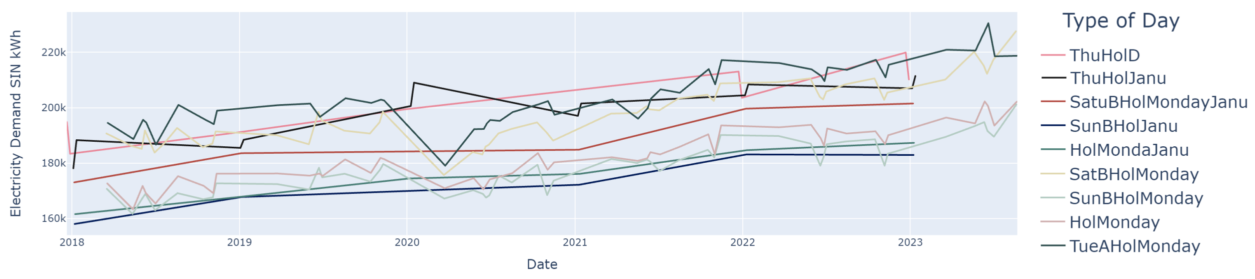


Figure 3-8.: Electricity demand from december 1, 2017 through september 30, 2023 SIN kWh by Type of holidays and also on days close to a holiday.

The Figure **3-9** is the last of the types of days which captures some additional days that are considered as holidays in the year, such as a friday in december or a saturday in december, among others. In Figure **3-10** shows the electricity demand indicating the period that was taken into account to make the COVID-19 period explicit. This was specified from march 19, 2020, when quarantine was decreed until july 23, 2021 because it can be seen that until july 23, 2021, the series recovers and returns to the point where the pandemic began.

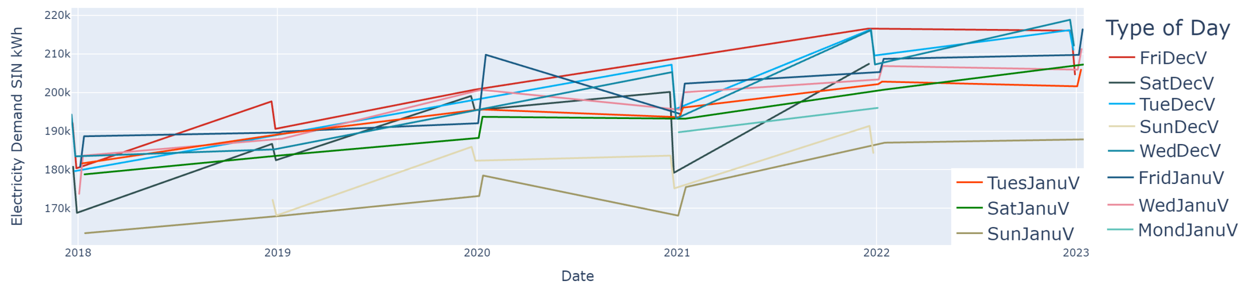


Figure 3-9.: Electricity demand from december 1, 2017 through september 30, 2023 SIN kWh by type of holidays.

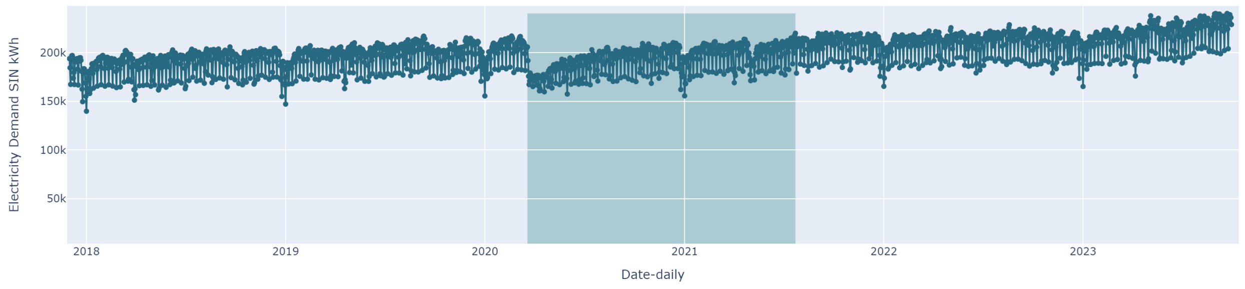


Figure 3-10.: Electricity demand from december 1, 2017 through september 30, 2023 SIN kWh, indicating the COVID-19 period.

In Figure **3-11** shows the electricity demand indicated since the period in which the level change in the series occurred, since the trend change or in other words, the slope change. The above originates from the date when COVID-19 began.

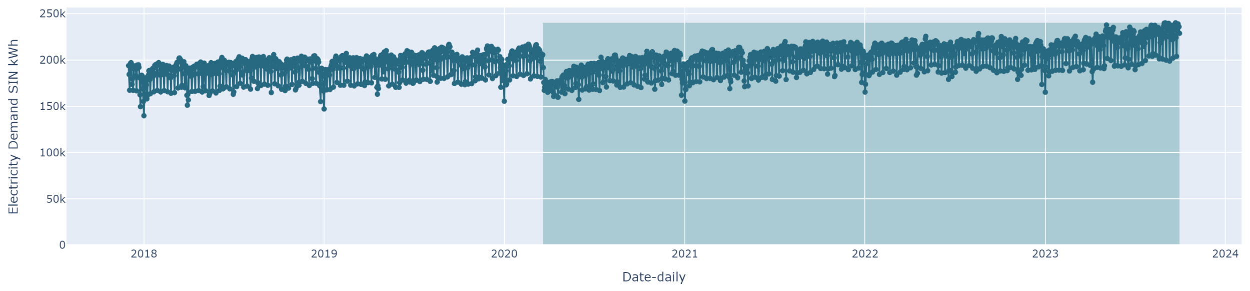


Figure 3-11.: Electricity demand from december 1, 2017 through september 30, 2023 SIN kWh, indicating the Period of level change.

With respect to the climatic variables, Figures **3-12** and **3-13** show the precipitation and maximum daily humid temperature respectively, which according to the literature, influence the demand for electricity. Due to missing values in the daily maximum temperature from august 1st to 30 september 2023, and daily total precipitation from august 3rd onwards, a procedure was created to impute these values with an average between a sixty-day window of time. If there is a null value in a given row, the average of the non-null values in a range of sixty values is calculated. If the index of the current row is greater than or equal to, the average of the previous sixty values is calculated. If the index is less than sixty, the average of all rows before the current row is calculated. Finally, in order to have a practical approach oriented towards simplicity and efficiency in the imputation process, the null value is replaced with the calculated mean as a starting point. Importantly, this decision does not rule out the relevance of addressing seasonal effects or more advanced imputes in subsequent analyses, which could enrich the understanding of climate variability and its impacts. This procedure is shown in Figure **3-14**.

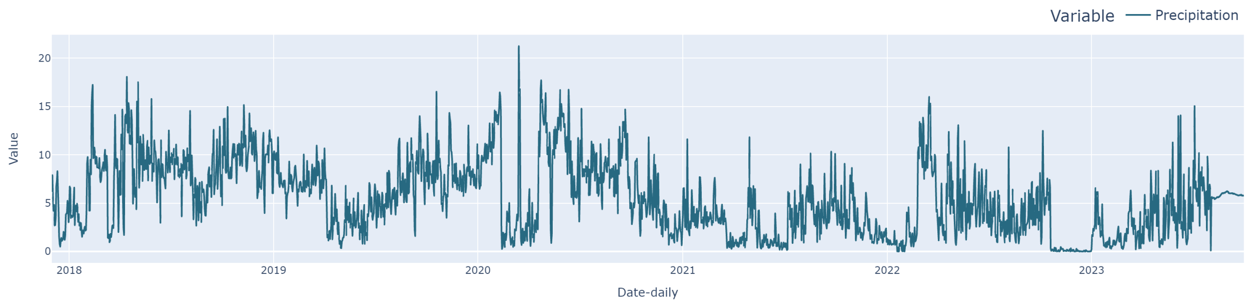


Figure 3-12.: Daily precipitation from december 1, 2017 through september 30, 2023.

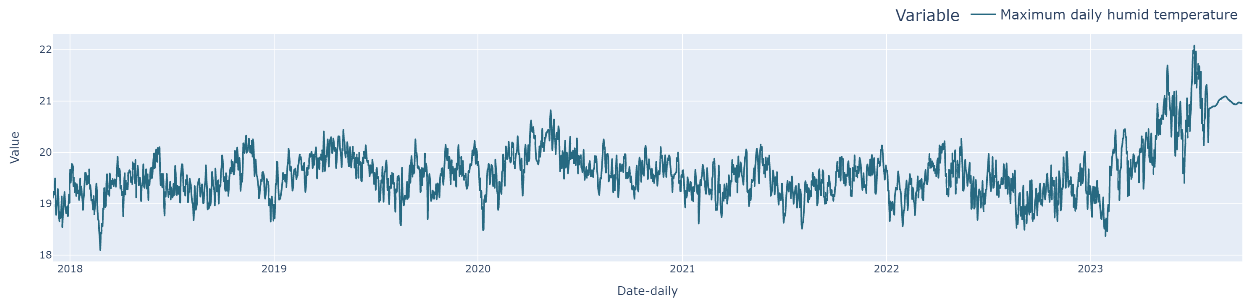


Figure 3-13.: Maximum daily humid temperature, from december 1, 2017 through september 30, 2023.

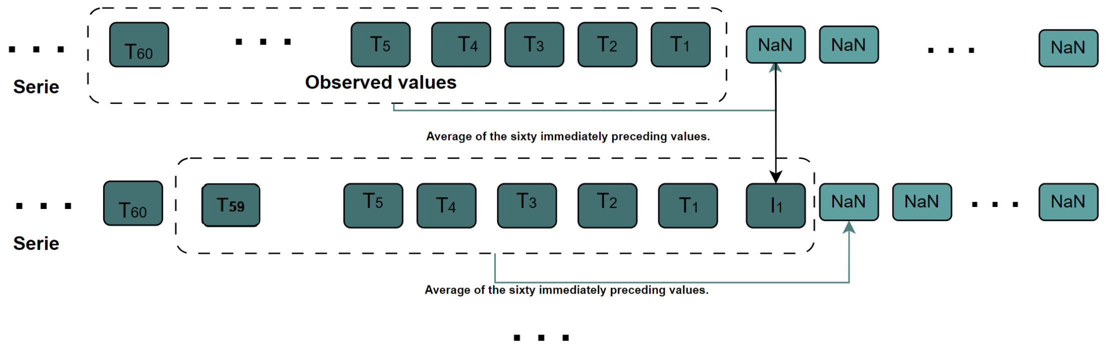

Serie
T60
T5
T4
T3
T2
T1
Observed values
NaN
NaN
NaN
Average of the sixty immediately preceding values.
Serie
T60
T59
T5
T4
T3
T2
T1
I1
NaN
NaN
NaN
Average of the sixty immediately preceding values.

Figure 3-14.: Imputation procedure with the average of a time window for the climatic variables.

3.3. Variable selection

The procedure for selecting the variables was as follows: A multiple linear model was fitted to see a first sighting of the relationship between calendar variables such as day type, COVID-19 epoch, the variable indicating the level change that occurred after the COVID-19 and the weather variables. The Equation (3-1) shows the model considered to have a first perception of the significance of the variables in the variation of energy demand.

$$y_t = \beta_0 + \sum_{j=1}^{7} \beta_j y_{t-j} + \sum_{j=2}^{4} \beta_{6+j} y_{t-7j} + \sum_{j=1}^{14} \alpha_j x_{tj} + \sum_{j=0}^{2} \delta_j \mathrm{D_{tj}} + \varepsilon_t, \quad t = 1, \ldots, T. \tag{3-1}$$

where ε_t is a random error term, with $E(\varepsilon_t) = 0, Var(\varepsilon_t) = \sigma^2$. The first seven lags, and from the seventh lag, lags multiplied by seven up to lag 28 were taken into account, as it is a series with weekly seasonality. The climatic variables, air relative humidity at two meters minimum daily (x_{t1}), air relative humidity at two meters maximum daily (x_{t2}), air relative humidity at two meters mean daily (x_{t3}), maximum daily wet temperature (x_{t4}), minimum daily wet temperature (x_{t5}), mean daily dry temperature (x_{t6}), minimum daily dry temperature (x_{t7}), maximum daily dry temperature (x_{t8}), mean daily dew point temperature (x_{t9}), mean daily wind speed (x_{t10}), daily persistence (x_{t11}), maximum daily precipitation (x_{t12}), total daily precipitation (x_{t13}) and total daily solar brightness (x_{t14}). And finally, a categorical variable denoted as $\mathrm{D_{ot}}$ is used, which represents the type of day and takes the value of 1 in each corresponding category and 0 otherwise. Likewise, another indicator variable denoted as $\mathrm{D_{1t}}$ is used to represent the period of greatest pandemic confinement, shown in Figure **3-10**, which takes the value of 1 in the specified period and 0 otherwise. And finally, there is another dummy variable denoted as $\mathrm{D_{2t}}$ representing the level change that has arisen since the pandemic, shown in Figure **3-11**, which takes the value of 1 from the pandemic which was from march 19, 2020.

The independent variables considered to adjust the multiple linear regression were all those described above, but finally the variables selected for their significance were: The day type variable have p-values close to zero, suggesting that they are statistically significant in predicting the demand, with at least one significant category in the day type variable, the variable is already significant in explaining model variability, the first seven lags and the multi-seasonal lags up to lag twenty-eight. A dummy indicating the COVID-19 period, another dummy indicating the level change. This reaffirms what is mentioned in Figures **3-10** and **3-11**. The Table **3-1** shows the variables with their respective p-values that were significant in this first approach with the multiple linear regression adjustment.

Variable	**Coefficient**	**P-value**
Demand lag 1	0.5690	0.000
Demand lag 2	0.0162	0.000
Demand lag 3	0.0521	0.000
Demand lag 4	0.0379	0.002
Demand lag 5	0.0513	0.000
Demand lag 6	0.0305	0.012
Demand lag 7	0.0581	0.000
Demand lag 14	0.0195	0.022
Demand lag 21	0.0390	0.000
Demand lag 28	0.0233	0.002
Type day 2 January	30344.264	0.000
Type day 1 May	13167.588	0.000
Type day 24 December	14485.944	0.000
Type day 31 December	13661.201	0.000
Type day 20 July	18410.364	0.000
...		
Covid	-2077.4939	0.000
Level change	1229.0048	0.000
Air relative humidity at two meters minimum daily	-11.1740	0.300
Air relative humidity at two meters maximum daily	0.6855	0.141
Air relative humidity at two meters mean daily	0.9632	0.284
Maximum daily wet temperature	2174.8138	0.014
Minimum daily wet temperature	-2729.5002	0.000
Mean daily dry temperature	-1086.2311	0.223
Minimum daily dry temperature	3486.4075	0.000
Maximum daily dry temperature	-504.8986	0.322
Mean daily dew point temperature	243.2641	0.806
Mean daily wind speed	-442.5085	0.026
Daily persistence	-4.0424	0.567
Maximum daily precipitation	-1003.4236	0.000
Total daily precipitation	96.7449	0.001
Total daily solar brightness	1022.4075	0.000

Table 3-1.: Coefficients adjusted for each variable with their respective p-values.

The reduced model is shown in Equation (3-2), where the variables chosen to define the reduced model are: The maximum daily wet temperature renowned for simplicity (x_{t1}), minimum daily wet temperature renowned for simplicity (x_{t2}), minimum daily dry temperature renowned for simplicity (x_{t3}), mean daily wind speed renowned for simplicity (x_{t4}), maximum daily precipitation renowned for simplicity (x_{t5}), total daily precipitation renowned for simplicity (x_{t6}) and finally total daily solar brightness renowned for simplicity (x_{t7}). Likewise, all lags included in the initial model and all categorical variables included in the initial model.

$$y_t = \beta_0 + \sum_{j=1}^{7} \beta_j y_{t-j} + \sum_{j=2}^{4} \beta_{6+j} y_{t-7j} + \sum_{j=1}^{7} \alpha_j x_{tj} + \sum_{j=0}^{2} \delta_j \mathrm{D}_{\mathrm{tj}} + \varepsilon_t, \quad t = 1, \ldots, T. \tag{3-2}$$

In the Figure **3-15** shows the standardized residuals and the fitted values given in the multiple linear model performed and referenced in equation (3-2). It is inferred that most of the residuals (95.21 %) are concentrated around the baseline (i.e. the line where the residuals are zero), indicating that the model may be making good predictions. However, the positive and negative residuals are not as evenly distributed on both sides, suggesting that the model may have a systematic bias in its predictions. Also, the distribution of the residuals might indicate that the variance of the residuals changes with the fitted values. This might suggest that the model is not correctly capturing the variability in the data.

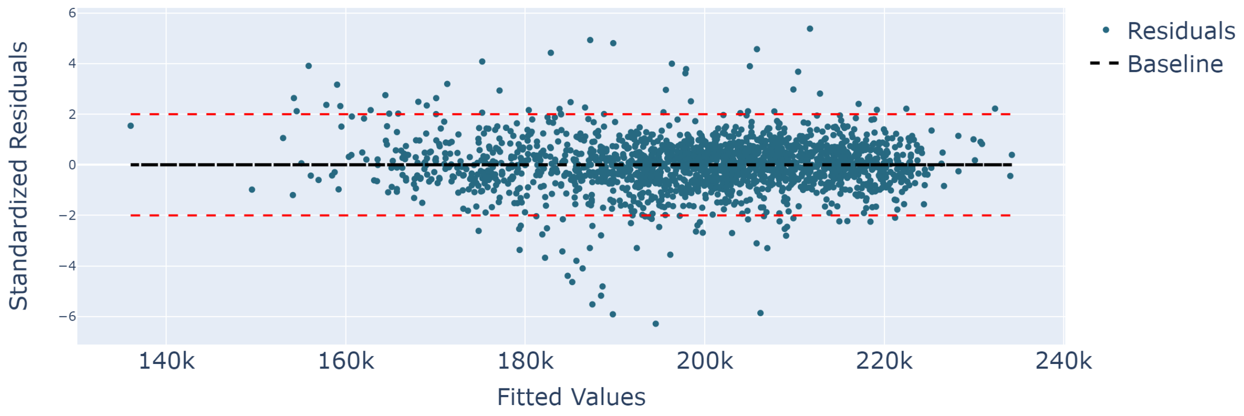


Figure 3-15.: Standardized residuals vs fitted values for the multiple linear model constructed.

Figure **3-16** compares the observed values with the values predicted by the model, it is inferred that since the observed values are very close to the perfect prediction line, it could be that the model is making accurate predictions. However, there is a considerable dispersion of points around the line, and this may indicate that the model is having difficulty accurately predicting the energy demand. Figure **3-17** shows a temporal pattern of constant mean residuals over time, which may indicate that the model could capture some of the variability in energy demand. Importantly, at the time when the pandemic began, the errors were larger, which supports the use of the covid indicator variable.

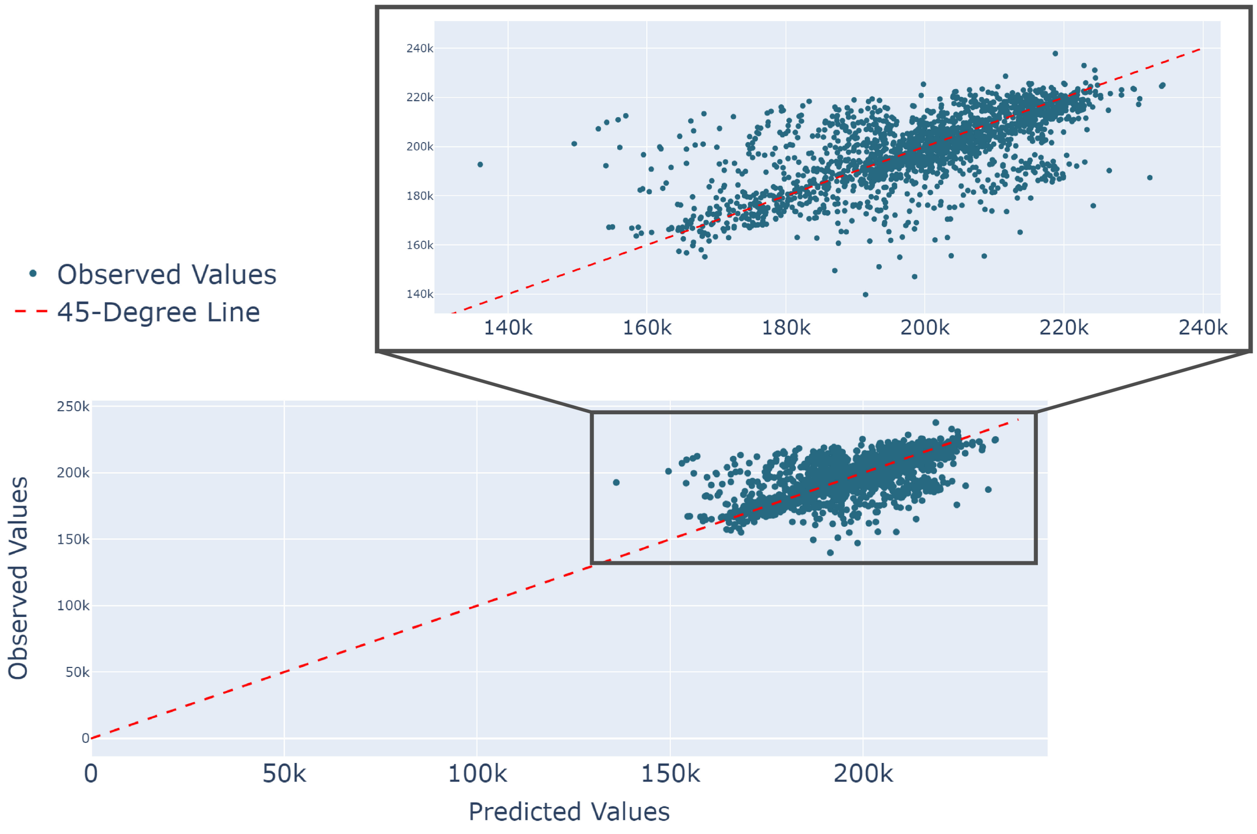


Figure 3-16.: Observed vs fitted values for the multiple linear model constructed.

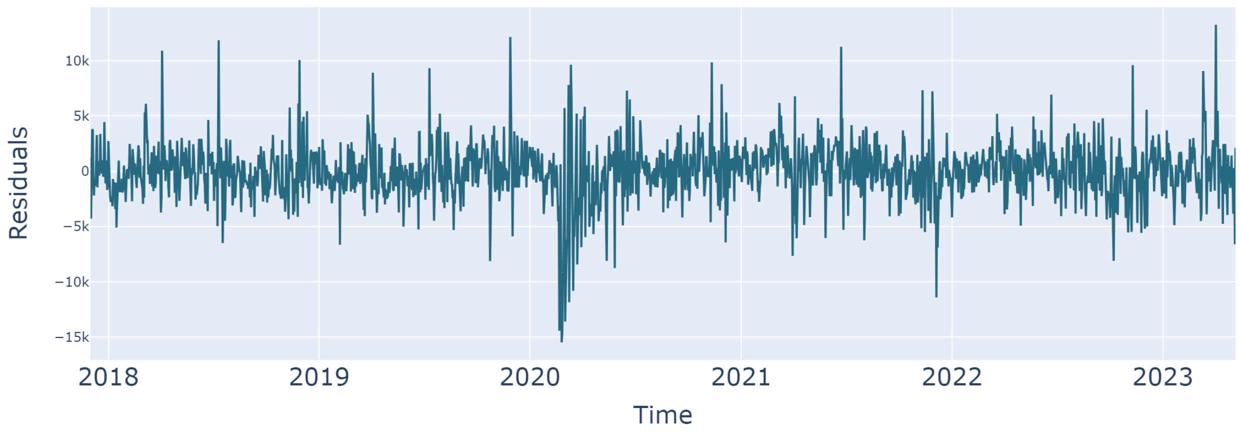


Figure 3-17.: Temporal pattern of constant mean residuals over time for the multiple linear model constructed.

Finally, Figure **3-18** graphs the correlogram of the residuals for time dependence, showing that some bars are not within the confidence bands, which means that there is sufficient evidence to reject the null hypothesis that there is no autocorrelation in these lags. Therefore, the model not seems to adequately capture the autocorrelation structure in the data.

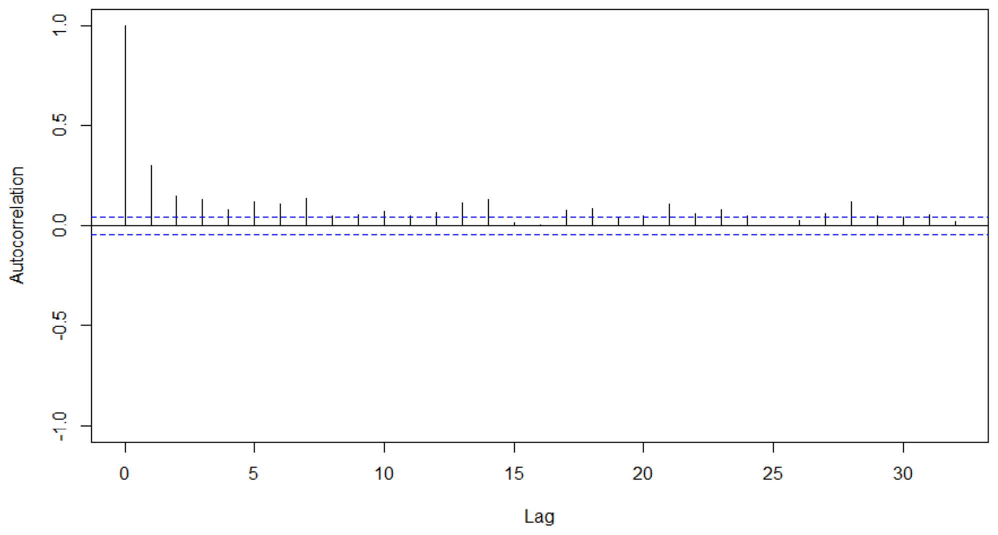


Figure 3-18.: Correlogram of the residuals for time dependence in the multiple linear model constructed.

In summary, in order to select the variables that have an effect on the variation of demand and at the same time demonstrate their respective significance, a multiple linear model was adjusted, where the Adjusted coefficient of determination (Adj. R-squared) was 0.976, which means that approximately 97.6 % of the variability in the dependent variable (Demand SIN) can be explained by the independent variables included in the model. However, there are indications that this model is no longer suitable for capturing demand variability and is therefore not the best for forecasting. For this reason, it is convenient to use other models that do not necessarily have a linear relationship between the dependent variable and their covariates.

4. Trained models

According to Zhang et al. (2023), modeling sequential data involves using ordered lists of feature vectors indexed by time steps, rather than individual feature vectors. When dealing with data that naturally occurs in sequences, the assumption of independence for individual inputs is replaced by acknowledging temporal dependencies within the sequences. Mathematically, this transition is expressed through specific formulations, like $\mathbf{y}_t \in R^d$, representing feature vectors at each time step t, and $P(\mathbf{y}_t|\mathbf{y}_{t-1}, \ldots, \mathbf{y}_1)$ capturing the temporal dependence. Moreover, the concept of unsupervised density modeling, particularly in sequence modeling $P(\mathbf{y}_1, \ldots, \mathbf{y}_T)$, plays a crucial role in estimating the likelihood of observing a sequence within a collection.

4.1. Machine Learning Models

In order to apply machine learning models to forecasting problems, we must transform the time series into a matrix where values are associated with lags or prior time windows. According to Amat Rodrigo and Escobar Ortiz (2023), a lag associated with a time step is defined as the value at a previous time step t. The matrix $\mathbf{L}$ is structured as a lag matrix representing a time series with m lags. Suppose that we have a series with T periods or observations. Each element y_{t-i} in the matrix corresponds to the value of the time series variable at time t with a lag of i periods. And each column represents a lagged observation of the time series variable. The resulting matrix would be the following, whose dimension is $T - m + 1$ rows and m columns.

$$\mathbf{L} = \begin{bmatrix} y_m & y_{m-1} & \cdots & y_1 \\ y_{m+1} & y_m & \cdots & y_2 \\ \vdots & \vdots & \ddots & \vdots \\ y_T & y_{T-1} & \cdots & y_{T-m+1} \end{bmatrix}. \tag{4-1}$$

This transformation is crucial for enabling machine learning models to discern the dependencies and patterns inherent between past and future values within a time serie. Using lags as input features, these models can assimilate information from historical data to make predictions about future values. The selection of the optimal number of lags as input features in the matrix is a pivotal hyperparameter that requires meticulous tuning to achieve the optimal performance of the model. This procedure is the same as the traditional hyperparameter search, which is based on generating combinations of lags with hyperparameters and finding the model with the best result or the best combination, this procedure will be explained in more detail in the section 4.2.

Let us illustrate the above with an example. Consider the construction of a lag matrix with three lags. Suppose we have a time series where $y_1 = 5$, $y_2 = 10$, $y_3 = 15$, $y_4 = 20$, $y_5 = 25$, $y_6 = 30$, $y_7 = 35$. Then, $m = 3$ and $T = 7$. The resulting matrix would be the following.

$$\mathbf{L} = \begin{bmatrix} 15 & 10 & 5 \\ 20 & 15 & 10 \\ 25 & 20 & 15 \\ 30 & 25 & 20 \\ 35 & 30 & 25 \end{bmatrix}. \tag{4-2}$$

The idea of constructing a lag matrix is to include it in the covariate matrix, where x_{ij} is the i-th value of the j-th variable for $i = 1 \dots T$ and $j = 1 \dots p$. The number of lags to be included in the matrix is determined in the hyperparameter search to be done in the training process. But in the selection of variables it is suggested to iterate over the seasonal lags of the periodicity of the series, in order to infer significance in the variation of the dependent variable. Likewise, the covariates are included in the matrix, in this case, the variables to be included in this matrix would be those set out in section 3.3 below. It is also important to note that in this project only the lags of the dependent variable are taken into account. The X matrix representing the covariate matrix is as follows.

$$\mathbf{X} = \begin{bmatrix} y_m & y_{m-1} & \cdots & y_1 & x_{11} & x_{12} & \cdots x_{1p} \\ y_{m+1} & y_m & \cdots & y_2 & x_{21} & x_{22} & \cdots x_{2p} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ y_T & y_{T-1} & \cdots & y_{T-m+1} & x_{T1} & x_{T2} & \cdots x_{Tp} \end{bmatrix}. \tag{4-3}$$

4.1.1. Linear Models

Let's proceed from the conception of linear models to generalize the methods most commonly used in these types of demand problems. Friedman et al. (2010) used the linear model regression (4-4), which estimators are represented as $\widehat{\beta}^{ols}$.

$$y_t = \beta_0 + \sum_{j=1}^{p} \beta_j x_{tj} + \varepsilon_t, \quad t = 1, \dots, T. \tag{4-4}$$

In equation (4-4), a time series regression model is presented to examine the relationship between the response variable y_t and a set of covariates x_{tj} over different time periods t. Here, t represents various time points considered. The term β_0 is the model's constant, β_j

are the coefficients associated with the covariates x_{tj}, and ε_t is the error term capturing any variability not explained by the model. This approach enables an understanding of how covariates influence the target variable over time, providing valuable insights for trend and pattern analysis in the time series. The objective then is to choose an estimator that fits the data given the minimum sum of squared errors.

$$\widehat{\beta}^{ols} = \min_{\beta} \left\{ \frac{1}{T} \sum_{t=1}^{T} \left(y_t - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{tj} \right) \right)^2 \right\}. \tag{4-5}$$

This linear model can be generalized in the first instance with a penalty to the coefficients, i.e., restricting them in size. In ridge regression the penalty is l_2 which is defined as $l_2 - norm, \|\beta\|_2^2 = \sum_{j=1}^{p} \beta_j^2$, in this procedure, there is a parameter called λ which controls the amount of shrinkage, the higher the value of λ, the larger the quantity of shrinkage, the ridge estimate is represented by the following equation (4-6).

$$\widehat{\beta}_{\lambda}^{ridge} = \min_{\beta} \left\{ \sum_{t=1}^{T} \left(y_t - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{tj} \right) \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}. \tag{4-6}$$

The other methodology is least absolute shrinkage and selection operator (lasso), proposed by Friedman et al. (2010). It is similar to ridge regression, but with some differences. Lasso uses an l_1 penalty defined by $\sum_{j=1}^{p} |\beta_j|$ and this results in a sparse statistical model, it means that there is a small number of predictors that have a significant role. And if the number of variables is greater than the number of observations, then there will be infinite solutions that overfit the data.

$$\widehat{\beta}_{\lambda}^{lasso} = \min_{\beta} \left\{ \sum_{t=1}^{T} \left(y_t - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{tj} \right) \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}. \tag{4-7}$$

The fourth methodology is elastic net proposed by Zou and Hastie (2005) adopt a penalization that is a combination of the ridge and lasso penalties and its estimator is represented as $\widehat{\beta}_{\lambda,\alpha}^{en}$. Following equations represent the parameters estimation with each methodology:

$$\widehat{\beta}_{\lambda,\alpha}^{en} = \min_{\beta} \left\{ \frac{1}{2T} \sum_{t=1}^{T} \left(y_t - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{tj} \right) \right)^2 + \lambda \left[\sum_{j=1}^{p} (1-\alpha) \frac{1}{2} \beta_j^2 + \alpha |\beta_j| \right] \right\}. \tag{4-8}$$

They employ distinct penalties that result in particular shrinkage patterns. In ridge regression, the quadratic penalty shrinks the parameters toward the origin. In lasso, the penalty on the sum of absolute values has the effect of reducing the least relevant parameters to zero, thus imposing a certain level of dispersion. In the elastic net an intermediate effect is achieved. Both λ and α play the role of hyperparameters: λ controls the strength of the parameter shrinkage, while α tunes the balance between penalties on the l_1 and the l_2 norm of β. Another notion to emphasize is that three methods share the same standard quadratic loss.

4.1.2. Ensemble Methods

An Ensemble Model is a Machine Learning strategy in which predictions from different models join forces to improve accuracy and performance. Rather than relying on just one model, Ensemble Methods combine several to deal with data complexity. There are two main approaches: Bagging, which trains several instances of the same model on different sets of data and combines their predictions for a more reliable outcome, and Boosting, where models are built one after the other, with each new model focusing on fixing mistakes from the previous ones. Ensemble Models are widely used because they make our predictions better, especially when dealing with new and unseen data (Mohammed and Kora, 2023).

According to Breiman (2001), Random Forests constitute an ensemble of predictor trees, where each tree's dependence relies on values from an independently and identically distributed random vector. This vector is sampled independently for each tree within the forest. In simpler terms, a Random Forest is a collection of decision trees that work together to enhance model accuracy and stability. The ideas presented in the paper are also applicable to regression problems. In fact, the paper mentions that the adaptive bagging algorithm in regression was designed to reduce bias and operates effectively in both classification and regression. Random forests have also been shown to give competitive results in regression problems.

According to Friedman et al. (2000), Boosting is one of the most important recent developments in classification methodology that works by sequentially applying a classification algorithm to reweighted versions of the training data and then taking a weighted majority vote of the sequence of classifiers thus produced. Boosting can be viewed as an approximation to additive modeling on the logistic scale using Bernoulli maximum likelihood as a criterion. The idea is to fit a sequence of weak models to the data, where each model is trained on a modified version of the data that emphasizes the examples that were misclassified by the previous models. In the context of boosting, a weak model refers to a simple and weak predictive model that is used as a building block in the construction of a more complex and accurate model through the boosting technique. The final prediction is then obtained by combining the predictions of all the weak models using a weighted majority vote.

In his paper Friedman (2001) describes how Boosting in regression works by putting together simpler regression models to create a more powerful additive model. In each iteration, a weak model is fitted to the residual of the previous model and added to the current model. This process repeats until a predetermined number of iterations is reached or a desired accuracy is achieved. The goal is to enhance the accuracy of the final model by combining multiple simpler models. The Boosting algorithm can be applied to various fitting criteria, such as mean squared error, mean absolute deviation, and Huber loss. The Gradient Boosting Machine algorithm relies on gradient descent, a technique used to minimize a loss function. The specific loss function that is minimized by the Gradient Boosting Machine algorithm depends on the problem being solved.

Based on Breiman (2001) Random Forests and Gradient Boosting, two prominent ensemble learning techniques, diverge in their approach to constructing decision trees. Random Forests assemble multiple independent decision trees, mitigating overfitting by averaging their predictions. In contrast, Gradient Boosting incrementally builds a single decision tree in each iteration, incorporating it into the evolving model. While Random Forests randomly sample features at each tree node, Gradient Boosting employs all features in every iteration. Moreover, Random Forests derive a final prediction by averaging the predictions of all trees, while Gradient Boosting uses a weighted sum. Notably, Random Forests demonstrate a lower susceptibility to overfitting due to the independent construction of trees and the subsequent averaging of predictions. Conversely, Gradient Boosting may be prone to overfitting when constructing too many trees or utilizing overly complex ones.

A new version of Boosting is explained in Nielsen (2016), XGBoost, it fits a new decision tree to the training dataset. After fitting the tree, XGBoost adds it to the existing ensemble of trees and adjusts the weights of the existing trees to better fit the training data. It uses a boosting technique called “Newton boosting”, which involves a way of adjusting the weights of existing trees during each boosting step. This approach allows to fine-tune tree weights more precisely and efficiently compared to traditional gradient boosting methods. To prevent overfitting, it applies L1 and L2 regularization techniques shown in subsection 4.1.1, to penalize tree weights. It also gives the option to penalize individual trees, affecting both, the structure of the trees and the weights of their branches to make the predictions more reliable. Additionally, it introduces a randomization feature that can make the individual trees less correlated, potentially reducing the variability of the model. Further mathematical details of the XGBoost method can be found in appendix A.3.

4.1.3. Deep Learning Models

So far, we have offered a comprehensive exploration of machine learning, we will now discuss deep learning, one of the specialized subsets within this field. The focus here is on NNs with multiple layers. The conceptual line proposed so far has been to start building regularized regression models, followed by ensemble models such as random forest and gradient boosting and finally more complex DL models. The aim is to find the model that best captures the generality of electricity demand in Colombia. More details of the ensemble models and the deep learning models built can be found in appendix A.4.

4.2. Model selection

Some methods are available to choose the model that effectively captures trend, seasonality, regular patterns, and even potential uncertainties. Metrics for predictive performance such as mean squared error, mean absolute error, and mean absolute percentage error are commonly employed for this purpose. However, it is imperative not only to assess how well the model performs in fitting a specific dataset but also to evaluate its predictive capacity, ensuring it can generalize learning and exhibit robust predictive performance on evaluation set.

For all models tested, the following procedure was carried out: The dataset is split into three periods for training, validation, and testing. The training set spans from december 1, 2017, to may 31, 2023, with a total of 2008 data points, which corresponds to 94.27 % of the data set. The validation set covers june 1, 2023, to september 23, 2023, containing 115 data points, which corresponds to 5.40 % of the data set. Finally, the testing set comprises the dates from september 24, 2023, to september 30, 2023, with a total of seven data points, which corresponds to 0.33 % of the data set. This is because the size of the test set must be equal to the seasonal periodicity and therefore periodicity in which the prediction or backtesting process is being done.

4.2.1. Backtesting

According to Amat Rodrigo and Escobar Ortiz (2023), backtesting involves retrospectively assessing the performance of a predictive model by applying it to historical data. Essentially, it represents a specialized form of cross-validation specifically applied to preceding time periods. The main objective of backtesting is to rigorously assess the precision and efficacy of a model while identifying potential issues or areas for improvement. Model performance can be evaluated on previously unseen data by analyzing historical data. This constitutes a critical phase in the modeling process, playing a key role in ensuring the model's reliability. Based on Brownlee (2017), backtesting can be executed through diverse techniques, ranging from straightforward train-test splits to more sophisticated approaches like rolling windows or expanding windows.

In Amat Rodrigo and Escobar Ortiz (2023), the model is trained on each iteration, increasing the training set while preserving the data temporal order. The model is trained before making predictions each time, using all available data up to that point in the training process. This approach contrasts with conventional cross-validation, where data is randomly partitioned into training and validation sets. And this sequential approach allows the model to be tested on progressively larger amounts of historical data, facilitating a more accurate assessment of its predictive capabilities.

Performing backtesting with refit includes the following steps: Start by training the model with an initial training set. Once the model is trained, use it to predict the next seven steps in the data, keeping these predictions for later evaluation. Expand the training set, retrain the model with the updated data, and use it again to predict the next seven steps. Repeat the last two procedures until the entire series has been tested. The above procedure is done in a sequential and orderly manner seven by seven until all the training data is completed. This refit is every seven days, due to the seasonal periodic is weekly. This approach ensures that the model is evaluated on multiple sets of test data, leading to a more accurate evaluation of its predictive capabilities, (Amat Rodrigo and Escobar Ortiz, 2023).

In Figure **4-1** illustrates the process described above, called "Backtesting with refit and

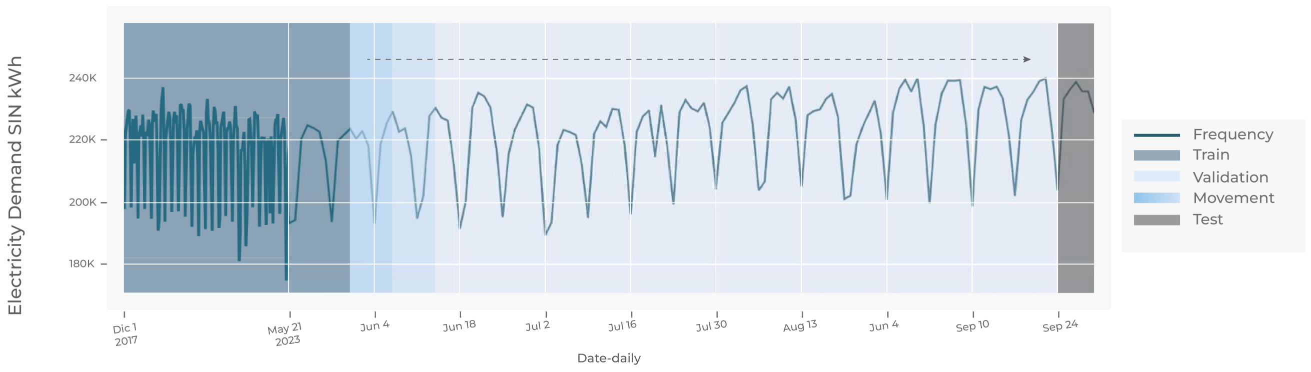


Figure 4-1.: Backtesting with refit and increasing training size (fixed origin).

increasing training size with a fixed origin", where the training part goes from december 1, 2017 to may 31, 2023, where the validation data and the backtesting process begins. This process ends in september 23, 2023 where finally starts the last seven days which is the test set. The purpose of this strategy was to ensure a precise assessment of the model and build confidence in its ability to predict new data accurately, comparing various retraining frequencies and choose the one where the error metric consistently demonstrates an improvement. With the help of backtesting, this pattern can be accurately simulated. This procedure can be found in more detail in A.2.

4.2.2. Model results

This section will present the results of the models developed for the point forecasting of electricity demand in Colombia. The variables finally used to test all models are: day type, COVID pandemic period indicator variable, level change indicator variable, total daily precipitation, maximum daily wet temperature, mean daily wind speed and daily persistence. This was also supported by testing some combinations of weather variables in the models and evidencing the forecasting performance in models. And the first twenty eight lags of electricity demand. All the development was done in Python and the code can be found in the repository exposed in the Appendix A.6.

Regression Models

The type of models used to predict daily electricity demand was divided into three categories; regression models starting with the Rigde model, the Lasso model and the ElasticNet model. Followed by ensemble models such as Random Forest and XGBoost. And finally Support Vector Machine models such as SVM and linear SVM. In this sub-section, the results of those models will be presented. All models include the same set of variables selected and discussed in section 3. It is also important to note that all models underwent the same pre-processing which involved standardizing features by removing the mean and scaling variance and coding the categorical variables.

The Figure **4-2** shows the predictions of all the models tested. In Figure **4-3** shows the forecast for the last seven days for the best models in each of the three subcategories of Machine Learning type models, among the regularized linear models are Ridge and Lasso with the best result. Among the ensemble models, the model with the best predictive performance was XGBoost. And finally for SVM, the best performing model was the Linear SVM Model. Including the forecast made by XM [1], which is the regulatory entity for energy in Colombia.

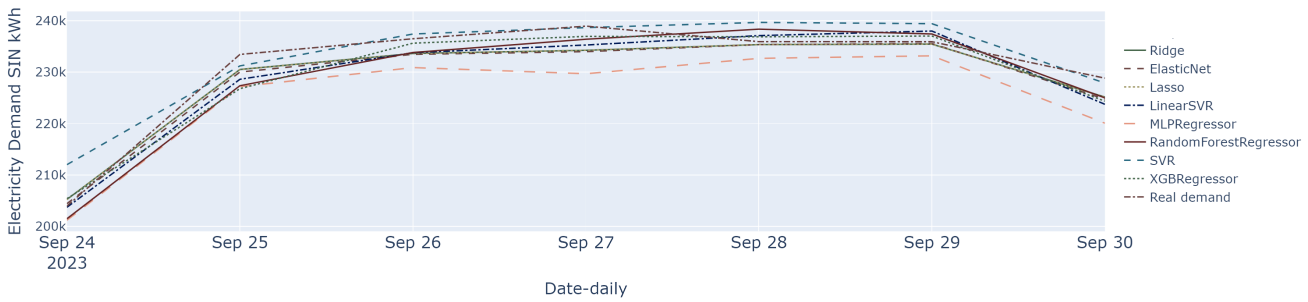


Figure 4-2.: Point forecasts for the last seven days of the data set, for all models tested, given by MAPE.

[1] See: `https://www.xm.com.co/consumo/informes-demanda/indicadores-de-pronosticos-oficiales-de-demanda`

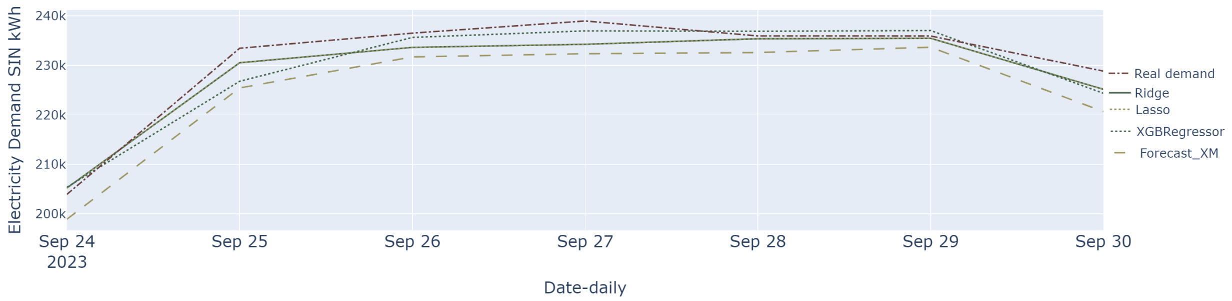


Figure 4-3.: Point forecasts for the last seven days of the data set, whose predictive performance was good, given by MAPE.

A first inference from the results is that the forecasts of the selected models are close to the actual value. In the Table **4-1** shows the optimal parameters for the selected models without including the XM forecast because they are the owners of the parametric information of the models they built. These include the number of lags of the dependent variable, which is the electricity demand. The regularization parameter in each case, whether it includes intercept, maximum number of iterations, and the type of optimizer in the case of regularized models. The ensemble model iterated on four hyperparameters: the number of lags included, the maximum number of tree depths, which means, it is the maximum number of partitions for each tree in the ensemble model. Increasing this value will make the model more complex and more likely to overfit. And finally, the number of estimators, which is the number of gradient boosted trees. Equivalent to number of boosting rounds. These parameters are those for which each model generated the best predictive performance.

In Table **4-2** shows the predictive performance metrics in each model, such as mean squared error, mean absolute error, and mean absolute percentage error. Due to the MAPE expresses errors as a percentage of the actual value, the selection of the best model was based on this metric, because it is easy to interpret. A lower MAPE indicates better model performance, as it means the predictions deviate less in percentage terms from the actual values. It can be seen that the models are organized from the one with the lowest MAPE to the model with the highest MAPE. Both MAE and RMSE confirm that the Ridge and Lasso models are the most accurate in this dataset, as they show the lowest values in both metrics. These metrics indicate that these models have a better fit and predictive ability compared to the other models evaluated. Details of the in-sample predictive performance metrics can be found in Appendix A.5.

Model	Optimal Parameters
Ridge	Energy demand lags: [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21] λ: 0.0378961698064472236 Fit Intercept: True Solver : auto Tol: 0.0001
Lasso	Energy demand lags: [1 2 3 4 5 6 7 8 9 10 11 12 13 14] λ: 1.5998587196060572e-05 Fit Intercept: True Max iter: 1000 Selection : cyclic Tol: 0.0001
XGBRegressor	Energy demand lags: [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21] Objective: reg:squarederror Max depth: 5 N estimators: 30

Table 4-1.: Optimal Parameters for the Best Trained Models.

Model	MAPE (%)	MAE (KWh)	RMSE (KWh)
Ridge	1.016	2355.948	2793.564
Lasso	1.017	2359.716	2798.997
ElasticNet	1.034	2416.559	2999.887
XGBRegressor	1.084	2499.921	3239.548
RandomForestRegressor	1.337	3079.088	3374.579
LinearSVR	1.212	2831.653	3295.441
SVR	1.273	2814.902	3743.540
MLPRegressor	2.379	5521.892	6082.823

Table 4-2.: Out-of-sample predictive performance metrics statistical and ML models.

Finally, the regression model chosen is Lasso, since it is one of the two models with the best predictive performance and it was also chosen for its parsimony in the parameters and variables, due to the type of regularisation it handles. The theoretical model is presented in equation (4-9), where y_t represents the energy demand in Colombia at time t, y_{t-j} represents the lags of the energy demand which includes the fourteen optimal lags ($m = 14$), x_{tj} represents each weather covariate j at time t, which are total daily precipitation, maximum daily wet temperature, mean daily wind speed and daily persistence and finally the Covid indicator variable, the level change and the day type are represented as D_{tj}. These covariates were discussed in section 3, the covariate matrix has the structure shown in matrix 4-3, and following is the matrix for the selected model:

$$y_t = \beta_0 + \sum_{j=1}^{14} \beta_j y_{t-j} + \sum_{j=1}^{4} \alpha_j x_{tj} + \sum_{j=0}^{2} \delta_j \mathrm{D_{tj}} + \varepsilon_t, \quad t = 1, \ldots, T. \tag{4-9}$$

$$\mathbf{X}^* = \begin{bmatrix} y_{14} & y_{13} & \cdots & y_1 & x_{11} & x_{12} & \cdots x_{1p} \\ y_{15} & y_{14} & \cdots & y_2 & x_{21} & x_{22} & \cdots x_{2p} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ y_T & y_{T-1} & \cdots & y_{T-15} & x_{T1} & x_{T2} & \cdots x_{Tp} \end{bmatrix}. \tag{4-10}$$

The Figures **4-4**, **4-5** and **4-6** gives an overview of how the different models compare in terms of their ability to predict observed values. And also, provides information on the goodness of fit of the models by looking at the distribution and patterns of forecast error over time. In general it is evident that the spread of the points around the line (where the forecast error is 0) for the prediction of some days is uniform suggesting that some models are close to adequately capturing most of the variability in the data. It is important to note that the regularized models, forecast errors are closer to zero than the other models. It can be inferred that the regularized models are closer to the real points.

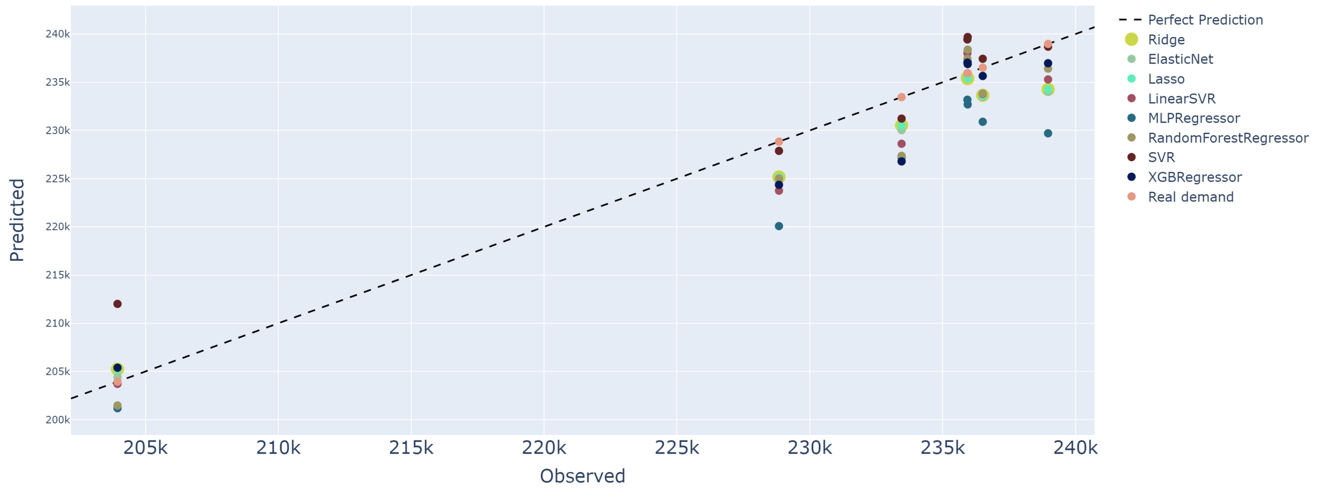


Figure 4-4.: Overview of how the different models compare in terms of their ability to predict observed values.

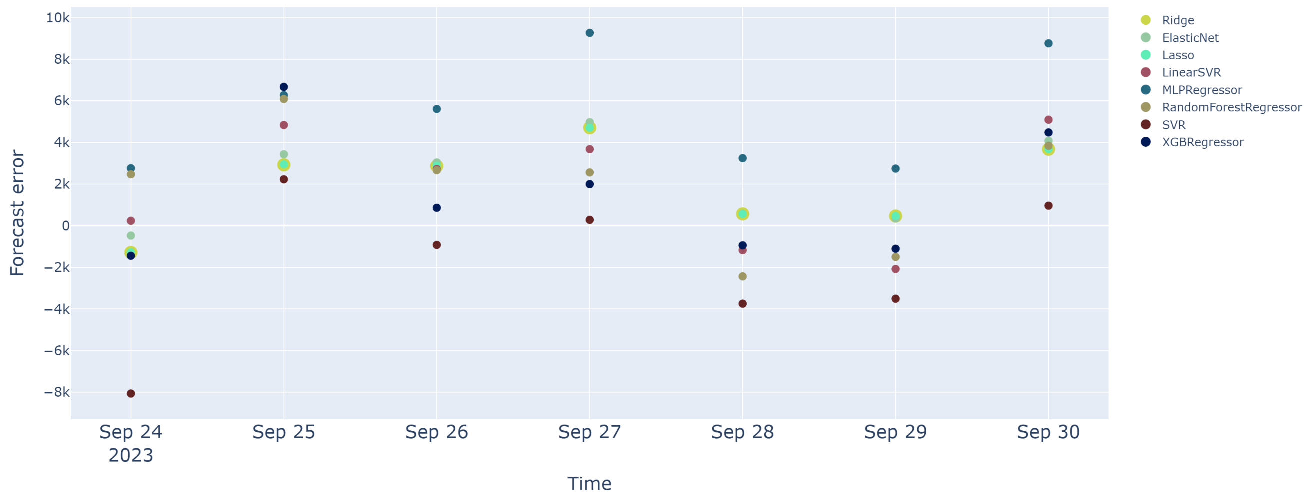


Figure 4-5.: Forecast errors for all trained models.

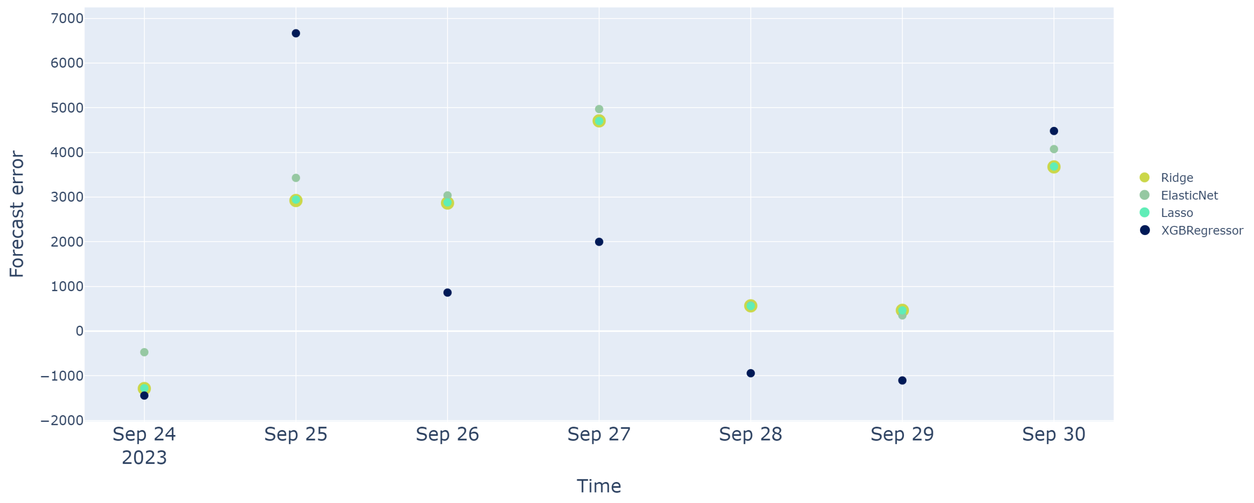


Figure 4-6.: Forecast error for Lasso, Ridge, ElasticNet and Extreme Gradient Boosting models.

Figure **4-7** provides information on the tendency of each model to over- or underestimate predictions and helps to assess the relative quality of the models in terms of bias. It is calculated as the difference between the forecast and the average actual value in each model. From which, it can be inferred that most models underestimate the values to be predicted. However, in terms of comparability, it is confirmed that the regularized regression models and Extreme Gradient Boosting could underestimate to a lesser extent than the other models. A figure that goes into more detail on the bias for each model can be found in Appendix A.5.

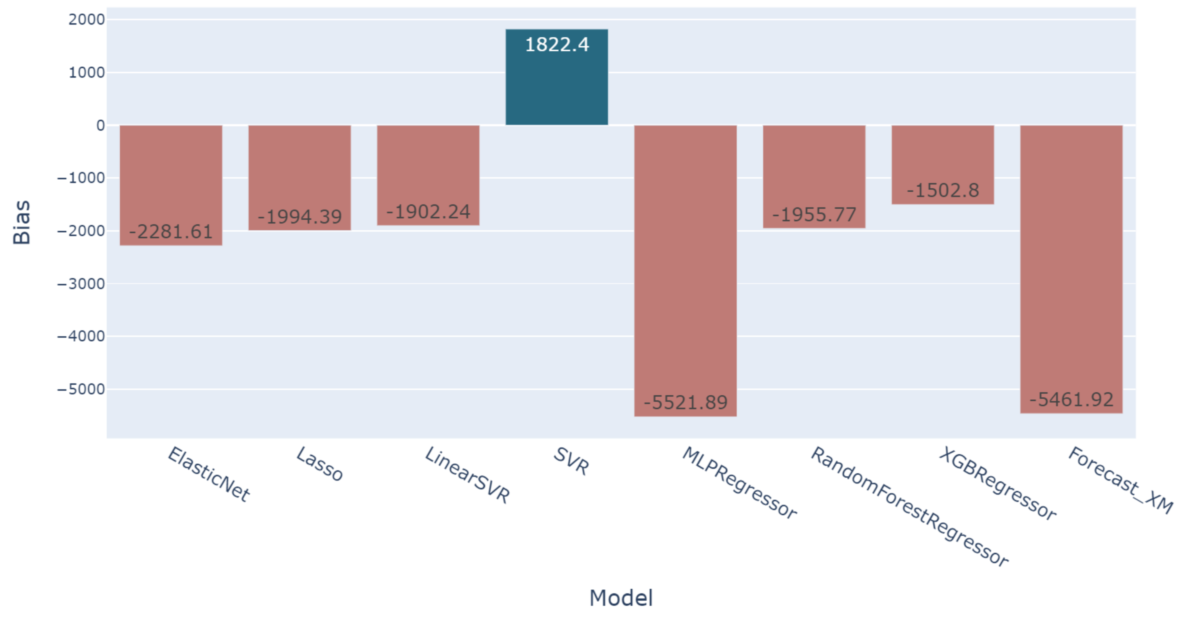


Figure 4-7.: Bias for all trained models.

Figure **4-8** allows us to visualise how the Lasso model predictions compare with the actual values and how these differences vary across the observed values. There is a pattern of concentration in the differences below the baseline, indicating that the prediction was lower than the actual value at most points. It can therefore be inferred that for these values, the Lasso model tends to underestimate the values.

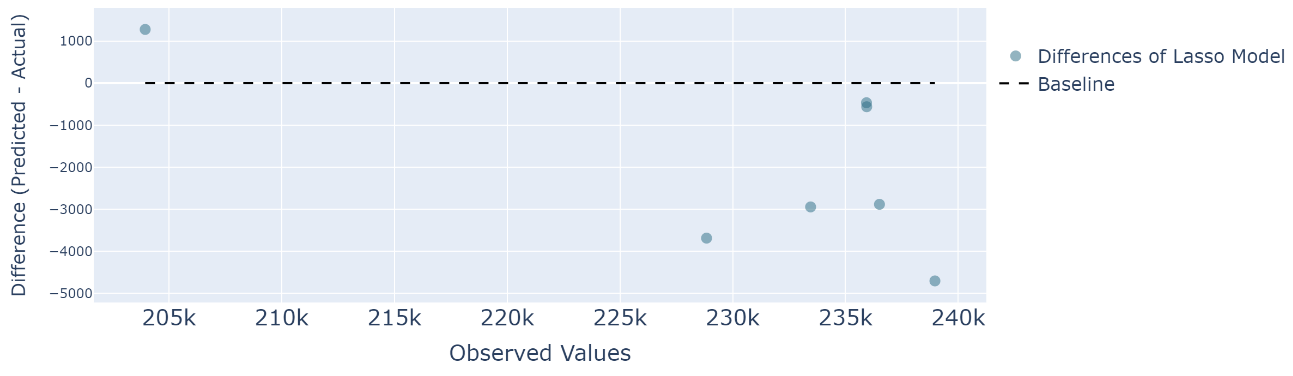


Figure 4-8.: Lasso model predictions compare with the actual values.

Figure **4-9** shows the observed values with respect to the fitted values within the sample, which shows that most of fitted values are dispersed around the line of perfect fit. This is an indication that the model may be capturing most of the variability in energy demand.

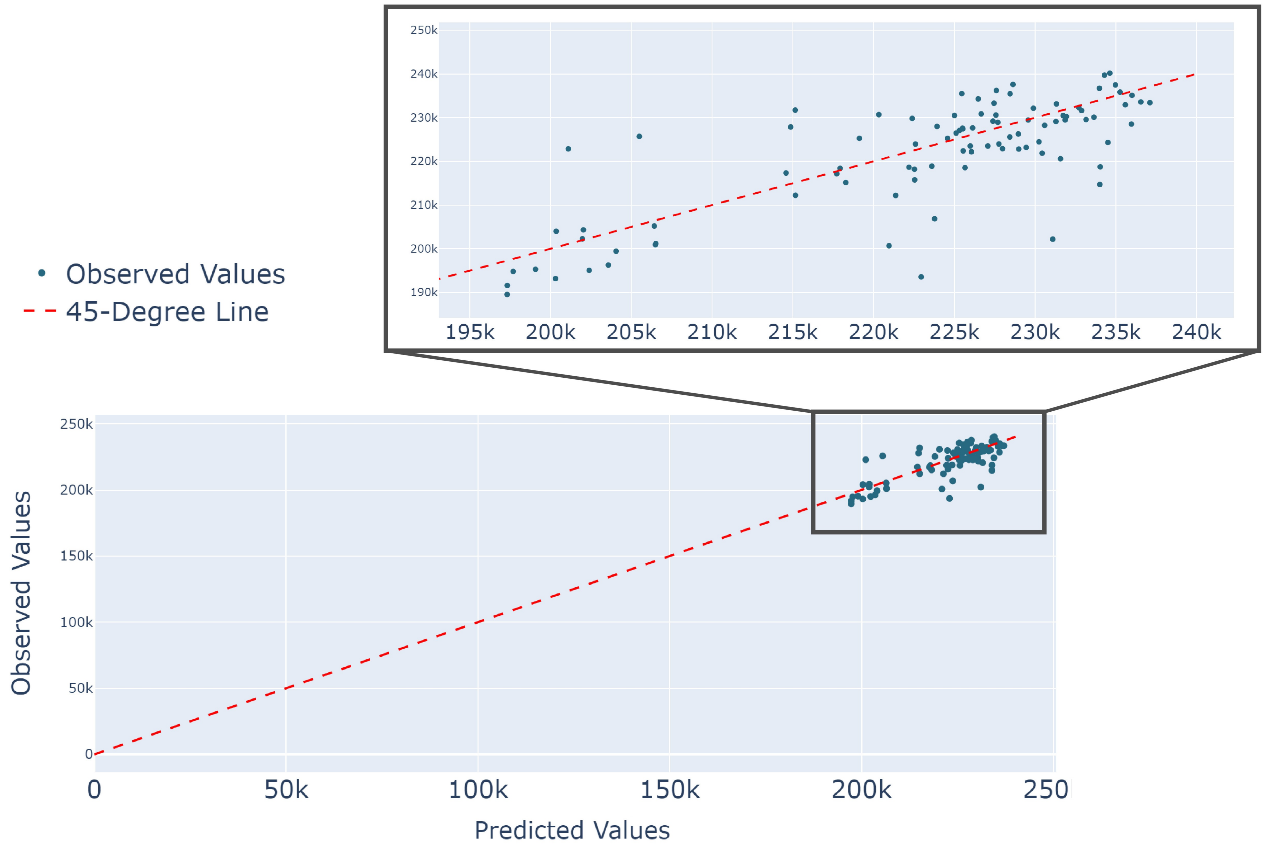


Figure 4-9.: Observed vs. fitted values for the Lasso model for validation data.

Figure **4-10** shows the pattern in the residuals as a function of time, in which, it is observed that in general, the residuals are randomly distributed around zero.

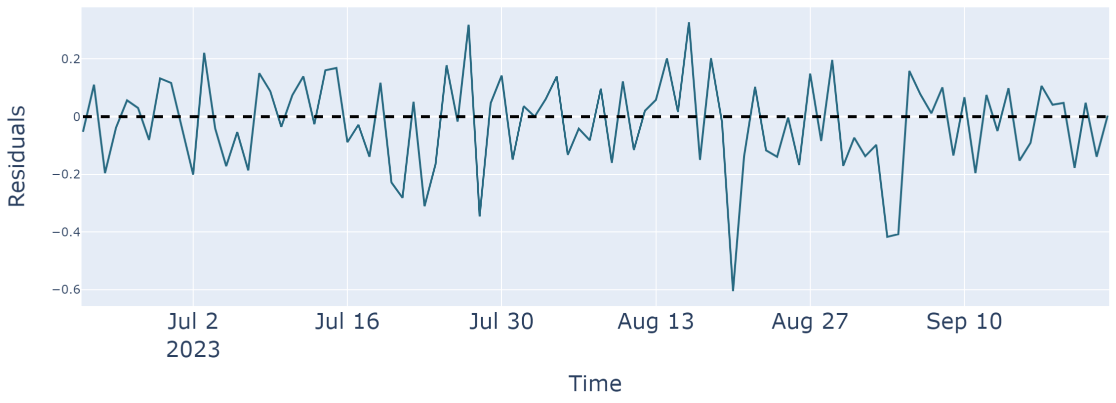


Figure 4-10.: Residuals over time for the Lasso model (In-sample).

Deep Learning Models

Following the search for a model that could better generalize the behavior of the daily energy demand, Machine Learning models were also built but focused on Deep Learning. These were simple RNN, LSTM, Gated Recurrent Units (GRU) and finally Bidirectional LSTM was estimated. But finally the model with the lowest MAPE is the one selected to expose and explain. In this sub-section the general results of the DL models and the detailed results of the chosen model will be presented.

In Figure **4-11** show the point forecast for all DL models trained, one inference that can be drawn is that the forecasting of such models, which are more complex and rely on capturing deeper patterns in the data, do not seem to be superior to the more classical ML models.

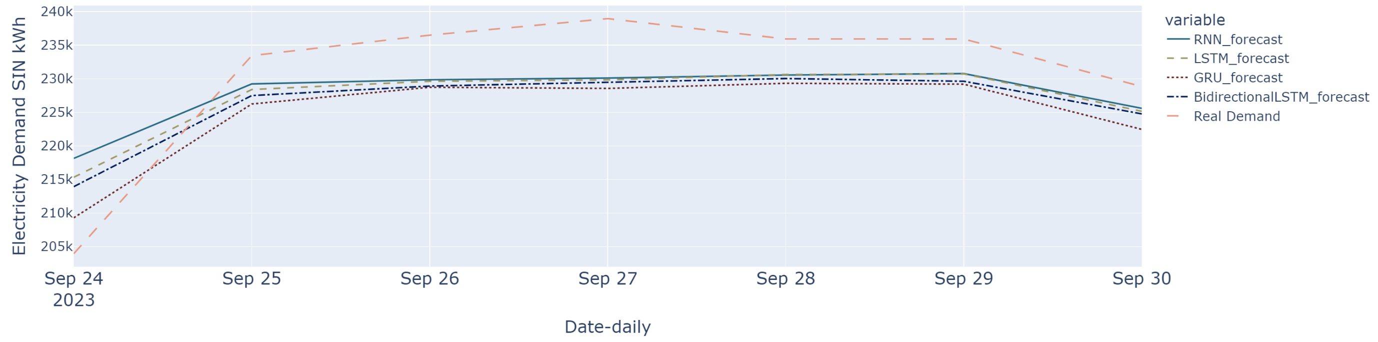


Figure 4-11.: Prediction of four additional deep learning models for the same test data set, the last seven days of database, and Real demand and XM Forecast.

In Table **4-3** show the optimal parameters for the selected DL model, which is LSTM, with a MAPE of 2.93 %. The parameters include the type of model, which is a sequential model with LSTM layers, dropout layers, and a final dense layer for regression tasks. The hyperparameters, such as the number of units in each LSTM layer, activation functions, dropout rates, and the number of layers. And the model is compiled with the Adam optimizer.

This parameters indicates that the model is defined sequentially, layer by layer. There is one LSTM layer with 96 long-term memory units. LSTM units are the basic units that make up an LSTM layer. Each unit has an internal structure that allows long-term information to be remembered. A dropout layer has been applied with a dropout rate of 96 % after the first LSTM layer. The dropout layer is used to regularise the model and avoid overfitting by randomly switching off a percentage of units during training. It also has another LSTM layer with 128 units, another two layers each with 32 units and finally there is another dropout layer with a dropout rate of 32 % after the third LSTM layer. In summary, this LSTM model has an architecture with three stacked LSTM layers, each followed by a dropout layer for regularisation. The units in the LSTM layers are 96, 128 and 32 respectively.

Model	**Optimal Parameters**
LSTM	Model: sequential
	lstm : 96
	dropout: 96
	lstm 1 : 128
	lstm 2 : 32
	lstm 3 : 32
	dropout : 32

Table 4-3.: Optimal parameters for the deep learning model with best MAPE.

And finally in the Table **4-4** shows the predictive performance metrics for each trained DL model. The model with the lowest MAPE among the DL-type models is selected. However, compared to the lasso model, which was chosen among the models built within the set of regression models, the LSTM model fails to improve the predictive performance of lasso.

Model	**MAPE (%)**	**MAE (KWh)**	**RMSE (KWh)**
RNN	3.02	6812.311	7634.647
LSTM	2.93	6658.487	7114.551
GRU	3.11	7203.061	7352.403
Bidirectional LSTM	3.08	7033.503	7300.022

Table 4-4.: Out-of-sample predictive performance metrics for different deep learning models.

The Figure **4-12** and **4-13** provide information on the goodness of fit of the models by looking at the distribution and patterns of forecast error over time. From which, it can be inferred that the deep learning models do not approach the real point as well as the previously analyzed models do. Finally, Figure **4-14** shows information on the tendency of each model to overestimate or underestimate predictions and helps to assess the relative quality of the models in terms of bias. In which, it can be inferred that there is a generalized underestimation in all deep learning type models.

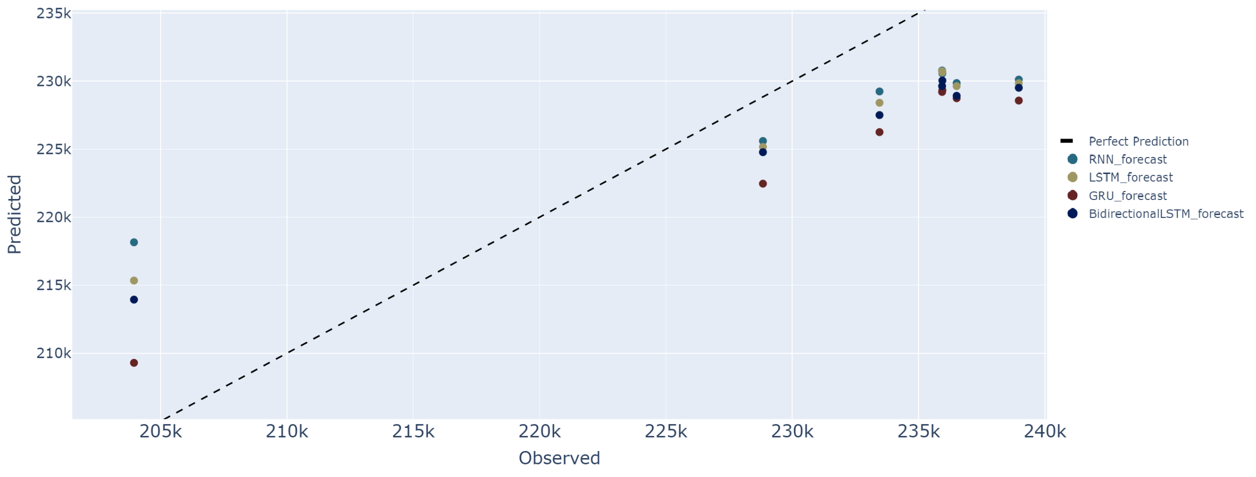


Figure 4-12.: Overview of how the DL models compare in terms of their ability to predict observed values.

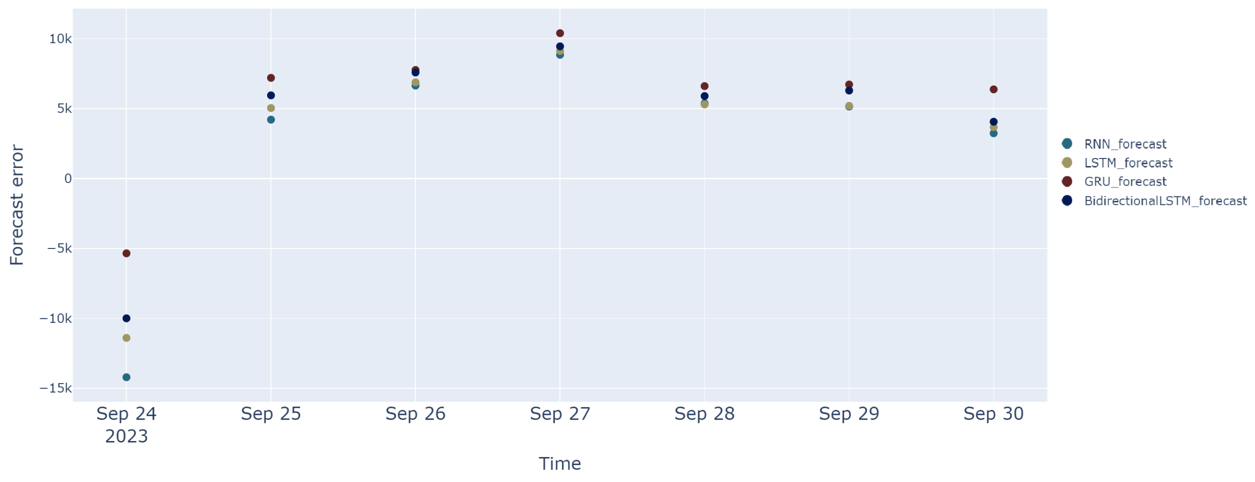


Figure 4-13.: Forecast-error for Deep Learning models.

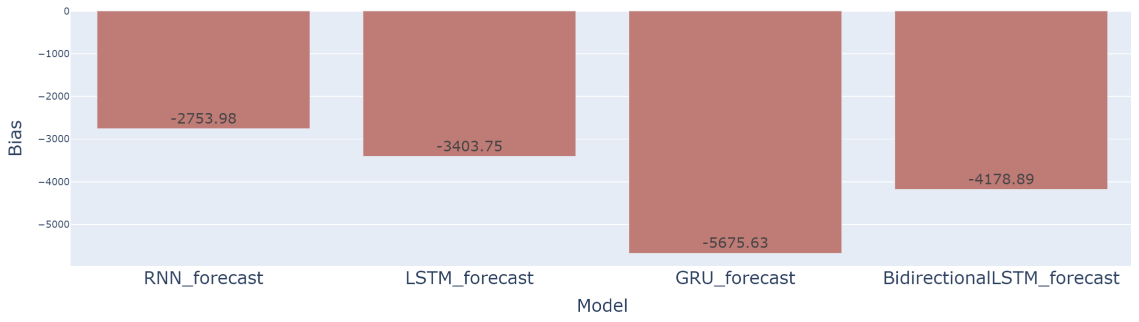


Figure 4-14.: Bias for Deep Learning models.

In conclusion, in this project, the model finally selected is Lasso, given its predictive capacity over the other models. Also because of the proportion of bias given in their forecasts compared to the other models. And finally, because its in-sample fit shows acceptable behavior. We considered detailing the fit of models whose predictive performance in terms of MAPE and out-of-sample was less than 2 %. For this reason, the prediction intervals that will be shown in chapter 5 will be calculated from the values predicted by this model. Details of the in-sample predictive performance metrics can be found in Appendix A.5.

5. Prediction intervals

We can define the prediction intervals (PIs) as in Jensen et al. (2024). Let $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ be the input and label random variables respectively. $f(x, y)$ the joint distribution of x and y, and $f(y|x)$ represents the conditional distribution of y given x. A PI was created using a sample collections $\{(x_t, y_t), t = 1, 2, ..., T\}$, it is given by $\widehat{\zeta}(x) = [L(x), U(x)]$, where L and U are functions. In other words, this approach uses these functions to define a range within which the predictions for the output variable are expected to fall for a given input value. The length of a PI is given by $(U(x_{T+m}) - L(x_{T+m}))$, where m represents the m forward predictions. This interval is conditioned by a confidence level, and this level is the probability that a new observation falls within the interval. This means that a more uncertain prediction interval produces broader intervals. A PI is valid, if the coverage probability for a new test point is guaranteed to be greater or equal to the designed confidence level.

5.1. Prediction intervals via bootstrap

According to Amat Rodrigo and Escobar Ortiz (2023), we can define bootstrapping method as an approach which creates numerous predictions. Each iteration involves selecting a sample from a group of previously observed prediction errors, resulting in a distinct set of predictions for every bootstrapping process. In other words, the error in a one-step-ahead forecast is the difference between the actual value and predicted value $\varepsilon_t = y_t - \widehat{y}_{t|t-1}$, known as forecast error. By assuming that future errors will be similar to past errors, it is possible to simulate different predictions by taking samples from the collection of errors previously seen in the past and adding them to the predictions.

According to Amat Rodrigo and Escobar Ortiz (2023), the iterative implementation of this procedure generates a set of predictions or bootstrapping iterations that are highly varied, reflecting the range of potential outcomes derived from the variability predicted in the forecasting process. Using the outcome of the bootstrapping process, prediction intervals can be computed by calculating the $\frac{\alpha}{2}$ and $1 - \frac{\alpha}{2}$ percentiles at each forecasting horizon. The procedure described above is shown in figures **5-1** and **5-2**.

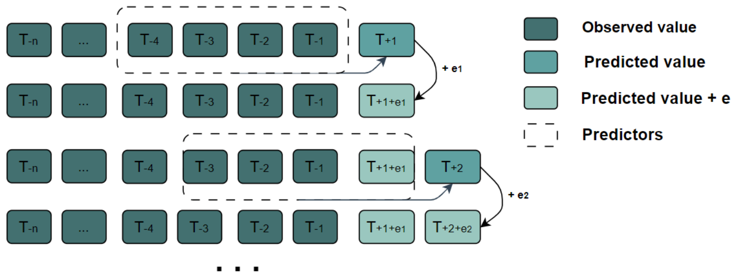


Figure 5-1.: Diagram bootstrapping prediction process. Figure is taken from Amat Rodrigo and Escobar Ortiz (2023).

Where T represents the sample size, and the sub-index represents the position of the lag value. T_{+1} represents the predicted value of the first value to be predicted, T_{+2} represents the predicted value of the second value to be predicted, and so on, the forecast error is added, and when finally the totality of values to be predicted is obtained, a bootstrap sample will be obtained. This process is repeated to create as many bootstrap samples as desired. The Figure **5-2** is the completed process, where each row represents the sample obtained from the same point several times, each time sampling is generated is one row.

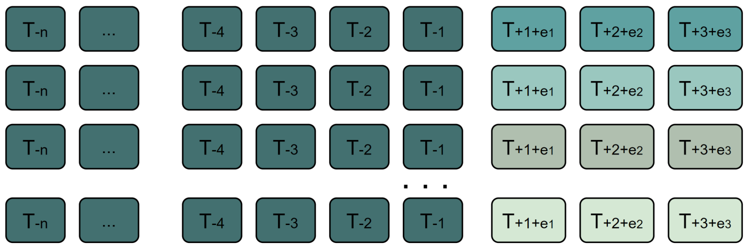


Figure 5-2.: Bootstrapping predictions resulting from bootstrapping process. Figure is taken from Amat Rodrigo and Escobar Ortiz (2023).

Figure **5-3**. presents ten bootstrap samples generated for the calculation of the prediction intervals for the test data. The process shown in **5-1** is repeated for the creation of each bootstap sample shown in figures **5-2** and **5-3**.

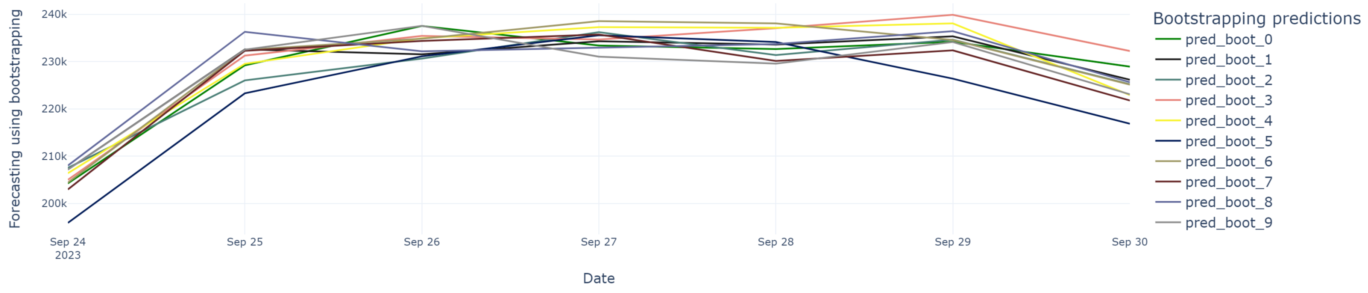


Figure 5-3.: Lasso ten Bootstrapping predictions.

5.2. Conformal Prediction

According to Kath and Ziel (2021), Conformal Prediction (CP) predicts intervals based on errors, makes weak assumptions about data characteristics, and can be used to create point prediction models. In comparison to other models, it provides more accurate and reliable prediction intervals than existing point prediction estimators. The article investigates the ability of CP to adapt to different market conditions while maintaining electricity price uncertainty.

The general procedure is as follows: Given a set of training data, a point prediction model is trained to predict the target variable. This model can be of any type, such as a linear regression model or a neural network. A test instance is then presented to the model, and it produces a point prediction. This point prediction is a single value that represents the model which give the best estimate of the target variable for the given input. Non-conformity scores are computed for the test instance. It measures how much the test instance deviates from the training data. This is done by comparing the test instance to the training data and computing a score that represents the degree of deviation. The non-conformity score can be computed in different ways, depending on the type of data and the model being used. In the equation (5-1) presents a way to calculate Non-conformity scores.

$$\lambda_t = |y_t - \hat{y}_t| \,. \tag{5-1}$$

These non-conformity scores are calculated for the test data, so they range from T+1 to M, where T is the observation where the training data ends. The aim of CP is to find the smallest value of lambda which satisfy the condition $r(\lambda) \geq 1 - \alpha$ where $r(\lambda)$ represents the fraction of training instances whose nonconformance values are lower than λ. In order for prediction intervals to be conformal, λ must be minimized, since it determines the lower bounds of a prediction interval. This is represented by equation (5-2).

$$\begin{aligned} \lambda^{\alpha}_{M+1} &= \min\left\{\lambda \in \{T+1, \ldots, M\} : r(\lambda) \geq 1-\alpha\right\}, \\ r(\lambda) &= \frac{\#\left\{t \in \{T+1, \ldots, M\} : \lambda_t < \lambda\right\} + 1}{\#Z_{test} + 1}. \end{aligned} \tag{5-2}$$

where Z_{test} is the test data set used to predict data not seen by training. The equation (5-3) represents the prediction interval.

$$y_{\alpha,M+1} = \hat{y}_{M+1} \pm \lambda^{\alpha}_{M+1}. \tag{5-3}$$

In this case, point T+1 is chosen to illustrate how the range prediction is performed when considering a future point in the time sequence. Therefore, when calculating the forecast interval for the forward point T+1, one is considering the information available up to point T to make the forecast and estimate the uncertainty associated with the forward projection in the time series. According to Stankeviciute et al. (2021), CP is a machine learning framework that provides theoretical guarantees of frequentist coverage for uncertainty estimation. In this framework, CP sets are constructed that contain the true label with a given probability of coverage. The paper proposes a framework for time-series forecasting that is based on RNNs and provides uncertainty estimates for any multi-horizon forecast predictor and any dataset.

According to Xu and Xie (2023), CP is a method for constructing prediction intervals that provides a guarantee of coverage without making assumptions about the underlying distribution of the data. It generates valid marginal coverage for the test point under the assumption of exchangeability in data. Exchangeability refers to the assumption that the order of the data points does not matter, and that the distribution of the data is invariant to permutations of the data points. In the context of CP, exchangeability is a key assumption that allows for the construction of prediction intervals with guaranteed coverage. However, this assumption is not always reasonable for time series data, which are often characterized by temporal dependencies and non-stationarity. As a result, adapting CP methods beyond exchangeable data has been gaining significant interest in recent years. The paper explores adapting CP for time series beyond exchangeable data, considering methods that account for unknown distribution shifts in test data. However, the assumption of data exchangeability in time series limits the direct applicability of these methods.

The paper of Xu and Xie (2023) proposes a new method called EnbPI that extends the existing literature on CP for time series by considering the case of dependent data and providing a unified treatment of conditional and marginal coverage guarantees. It ensures that the prediction intervals constructed using this method contain the true value of the time series with a given probability, both at specific time points and across all time points. A prediction range is generated by calculating a series of scores for every data point in the time series. As a result of these scores, a set of potential intervals is generated, and the shortest interval that satisfies the coverage probability is chosen as the final prediction range.

According to Xu and Xie (2023), the general procedure of EnbPI strategy for estimating prediction intervals on single-output time series is follow: A time series dataset and an ensemble of prediction algorithms are the inputs. For each prediction algorithm in the ensemble, train the prediction algorithm on a bootstrap sample of the time series dataset, use the trained algorithm to make predictions for the entire time series, compute the prediction errors (the difference between the predicted and actual values) for each time point in the time series. And compute the conformal scores (a measure of how well the prediction errors conform to a certain distribution) for each time point in the time series. In summary, they apply each prediction algorithm in the ensemble to the bootstrap sample of the time series dataset and compute the conformal scores for each time point in the time series. Since we have multiple prediction algorithms in the ensemble, we obtain multiple sets of conformal scores for each time point.

The following is to combine these multiple sets of conformal scores to obtain a single set of conformal scores for each time point in the time series. This is typically done by taking the average of the conformal scores across all the prediction algorithms in the ensemble. The idea behind this step is to obtain a more robust estimate of the conformity of the prediction errors to a certain distribution by combining the information from multiple prediction algorithms. Use the conformal scores to construct a set of candidate prediction intervals for each time point in the time series, and choose the shortest prediction interval among the candidate intervals that satisfies a given coverage probability (Xu and Xie, 2023).

The EnbPI algorithm is designed to be computationally efficient and scalable to large datasets. This method takes into account the dynamics of time series data and provides prediction intervals that adapt to the specific characteristics and noise present in the time series. The EnbPI algorithm is based on the concept of conformity scores, which measure the conformity of predictions with the training data, and uses block bootstrapping to generate prediction intervals that capture the uncertainty in the time series data. This approach allows for more accurate and reliable uncertainty estimates in time series analysis (Xu and Xie, 2023).

In conformal prediction method, α is one of the key parameters used to construct prediction intervals, it is between 0 and 1, and represents the uncertainty of the confidence interval. A lower α produces a larger or more conservative prediction interval. And a larger α implies a narrower interval, or in other words, it generates a more regularized interval (Xu and Xie, 2023).

5.3. Evaluating prediction intervals

Gao et al. (2022) showed one way to evaluate prediction intervals, they used prediction interval coverage probability (PICP) which is presented in the equation (5-4). The prediction interval with larger PICP value means that the prediction interval obtained is more reliable.

$$PICP = \frac{1}{T}\sum_{t=1}^{T}\varepsilon_t, \text{where} \quad \varepsilon_t = \begin{cases} 0, & y_t \notin [\ell_{\alpha,t}, u_{\alpha,t}] \\ 1, & y_t \in [\ell_{\alpha,t}, u_{\alpha,t}] \end{cases}, \tag{5-4}$$

where ε_t denotes the boolean variable, y_t represents the dependent variable and $u_{\alpha,t}$ and $\ell_{\alpha,t}$ represent the upper and lower bound of corresponding interval. According to Nowotarski and Weron (2018) and Kath and Ziel (2021) there are also other metrics such as the winkler score which is also used to quantify the reliability and the width of the prediction interval. Zhang et al. (2016) introduced a specific metric for evaluating prediction intervals, thereby extending the estimation of quantiles, known as the Pinball Loss Function. This function plays a crucial role in assessing the accuracy and precision of these prediction intervals, thereby enhancing the comprehensive understanding of quantiles within the domain of quantile predictive modelling. In this development the PICP was used to assess the coverage or width. Finally, the winkler score metric was not used because the Python development was not reliable. And pinball loss was also not used, since its focus is on assessing accuracy in quantile regression.

5.4. Prediction intervals results

This section is dedicated to expose the results obtained from applying the methods to calculate the forecast intervals, the order of this section is as follows: First we will expose the prediction intervals using bootstrap for the chosen model in ML section 4, Lasso. Second we will expose the prediction intervals using CP for the chosen model. The scope of this project is to calculate the prediction intervals only for the chosen model, i.e. the model with the highest point predictive capability. And the limitation to calculate prediction intervals in DL models are limited because the development of the code used does not incorporate the open source library written in Python that provides an interface to build and train neural network models used in this project. The metrics obtained are global, the coverage is an average of all the points that do fall within the interval.

The prediction intervals based on bootstrap residuals for validation data is presented in the Figure **5-4**, which was obtained with an effective coverage score of 84.34 % which means that approximately 84 % of the predictions in the validation data are within the prediction interval. There are some actual electricity demand points that do not fall within the range and this is because the range does not contain 100 % of the forecast values.

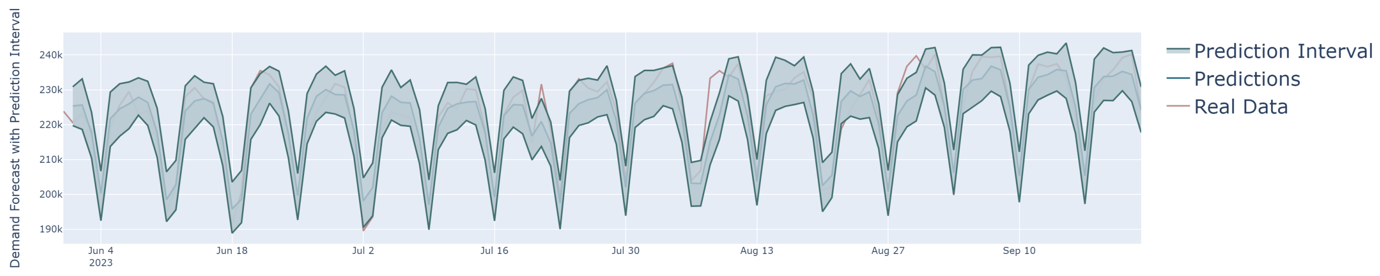


Figure 5-4.: Predictions, Lasso bootstrap prediction intervals for validation data and Real demand.

Figure **5-5** shows the evolution of the mean of the variable "amplitude"through different time intervals for the validation data, the amplitude understood as the difference between the intervals. The amplitude in the prediction interval was calculated for the validation data, divided into 4 groups, in which statistics were calculated to show the difference in each interquartile range, and confirming that the prediction intervals are increasing over time. When comparing the groups, an increase in the median is observed as the temporal distance between the point of origin of the forecast increases. In addition, an increase in the maximum value of the amplitude is observed in each successive group. This reaffirms that there is greater uncertainty in the predictions as the temporal distance increases, which is consistent with the expectation that the prediction intervals grow over time to reflect the greater uncertainty in the predictions at longer prediction horizons.

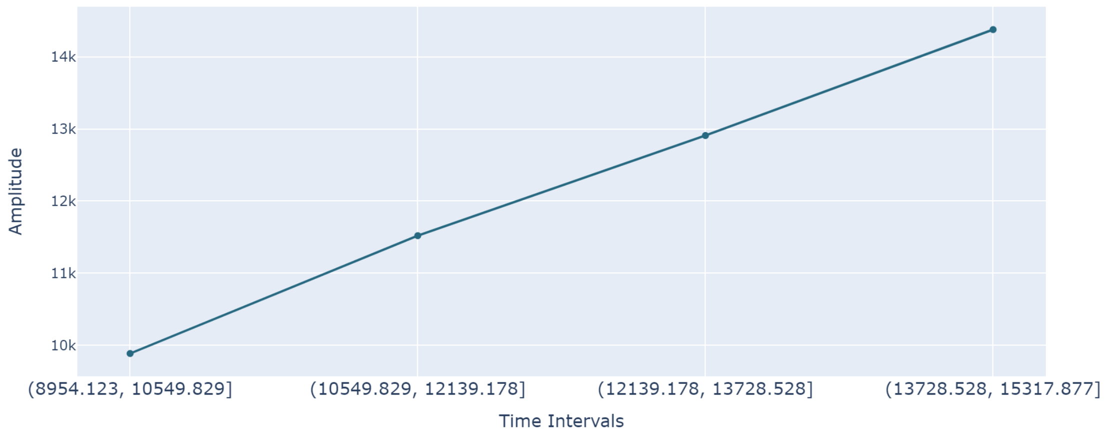


Figure 5-5.: Amplitude over time for validation data in Lasso bootstrap prediction intervals.

The predicted interval's coverage on the test data reached 71.42 %, it is shown in Figure **5-6**. This indicates a sustained high coverage for the test data. One possible reason for this difference in metrics may be the difference between the amount of validation and test data. The mean of the variable "amplitude"through different time intervals for the test data is

6703.599 and 7600.718. From which, it is inferred that the uncertainty of the predictions increases as the predictions increase as in validation data.

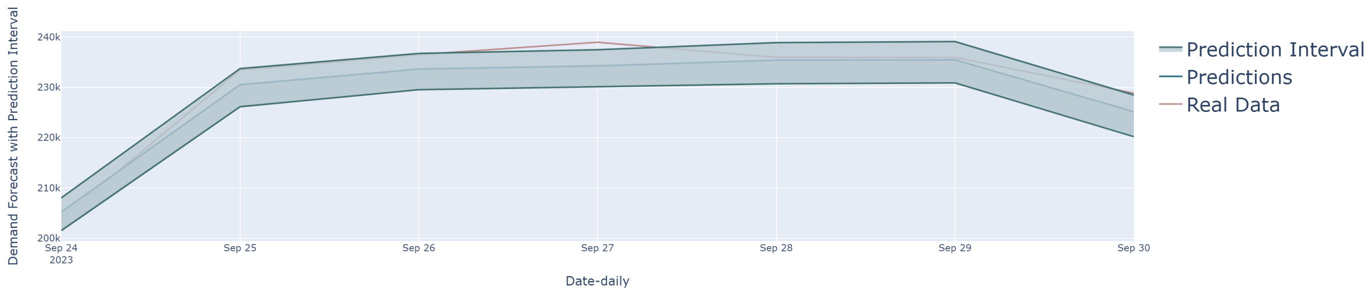


Figure 5-6.: Predictions, Lasso bootstrap prediction intervals for test data, last seven days of database and Real demand.

In Figure **5-7** shows the kernel density estimation for each step by using the bootstrapped predictions. It represents the level of uncertainty in a forecasting model, because it's possible to determine the potential values that each point can assume. The details of this estimate are shown in Figure **5-2** of the section 5.1. Finally, it should be noted that the actual values are in fact in each density estimation at each point. Table **5-1** shows the actual electricity demand values for the last seven days, with the respective values of the prediction interval, which are being used for test data.

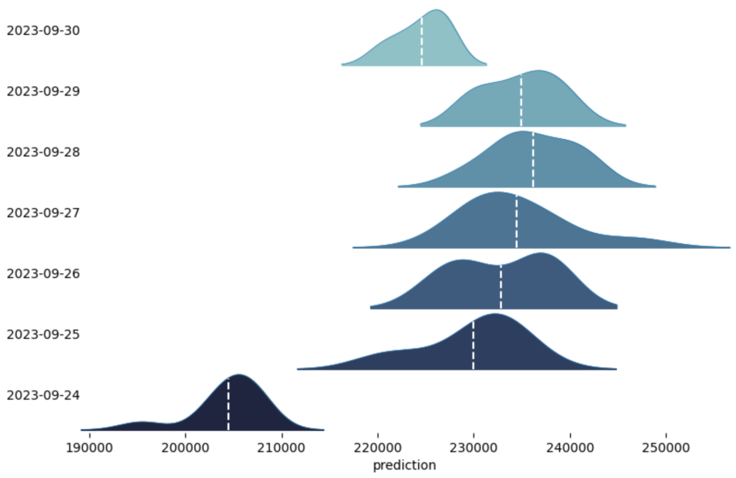


Figure 5-7.: Forecasting distribution per step in the test data.

Lower bound	Real demand	Upper bound
201525.328	203949.278	208032.358
226146.446	233453.085	233724.306
229528.894	236506.302	236744.624
230085.730	238964.909	237464.804
230701.488	235940.357	238876.106
230874.255	235929.693	239095.187
220185.397	228839.628	228462.696

Table 5-1.: Real demand for the test data with the prediction intervals.

Figure **5-8** shows the prediction interval for validation data calculated with EnbPI proposed by Xu and Xie (2023) for selected model, whose coverage is 84.3 % and its effective mean width obtained was 32211.192 which means that 84 % of the validation data are within the prediction interval.

The Figure **5-9** shows the prediction interval for test data calculated with EnbPI proposed by Xu and Xie (2023) for selected model, Lasso whose coverage is 85.7 % and its effective mean width obtained by the prediction intervals was 32569.631 which means that 85 % of the test data are within the prediction interval and the coverage obtained by CP is higher than that obtained via bootstrap, which was 71 %. Based on the above, one possible deduction is that by using CP, a higher interval coverage could be achieved, but at the expense of longer intervals. Figures **5-8** and **5-9** also show a progressive increase in bias in the predictions as the temporal distance increases, which leads to greater risk in both the point predictions and the interval predictions. And finally, the Table **5-2** shows the interval values for the test data. For the conformal prediction case, the amplitude is constant, so only the value of the amplitude of the entire prediction interval was obtained.

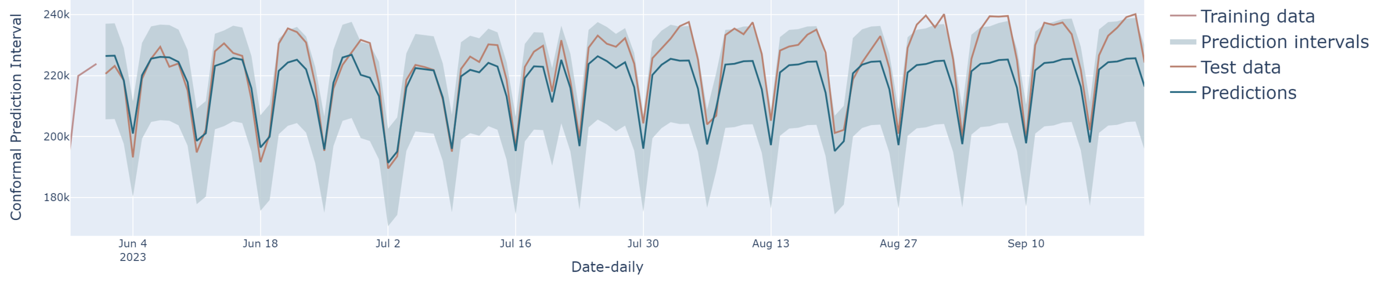


Figure 5-8.: Predictions, Lasso Conformal Prediction Intervals for validation data of database and Real demand.

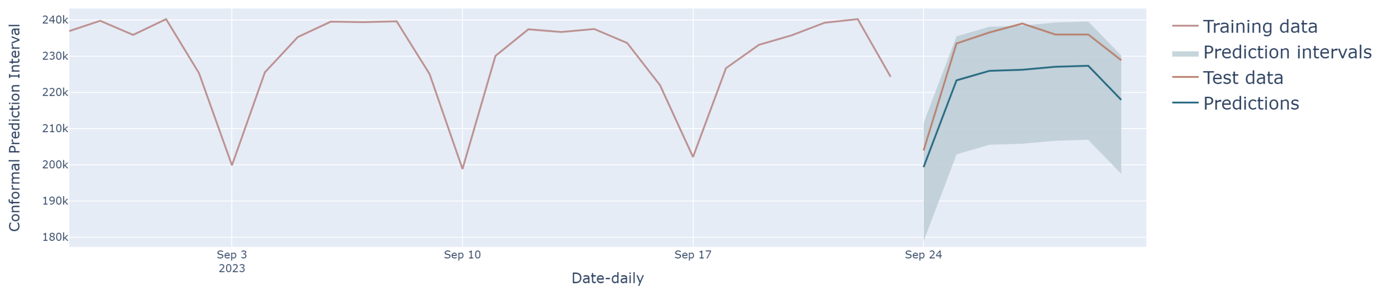


Figure 5-9.: Predictions, Lasso Conformal Prediction Intervals for test data, last seven days of database and Real demand.

Lower bound	**Real demand**	**Upper bound**
178922.298	203949.278	211491.929
202871.824	233453.085	235441.456
205497.578	236506.302	238067.210
205782.182	238964.909	238351.814
206608.394	235940.357	239178.025
206887.708	235929.693	239457.340
197504.247	228839.628	230073.879

Table 5-2.: Real demand for the test data with the prediction intervals.

6. Concluding remarks

It is essential for power systems and energy providers to be able to accurately predict electricity demand. Over the past few years, big data and artificial intelligence have been increasingly integrated into several aspects of human life, especially deep learning. The development of electricity demand forecasting methods based on conventional approaches has been of great interest in the field. We proposed a framework to compare some statistical, ML and DL models to predict daily electricity demand in Colombia. Also, we propose the calculation of forecast intervals using the most common method for this purpose which is bootstrap and another more recently developed method which is conformal prediction. We first identify several factors that may affect electricity demand levels such as temporal patterns, calendar details and climatic variables, calendar variables. Second, some popular ML and DL models were used for electricity demand forecasting and feature selection. The models trained were: Ridge, which gave a MAPE of 1.016 %, followed by Lasso with a MAPE of 1.0176 %, ElasticNet, XGBoost, Radom Forest, Support Vector Machine, Simple RNN, LSTM, GRU and finally Bidirectional LSTM. Starting from the model with the best predictive performance given the MAPE to the worst performance.

Our results demostrates that (1) for the daily energy demand of Colombia, with the variables obtained at the same frequency, a simple model works better than an advanced and much more complex model. In short-term predictive performance, the model that outperformed was a regularized linear model, Lasso, surpassing the capability of a more complex model such as a deep learning model. Also because the proportion of bias in its forecasts is small compared to the other models, and finally, because its in-sample fit performs acceptably.

(2) Regarding feature selection concerns, the most important variables are the energy demand lags and demand structure variables for the Lasso model, which works as a feature selection method, due to its regularization nature. This confirms that the inclusion of lags or having an autocorrelated structure is important in this type of problem. It is also worth mentioning that it may be the case where the lags of the covariates could also influence, for future evolution of this idea, it would be plausible to inquire the correct way to include these lags and what additional processes would be required. The final set of variables used were: The type of day. Likewise, the first fourteen energy demand lags, the indicator variable of the COVID-19 period, the indicator variable of the level change generated since the beginning of the pandemic and finally four weather variables were taken into account:

Total daily precipitation, Maximum daily wet temperature, Average daily wind speed and Daily persistence. This was also supported by testing different combinations of variables and obtaining the predictive ability measure.

Our other focus was the calculation of forecast intervals, these prediction intervals were calculated for the chosen model, Lasso, in which we use two methods. The first and most common was the bootstrap method and the second, whose development is more recent, is Conformal Prediction. It is important to mention that one of the important conclusions is that the topic of conformal prediction was introduced in its generality for the calculation of prediction intervals and also focused on time series. On the one hand, in the prediction intervals calculated with bootstrap for validation data, effective coverage score of 84.34 % which means that approximately 84 % of the predictions in the validation data are within the prediction interval. The predicted interval's coverage on the test data reached 71.42 %, it can be seen that for the validation data there is a high coverage. Finally, a basic analysis of the width of the prediction interval was performed.

In contrast to the prediction intervals calculated via Conformal prediction, which has a coverage of 84 % for validation data and 85 % for test data, but a constant amplitude. It is important to emphasize that the construction of prediction intervals allowed us to give a 99 % confidence level to the predicted point and not just rely on the comparison between the actual and predicted values. This means that there is a 99 % probability that the predicted values are contained in the calculated interval. The process of calculating the prediction intervals was performed only for the chosen model, however, for a future evolution of this idea, it would be plausible also to compare the capability of the mentioned methods with different models.

A. Appendix:

A.1. Nomenclature of the categories of the day type variable

category	Nomenclature
1st January	1–Janu
1st May	1–May
2nd January	2–Janu
20th July	20–July
24th December	24–Dec
25th December	25–Dec
31st December	31–Dec
7th August	7–Aug
8th December	8–Dec
Sundays before a Holiday Monday	SunBHolMonday
Sundays before a Holiday Monday in January	SunBHolMondayJanu
Sunday	Sunday
Sunday of december vacation	SunDecV
Sunday of january vacation	SunJanuV
Easter sunday	EasterSun
Easter thursday	EasterThurs
Thursday	Thursday
Thursday holidays in december	ThuHolD
Thursday holidays in january	ThuHolJanu
Holiday mondays	HolMonday
Holiday mondays in january	HolMondaJanu
Monday	Monday
Monday of december vacation	MondDecV
Monday of january vacation	MondJanuV
Tuesdays after a holiday monday	TueAHolMonday
Tuesdays	Tuesdays
Tuesdays of december vacation	TueDecV
Tuesdays of january vacation	TuesJanuV
Wednesday	Wednesday
Wednesday of december vacation	WedDecV
Wednesday of january vacation	WedJanuV
Easter wednesday	EasterWed
Saturdays before a Holiday Monday	SatBHolMonday
Saturdays before a Holiday Monday in January	SatuBHolMondayJanu
Saturdays	Saturdays
Saturdays of december vacation	SatDecV
Saturdays of January vacation	SatJanuV
Easter saturday	EasterSat
Friday	Friday
Friday of december vacation	FridDecV
Friday of january vacation	FridJanuV
Easter Friday	EasterFrid

Table A-1.: Nomenclature of the categories of the day type variable

A.2. Backtesting process in a flowchart

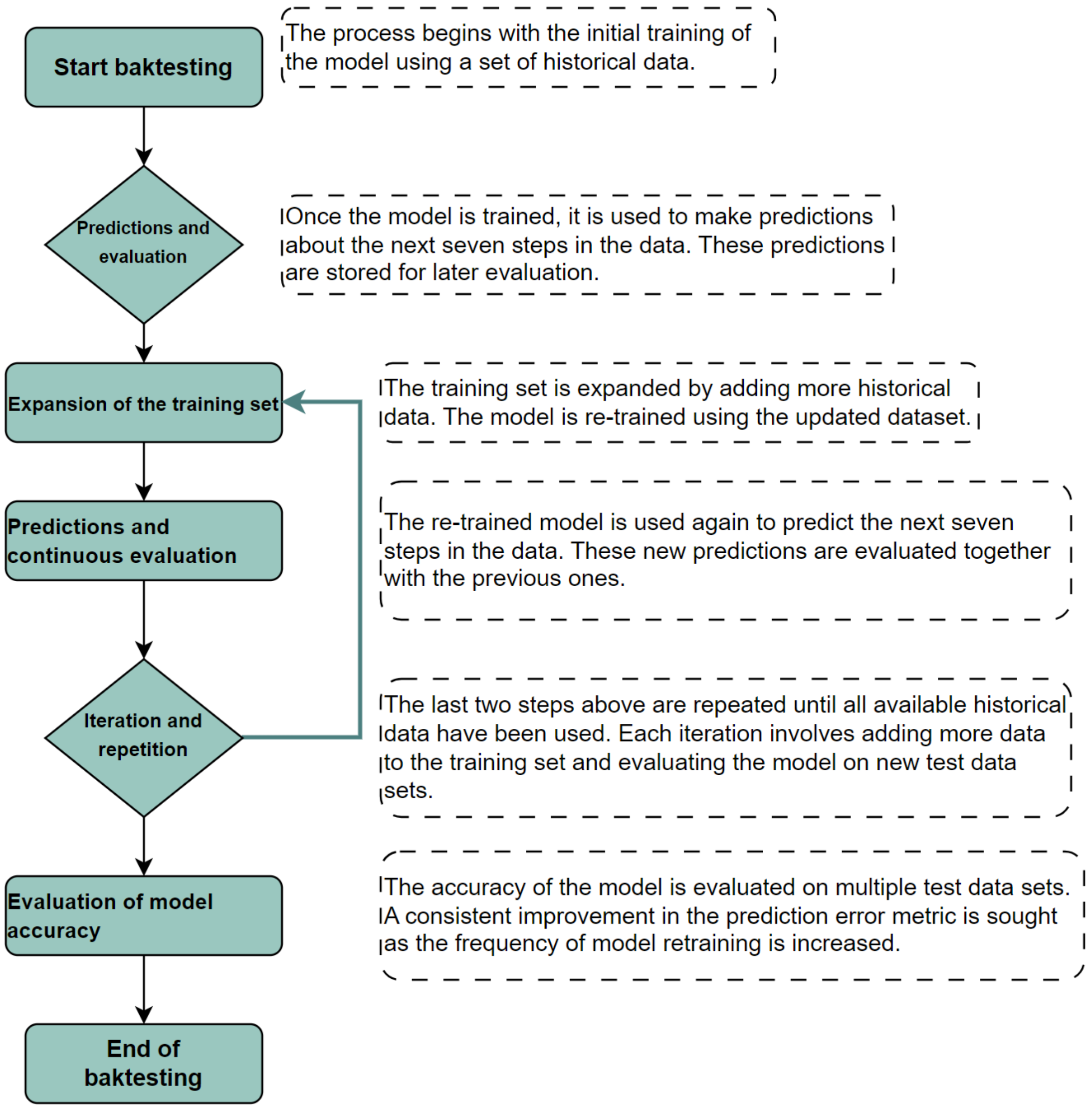

Start baktesting
The process begins with the initial training of the model using a set of historical data.
Predictions and evaluation
Once the model is trained, it is used to make predictions about the next seven steps in the data. These predictions are stored for later evaluation.
Expansion of the training set
The training set is expanded by adding more historical data. The model is re-trained using the updated dataset.
Predictions and continuous evaluation
The re-trained model is used again to predict the next seven steps in the data. These new predictions are evaluated together with the previous ones.
Iteration and repetition
The last two steps above are repeated until all available historical data have been used. Each iteration involves adding more data to the training set and evaluating the model on new test data sets.
Evaluation of model accuracy
The accuracy of the model is evaluated on multiple test data sets. A consistent improvement in the prediction error metric is sought as the frequency of model retraining is increased.
End of baktesting

Figure A-1.: Backtesting with refit and increasing training size (fixed origin) process in a flowchart.

A.3. Ensemble methods details

According to Chen and Guestrin (2016), the objective function is given by (A-1), for a given data set with t examples and m features $X = \{x^1, x^2, \cdots, x^m\}$ which are within the data set $D = \{(x_t, y_t)\}$, a tree ensemble model uses K additive functions to predict the output.

$$\hat{y}_t = \phi(x_t) = \sum_{k=1}^{K} f_k(x_t), \quad f_k \in \mathcal{F}. \tag{A-1}$$

Where $\mathcal{F} = \{f(x) = w_q(x), q : R^m \to \tau, w \in R^\tau\}$ is the space of regression trees. Here q represents the structure of each tree that maps an example to the corresponding leaf index. τ is the number of leaves in the tree. Each f_k corresponds to an independent tree structure q and leaf weights w. Unlike decision trees, each regression tree contains a continuous score on each of the leaves, we use w_i to represent score on i-th leaf. For a given example, we will use the decision rules in the trees (given by q) to classify it into the leaves and calculate the final prediction by summing up the score in the corresponding leaves (given by w). To learn the set of functions used in the model, we minimize the following regularized objective.

$$\begin{aligned} &\mathcal{L}(\phi) = \sum_{t=1}^{T} \iota(y_t, \hat{y}_t) + \sum_{k=1}^{K} \Omega(f_k), \\ &\text{where} \\ &\Omega(f_k) = \gamma\tau + \frac{1}{2}\lambda \left\| w_k \right\|^2 . \end{aligned} \tag{A-2}$$

Here ι is a differentiable convex loss function that measures the difference between the prediction $\hat{y}_t$ and the target y_t. The term, $\sum_{t=1}^{T} \iota(y_t, \hat{y}_t)$, represents sum of the loss function for each data point t. The second term Ω penalizes the complexity of the model (i.e., the regression tree functions). Here, γ and λ are parameters controlling the regularization strength. The term $\frac{1}{2}\lambda \sum_{j=1}^{\tau} w_{jk}^2$ penalizes complex models by considering the weights (w_{jk}) assigned to each leaf node in the decision trees. The goal is to find values for these weights that minimize the loss function. The additional regularization term helps to smooth the final learn weights to avoid over-fitting. Intuitively, the regularized objective will tend to select a model employing simple and predictive functions. When the regularization parameter is set to zero, the objective falls back to the traditional gradient tree boosting.

The sum of predictions from all decision trees f_k in the model are trees which collectively form the space $\mathcal{F}$ encompasses the space of all conceivable decision trees. And finally, the model parameters involve the number of decision trees K, the maximum depth of each tree τ or the number of leaves, and regularization parameters γ and λ. In summation, this regularized learning objective creates the training of the XGBoost model by simultaneously minimizing the loss function, ensuring accurate predictions, and incorporating regularization to control model complexity.

A.4. Deep Learning models details

In Zhang et al. (2023), a neural network representation is possible for linear regression where each feature is represented by an input neuron, which are all directly connected to the output. The inputs are denoted as $x_1, \ldots, x_d$ and the output is represented as o_1. In Equation A-3, the loss function for this case is shown.

$$L(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^{n} l^{(i)}(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} \left(\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)} \right)^2 . \tag{A-3}$$

The corresponding weights, which are different for each i, are represented by a vector $\mathbf{w} \in R^d$ and the term b is referred to as a bias. Each weight determines the impact of its corresponding feature on the prediction, while the bias contributes to the predictions values when all features are zero. Essentially, the bias introduces flexibility and learning capacity to the network, influencing activation points and facilitating complex modeling of data. The graphical representation of linear regression as a neural network is illustrated in Figure **A-2**, which shows the general connectivity pattern Zhang et al. (2023).

Mathematically, the prediction in a linear regression model $\hat{y}$ can be expressed as $\hat{y} = w_1 x_1 + \cdots + w_t x_t + b$. The equation represents the linear transformation of input features with a bias. It is crucial to understand fundamental concepts for comprehending NN operations. These considerations arise from the limitations of linear models, particularly their inherent linearity, which assumes monotonicity. Linear models can be improved by introducing hidden layers. A simple solution is to place multiple fully connected layers sequentially. Conceptually, the first $L - 1$ layers capture a representation of the data, and the last layer serves as our linear predictor Zhang et al. (2023).

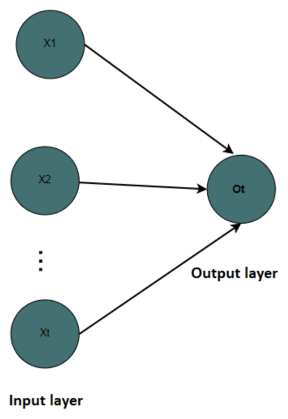


Figure A-2.: Linear regression as a single layer NN architecture, figure taken from Zhang et al. (2023).

This assumption suggests that any increase in a feature should consistently result in either an increase or a decrease in the model's output. Now, the interesting part is that although the relationship is monotonic, it does not mean that it is linear. Therefore, may require more complex modeling techniques than simply fitting a straight line.

We will now introduce some technical concepts by referencing from Zhang et al. (2023); h_t refers to the hidden state at time t, is an internal representation, it is used to capture and retain relevant information from previous inputs in the sequence. h_t is commonly considered a latent variable, a latent variable is a variable that is not directly observable but is inferred from other observable variables, in NN, it can be expressed as $h_t = f(y_t, h_{t-1})$.

Activation functions, they decide whether a neuron should activate based on the weighted sum of inputs and an added bias. These functions are like buttons, converting input signals into outputs and introducing non-linearity to the model. Some of the most common activation functions are: Sigmoid Activation, Hyperbolic Tangent Activation, Rectified Linear Unit (ReLU), Exponential Linear Unit and Scaled Exponential Linear Unit.

In the paper developed by Ramachandran et al. (2017), the activation functions can be explored in more detail, since they investigate the impact of activation functions in deep networks and proposes a new activation function called Swish, which is proposed as a superior alternative to ReLU in deeper models. It sets out a list of activation functions against which the proposed new function is compared. In addition, automatic search techniques are used to discover multiple novel activation functions that are empirically evaluated to determine their effectiveness compared to ReLU in various application scenarios.

Simple Recurrent Neural Networks

Now, we will focus on Recurrent Neural Networks (RNNs), a type of deep learning model designed for understanding sequential data. It uses recurrent connections, creating cycles in the network to capture temporal dependencies in the input data, in other words, it dynamically transfers information across adjacent time steps. At each step, the same parameters are used. This has the advantage of maintaining a constant cost associated with describing and setting those parameters, regardless of how many time steps are involved in the process. The Figure **A-3** reveals the behavior described above.

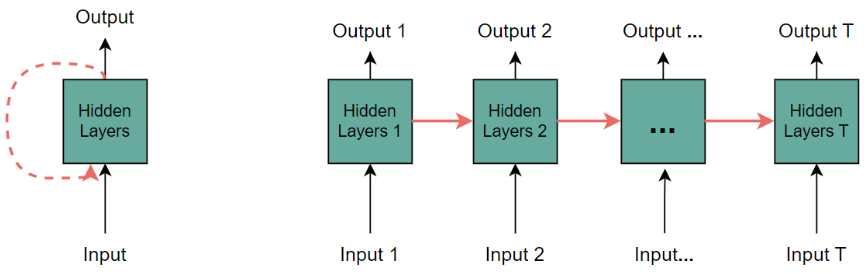


Figure A-3.: On the left recurrent connections are depicted via cyclic edges. On the right, we unfold the RNN over time steps. Taken from Zhang et al. (2023).

As explained by Zhang et al. (2023), suppose we have inputs denoted as $\mathbf{Y}_t \in R^{n \times t}$ at time step t. In simpler terms, for a mini-batch of n sequence examples, then, n is the number of instances considered simultaneously in a mini-bin during the training or processing of a recurrent neural network (RNN). Each row of $\mathbf{Y}_t$ corresponds to an example at time step t in the sequence. Now, let's represent the output of the hidden layer at time step t as $\mathbf{H}_t \in R^{n \times h}$, which is presented in the equation (A-4). We retain the output of the hidden layer, $\mathbf{H}_{t-1}$, from the previous time step. The weight parameter $\mathbf{W}_{\text{hh}} \in R^{h \times h}$ is include, so that the output at any given time step is based on both, the input and the output at that time step.

$$\mathbf{H}_t = \phi(\mathbf{X}_t \mathbf{W}_{\text{xh}} + \mathbf{H}_{t-1} \mathbf{W}_{\text{hh}} + \mathbf{b}_{\text{h}}). \tag{A-4}$$

Here, ϕ is the activation function, $\mathbf{W}_{\text{xh}}$ represents the weight parameters for the input, $\mathbf{W}_{\text{hh}}$ represents the weight parameters for the hidden layer, and $\mathbf{b}_{\text{h}}$ is the bias term for the hidden layer. The connection between the outputs of the hidden layer ($\mathbf{H}_t$) and the previous time step ($\mathbf{H}_{t-1}$) in consecutive time steps, serves as the memory or state of the NN at the current time step, leading to the term "hidden state"for such hidden layer outputs.

At time step t, the output layer is calculated as $\mathbf{O}_t = \mathbf{H}_t \mathbf{W}_{\text{hq}} + \mathbf{b}_{\text{q}}$. This output parameters include the weights $\mathbf{W}_{\text{hq}} \in R^{h \times q}$ and the bias $\mathbf{b}_{\text{q}} \in R^{1 \times q}$.

Long Short-Term Memory Neural Networks

A Long Short-Term Memory (LSTM) network captures long-term dependences in sequential data by overcoming the vanishing gradient problem. Continuing with the reference to Zhang et al. (2023), gradient problems occur when gradients become too small or too large, which make it difficult to capture long-term input and output dependencies. As a result, the model's performance and convergence can be adversely affected. From this intuition derives the term "long short-term memory".

According to Zhang et al. (2023), this memory is achieved through activations passing between nodes. LSTMs use a unique memory cell structure made up of simpler nodes arranged in a specific pattern and enhanced with multiplicative nodes. The internal state of each memory cell is controlled by various gates: the input gate affects the internal state, the forget gate decides whether to reset the internal state, and the output gate determines the cell's output based on its internal state. The key distinction between traditional RNNs and LSTMs is that LSTMs can manage the hidden state using gating mechanisms. In simply words, the input gate controls how much of the input should be added to the memory cell's current state, the forget gate decides whether to keep or reset the current memory value, and the output gate determines if the memory should influence the output at the current time step. The Figure **A-4** reveals the behavior described above.

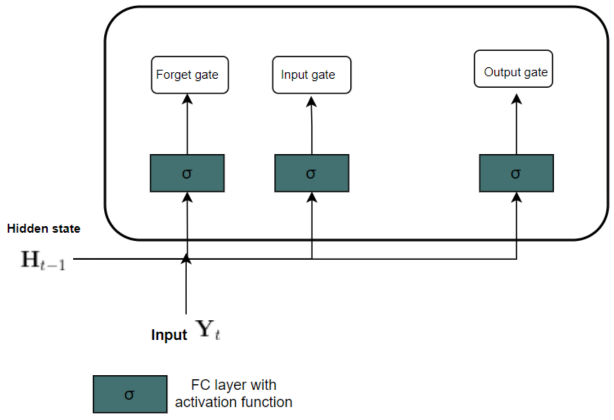


Figure A-4.: Computing the input gate, the forget gate, and the output gate in an LSTM model. Figure is taken from Zhang et al. (2023).

The input gate at time step t are defined as $\mathbf{I}_t \in R^{n\times h}$, the forget gate is $\mathbf{F}_t \in R^{n\times h}$, and the output gate is $\mathbf{O}_t \in R^{n\times h}$. These gates are calculated as follows:

$$\begin{aligned} \mathbf{I}_t &= \sigma(\mathbf{Y}_t\mathbf{W}_{\text{yi}} + \mathbf{H}_{t-1}\mathbf{F}_t &&= \sigma(\mathbf{Y}_t\mathbf{W}_{\text{yf}} + \mathbf{H}_{t-1}\mathbf{W}_{\text{hf}} + \mathbf{b}_{\text{f}}), \\ \mathbf{O}_t &= \sigma(\mathbf{Y}_t\mathbf{W}_{\text{yo}} + \mathbf{H}_{t-1}\mathbf{W}_{\text{ho}} + \mathbf{b}_{\text{o}}), \end{aligned} \tag{A-5}$$

where $\mathbf{W}_{\text{yi}}, \mathbf{W}_{\text{yf}}, \mathbf{W}_{\text{yo}} \in R^{d\times h}$ are weight parameters, and $\mathbf{b}_{\text{i}}, \mathbf{b}_{\text{f}}, \mathbf{b}_{\text{o}} \in R^{1\times h}$ are bias parameters. The bias parameters allow the NN to learn offsets and adjust the output of the gates. And σ is the activation function on all three gates. To define the actions of the gates, we introduce the input node $\tilde{\mathbf{C}}_t \in R^{n\times h}$. Its computation is analogous to the previously discussed gates but involves a tanh function. At time step t, this is expressed as:

$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{Y}_t\mathbf{W}_{\text{yc}} + \mathbf{H}_{t-1}\mathbf{W}_{\text{hc}} + \mathbf{b}_{\text{c}}), \tag{A-6}$$

where $\mathbf{W}_{\text{yc}} \in R^{d\times h}$ and $\mathbf{W}_{\text{hc}} \in R^{h\times h}$ are weight parameters and $\mathbf{b}_{\text{c}} \in R^{1\times h}$ is a bias parameter. And tanh is the activation function for the input node. The input gate $\mathbf{I}_t$ controls how much new information ($\tilde{\mathbf{C}}_t$) influences the cell's internal state, while the forget gate $\mathbf{F}_t$ determines how much of the old internal state ($\mathbf{C}_{t-1} \in R^{n\times h}$) is retained. The update equation is:

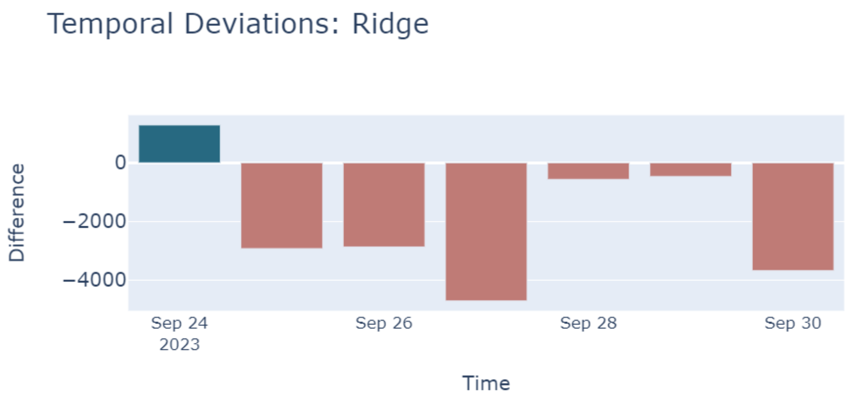
$$\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \tilde{\mathbf{C}}_t. \tag{A-7}$$

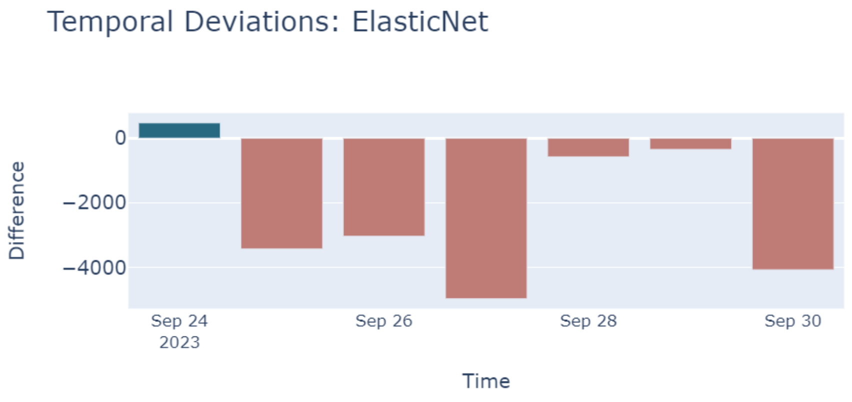
The symbol $\odot$ represents the element-wise multiplication, it means that each element of matrix $\mathbf{F}_t$ is multiplied by the corresponding element of matrix $\mathbf{C}_{t-1}$, in the same way each element of matrix $\mathbf{I}_t$ is multiplied by the corresponding element of matrix $\tilde{\mathbf{C}}_t$. If the forget gate is always 1 and the input gate is always 0, the memory cell $\mathbf{C}_{t-1}$ remains constant, unchanged in each step. This helps overcome the vanishing gradient problem, making models easier to train, especially with long sequences.

Finally, let's define how to calculate the output of the memory cell, represented as the hidden state $\mathbf{H}_t \in R^{n\times h}$ seen by other layers. tanh function is used to compute the internal state of the memory cell and then perform elementwise multiplication, this time with the output gate.

$$\mathbf{H}_t = \mathbf{O}_t \odot \tanh(\mathbf{C}_t). \tag{A-8}$$

A.5. Analysis details for statistical and ML models





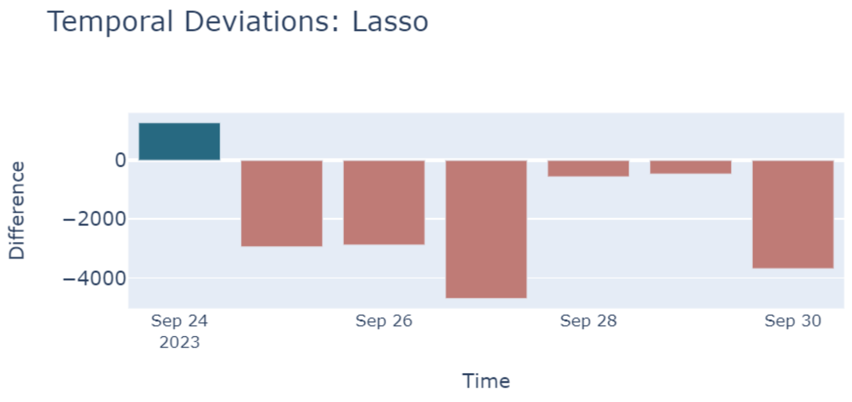


Figure A-5.: Temporal desviations for Ridge, Elasticnet and Lasso models.

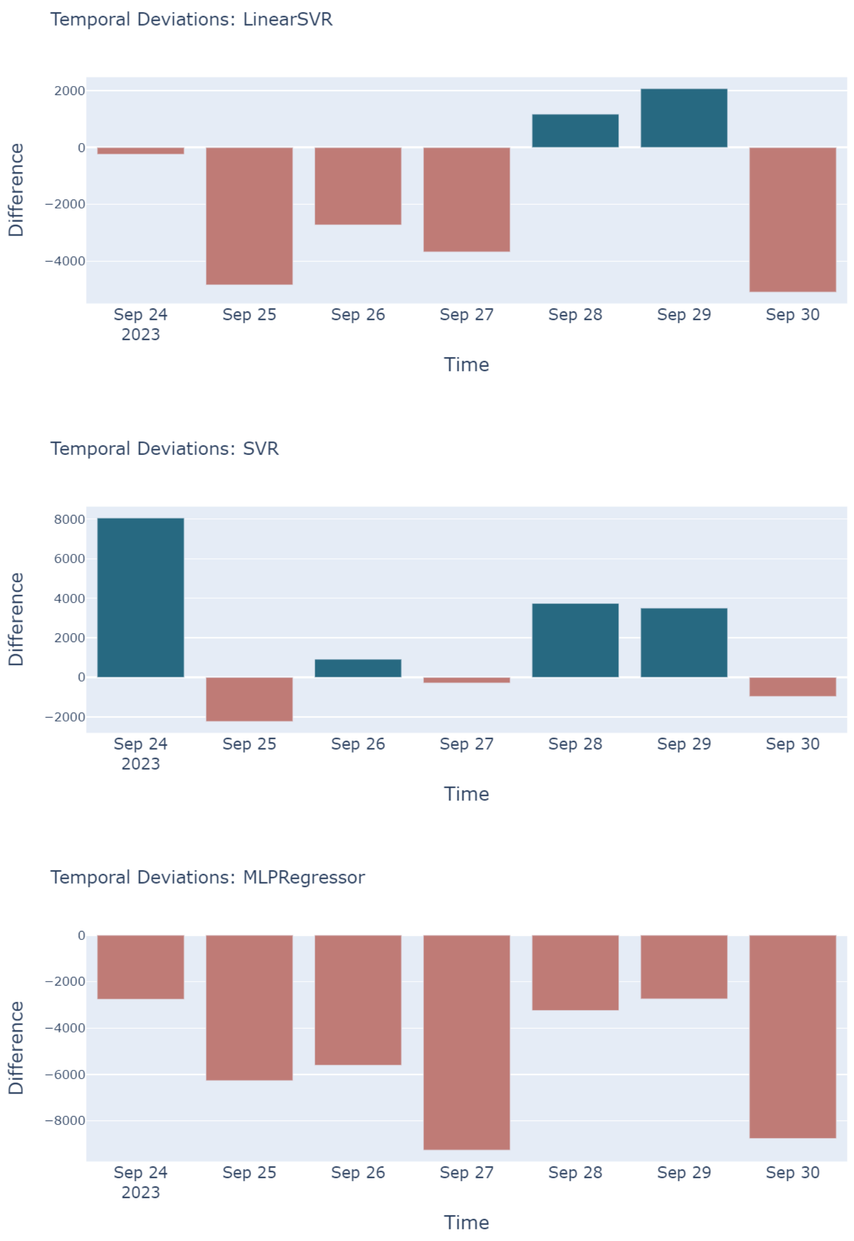


Figure A-6.: Temporal desviations for SRV and MLP models.

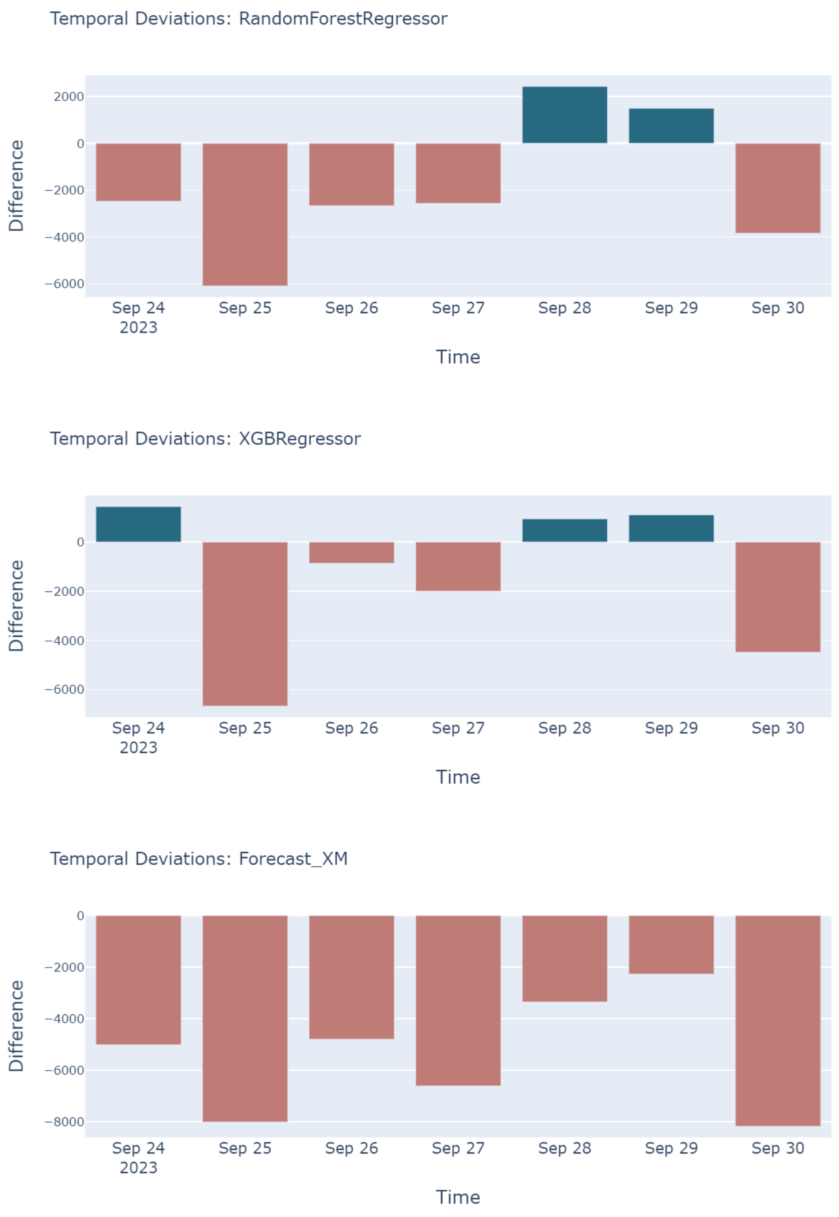


Figure A-7.: Temporal desviations for Ensemble and XM models.

Model	**MAPE (%)**	**MAE (KWh)**	**RMSE (KWh)**
Ridge	1.588	3527.80	21429788.70
Lasso	1.587	3526.43	21372590.07
ElasticNet	1.596	3546.99	20740280.68
XGBRegressor	1.868	4150.78	29030830.76
RandomForestRegressor	1.830	4067.22	30470867.60
LinearSVR	5.965	13251.11	466346447.47
SVR	2.446	5434.41	47007724.90
MLPRegressor	2.052	4559.17	34883771.44

Table A-2.: In-sample predictive performance metrics.

A.6. Code repository

The code for this project can be found in the following repository:

`https://github.com/JenniferMC22/ProbabilisticForecastingElectricityDemandColombiaTESISUNAL/tree/main`

References

Al-Musaylh, M. S., Deo, R. C., Adamowski, J. F., and Li, Y. (2018). Short-term electricity demand forecasting with mars, svr and arima models using aggregated demand data in Queensland, Australia. *Advanced Engineering Informatics*, 35(16):1–16.

Amat Rodrigo, J. and Escobar Ortiz, J. (2023). skforecast (version 0.11.0) [computer software]. https://doi.org/10.5281/zenodo.8382788.

Bedi, J. and Toshniwal, D. (2019). Deep learning framework to forecast electricity demand. *Applied Energy*, 238(C):1312–1326.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Brownlee, J. (2017). *Introduction to Time Series Forecasting With Python: How to Prepare Data and Develop Models to Predict the Future*. Machine Learning Mastery (Publisher).

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16. ACM.

Clements, M. P. and Kim, J. H. (2007). Bootstrap prediction intervals for autoregressive time series. *Computational Statistics Data Analysis*, 51(7):3580–3594.

Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Number 57 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Boca Raton, Florida, USA.

Fabbiani, E., Marziali, A., and Nicolao, G. D. (2021). Ensembling methods for countrywide short-term forecasting of gas demand. *International Journal of Oil, Gas and Coal Technology*, 26(2):184.

Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 29(5):1189–1232.

Friedman, J. H., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Gao, T., Niu, D., Ji, Z., and Sun, L. (2022). Mid-term electricity demand forecasting using improved variational mode decomposition and extreme learning machine optimized by sparrow search algorithm. *Energy, Elsevier*, vol. 261(PB).

Gneiting, T. and Katzfuss, M. (2014). Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(1):125–151.

Guo, Z., Zhou, K., Zhang, X., and Yang, S. (2018). A deep learning model for short-term power load and probability density forecasting. *Energy*, 160(C):1186–1200.

Huurman, C., Ravazzolo, F., and Zhou, C. (2012). The power of weather. *Computational Statistics Data Analysis*, 56(11):3793–3807. 1st issue of the Annals of Computational and Financial Econometrics Sixth Special Issue on Computational Econometrics.

Jensen, V., Bianchi, F. M., and Anfinsen, S. N. (2024). Ensemble conformalized quantile regression for probabilistic time series forecasting. 1-12.

Jiménez, J., Pertuz, A., Quintero, C., and Montaña, J. (2019). Multivariate statistical analysis based methodology for long-term demand forecasting. *IEEE Latin America Transactions*, 17(01):93–101.

Kath, C. and Ziel, F. (2021). Conformal prediction interval estimation and applications to day-ahead and intraday power markets. *International Journal of Forecasting*, 37(2):: 777–799.

Khosravi, A., Nahavandi, S., and Creighton, D. (2013). Quantifying uncertainties of neural network-based electricity price forecasts. *Applied Energy*, 112:120–129.

Kilian, L. (1998). Small-sample confidence intervals for impulse response functions. *The Review of Economics and Statistics*, 80(2):218–230.

Li, R., Chen, X., Baleentis, T., Treimikienė, D., and Niu, Z. (2020). Multi-step least squares support vector machine modeling approach for forecasting short-term electricity demand with application. *Neural Computing and Applications*, 33:301–320.

Maciejowska, K., Nowotarski, J., and Weron, R. (2016). Probabilistic forecasting of electricity spot prices using factor quantile regression averaging. *International Journal of Forecasting*, 32(3):957–965.

Marino, A., Arango., A., Lotero, L., and Jimenez, M. (2021). Modelos de series temporales para pronóstoco de la demanda eléctrica del sector de explotación de minas y canteras en Colombia. *Revista EIA*, 18:77 – 99.

Masarotto, G. (1990). Bootstrap prediction intervals for autoregressions. *International Journal of Forecasting*, 6(2):229–239.

Misiorek, A., Trueck, S., and Weron, R. (2006). Point and interval forecasting of spot electricity prices: Linear vs. non-linear time series models. *Studies in Nonlinear Dynamics Econometrics*, 10(3):1–34.

Mohammed, A. and Kora, R. (2023). A comprehensive review on ensemble deep learning: Opportunities and challenges. *Journal of King Saud University - Computer and Information Sciences*, 35(2):757–774.

Nielsen, D. (2016). Tree boosting with xgboost-why does xgboost win every machine learning competition. Master's thesis, Norwegian University of Science and Technology.

Ning, Y., Zhao, R., Wang, S., Yuan, B., Wang, Y., and Zheng, D. (2022). Probabilistic short-term power load forecasting based on b-scn. *Energy Reports*, 8:646–655. The 2022 International Conference on Energy Storage Technology and Power Systems.

Nowotarski, J. and Weron, R. (2013). Computing electricity spot price prediction intervals using quantile regression and forecast averaging. HSC Research Reports HSC/13/12, Hugo Steinhaus Center, Wroclaw University of Technology.

Nowotarski, J. and Weron, R. (2018). Recent advances in electricity price forecasting: A review of probabilistic forecasting. *Renewable and Sustainable Energy Reviews*, 81:1548–1568.

Panagiotelis, A. and Smith, M. (2008). Bayesian density forecasting of intraday electricity prices using multivariate skew t distributions. *International Journal of Forecasting*, 24(4):710–727.

Pascual, L., Romo, J., and Ruiz, E. (2005). Bootstrap prediction intervals for power-transformed time series. *International Journal of Forecasting*, 21:219–235.

Ramachandran, P., Zoph, B., and Le, Q. V. (2017). Searching for activation functions. retrieved from https://arxiv.org/abs/1710.05941.

Son, N., Yang, S., and Na, J. (2020). Deep neural network and long short-term memory for electric power load forecasting. *Applied Sciences*, 10(18):6489.

Stankeviciute, K., M. Alaa, A., and van der Schaar, M. (2021). Conformal time-series forecasting. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages : 6216–6228. Curran Associates, Inc.

Ullah, F. U. M., Ullah, A., Khan, N., Lee, M. Y., Rho, S., Baik, S. W., and Bai, X. (2022). Deep Learning-Assisted Short-Term Power Load Forecasting Using Deep Convolutional LSTM and Stacked GRU. *Complexity*, 2022:1–15.

Weron, R. and Misiorek, A. (2008). Forecasting spot electricity prices: A comparison of parametric and semiparametric time series models. *International Journal of Forecasting*, 24(4):744–763.

Xu, C. and Xie, Y. (2023). Conformal prediction for time series. Preprint. https://arxiv.org/pdf/2010.09107.pdf.

Zhang, A., Lipton, Z. C., Li, M., and Smola, A. J. (2023). *Dive into Deep Learning*. Cambridge University Press. `https://D2L.ai`.

Zhang, Y., Liu, K., Liang, Q., and An, X. (2016). Deterministic and probabilistic interval prediction for short-term wind power generation based on variational mode decomposition and machine learning methods. *Energy Conversion and Management*, 112:208–219.

Zhao, J. H., Dong, Z. Y., Xu, Z., and Wong, K. P. (2008). A statistical approach for interval forecasting of the electricity price. *IEEE Transactions on Power Systems*, 23(2):267 – 276.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.