UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

Research on application of machine learning in energy supply bidding

Author:
Nazar LIUBAS

Supervisor:
Ph.D. Taras FIRMAN

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Department of Computer Sciences and Information Technologies
Faculty of Applied Sciences

Lviv 2023

Declaration of Authorship

I, Nazar LIUBAS, declare that this thesis titled, “Research on application of machine learning in energy supply bidding” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

Research on application of machine learning in energy supply bidding

by Nazar LIUBAS

Abstract

This thesis proposes an approach for time series forecasting in the electricity bidding domain. Machine learning and deep learning models were applied to address the considered problem. External factors were selected and validated to include in the models. Forecasting based electricity supply bidding strategy was proposed, with pros and cons outlined.

Acknowledgements

First, I want to thank my Mother for her constant support. Thank my supervisor, Dr. Taras Firman, who has guided and helped me during my work on this thesis. Thank the community of Ukrainian Catholic University that presented me with unforgettable times of my studying here.

Contents

List of Figures

List of Tables

List of Abbreviations

ISO	Independent System Operator
NYISO	New York Independent System Operator
LBMP	Locational Based Market Price
SCUC	Security Constrained Unit Commitment
RTD	Real Time Dispatch
CNN	Convolutional Neural Network
MAPE	Mean Absolute Percentage Error
IQR	Inter-Quartile Range
APE	Absolute Percentage Error
MSE	Mean Square Error
OLS	Ordinary Least Squares
PACF	Partial Autocorrelation Function

Dedicated to Armed Forces of Ukraine

Chapter 1

Introduction

1.1 Deregulated electricity markets

Firstly adopted in Chile in the 1980s [25], deregulated and demonopolized electricity markets then became vastly and quickly spread among developed countries, mainly, but not exceptionally, in North America and Europe [12]. The capitalist nature of these countries helped to develop, adapt and significantly improve the efficiency of such markets.

The main difference between such markets and government-regulated ones is that, unlike the latter, there are a lot of independent producers of electricity that compete to sell their produced electricity to independent consumers that compete to buy electricity. Producers aim to maximize profits, while consumers want to minimize costs. And there is where Independent System Operator (ISO) came into a stage [14]. Its responsibilities are to be a marketplace and balance the market so that demand is met, with the lowest total system costs, and with meeting other requirements and constraints, specific to the electricity domain. The overall schema of ISO can be seen in Fig. 1.1.

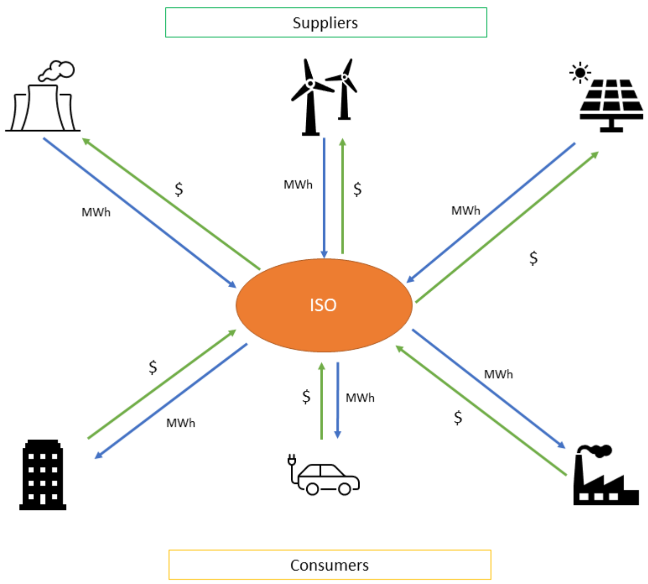

FIGURE 1.1: Independent System Operator schema

Usually, ISO operates in some regions — e.g., the National Grid in Great Britain [18] or the Nord Pool in most of Northern Europe [19]. In the USA, it is responsible for single or several states, e.g., California ISO for California [4], Midcontinent

ISO [1] for several states in the middle of the USA, and New York ISO for New York state [20].

Electricity undoubtedly plays an essential role in the everyday life of people. Also, a constant improvement process is ongoing because the electricity market process involves different parties with their goals that must be satisfied as effectively as possible. Related research topics include digitalization and algorithmization of trading, effective scheduling, dealing with the transmission system constraints and losses, cost optimization, system state control, supplier profit maximization, and others [16]. The aspect of effective and profitable supply offerings is addressed in this work.

Particularly, electricity systems improvements are even more wanted now in the context of the energy crisis caused by the Russian invasion of Ukraine.

Chapter 2

New York Independent System Operator overview

2.1 Market description

This work is focused on New York Independent System Operator (NYISO). The main reasons for such a choice are the availability of proper documentation and learning materials and a vast amount of well-structured real historical data about many system indicators. And, of course, it is operating in a big, deregulated, and competitive electricity market. Although all logic and methodology described below are specific to NYISO, with minor modifications, they can easily be applied to other ISOs, as the principles they operate are usually very similar.

The goal of NYISO is to provide a reliable electricity grid and competitive wholesale electricity market in the state of New York. They do not generate power or any transmission lines, but they work with power producers, utility companies, and stakeholders to provide power to meet New Yorkers' electricity needs on a daily, hourly, and minute-to-minute basis [20].

The overall process is the following:

1. Energy generators submit their electricity supply offers to the system. These offers are also referred to as generator bids. They provide information about the wanted price per Megawatt of electricity and the number of Megawatts they can offer. Also, additional operational information is submitted, such as the time at which they operate, startup time and cost, possible constraints, etc.;

2. At the same time as generators, electricity consumers submit the number of Megawatts they will consume at a certain operating hour, and the price they are willing to pay per Megawatt. These offers are also referred to as load bids.

3. Automated system solves the problem of scheduling, optimization, and balancing the market, so that demand is met, ideally with the lowest total system cost. The main things that are taken into account during this process:

    (a) Generator bids;

    (b) Load bids;

    (c) Grid constraints;

    (d) Outages;

    (e) Electricity losses in a grid;

    (f) Congestion of a grid;

    (g) Battery constraints;

(h) Weather forecast;

4. Schedule and Locational Based Market Price (LBMP), are posted on the website `https://www.nyiso.com/energy-market-operational-data`;

LBMP mentioned above is a cornerstone of NYISO operations, so additional explanations should be provided. As the definition states, it is a cost to provide the next MW of Load at a specific location in the grid [7]. It consists of three components and their relations are given by the Eq. 2.1:

$$LBMP = Marginal\ Energy + Marginal\ Loss - Marginal\ Congestion \tag{2.1}$$

Marginal cost is the cost of producing the energy needed to meet demand. It is set at the same level for all generators and load points for a certain period. This price is achieved in a way that system chooses generation bids to meet the demand, starting from the lowest one and gradually increasing it. The price of the last Megawatt provided before the demand was met is a marginal energy cost [28]. This can be visualized in figure 2.1.

Marginal loss is a cost of energy that should be generated additionally due to physical losses in a grid [8]. Usually, it is around 2% of the Locational Based Market Price.

Marginal congestion - is an additional price paid due to the transmission limits of lines or other grid constraints [27]. E.g., a system will choose a bid from A if there is a generator A that bids 40 \$/MW, and generator B that bids 50 \$/MWHr. However, there can be a case that the electricity line from A to the load point is already in use and cannot accept any more electricity. In such a scenario, a system will choose generator B, where the line is free, and pay him 50\$/MW. In this case, there will be a marginal congestion cost of 10\$/MWHr [17].

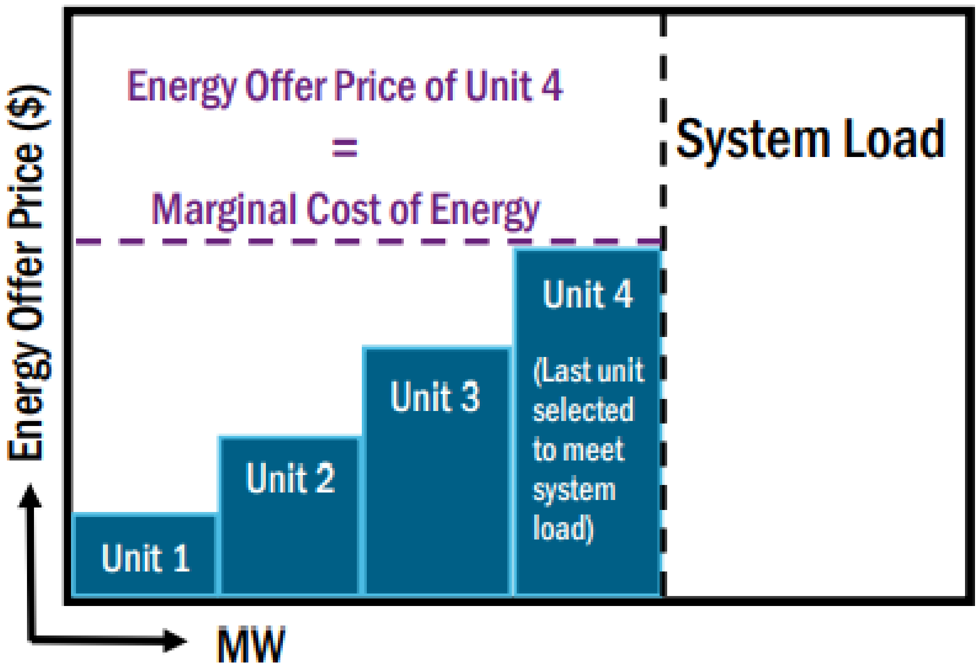

FIGURE 2.1: Marginal Energy Cost. Figure from [20]

Worth mentioning that sometimes LBMP can go negative. This can happen because there is more electricity generation available than load, and the price for electricity on the wholesale market can go negative to reduce electricity generation. Hence, it meets the load, through a market signal. Also, for example, wind resources benefit from federal production tax credits meaning they can often operate economically even while accepting negative prices [11]. However, only 2% of data possesses such property, so this fact should not cause any issues in the scope of this work.

NYISO operates as a two-settlement system, namely the Day Ahead and the Real Time market [29]. Day Ahead market works with estimations of load and creates schedules accordingly. These estimations may not be exact, so this is compensated with Real Time trade. Suppliers and consumers can submit demand or supply bids to cover differences in forecasts and real load. Timelines of the process can be seen in Fig. 2.2. Schedule and LBMP calculator, referred to above as an Automated system, is called Security Constrained Unit Commitment (SCUC) for the Day Ahead market and Real Time Dispatch (RTD) for the Real Time market [9].

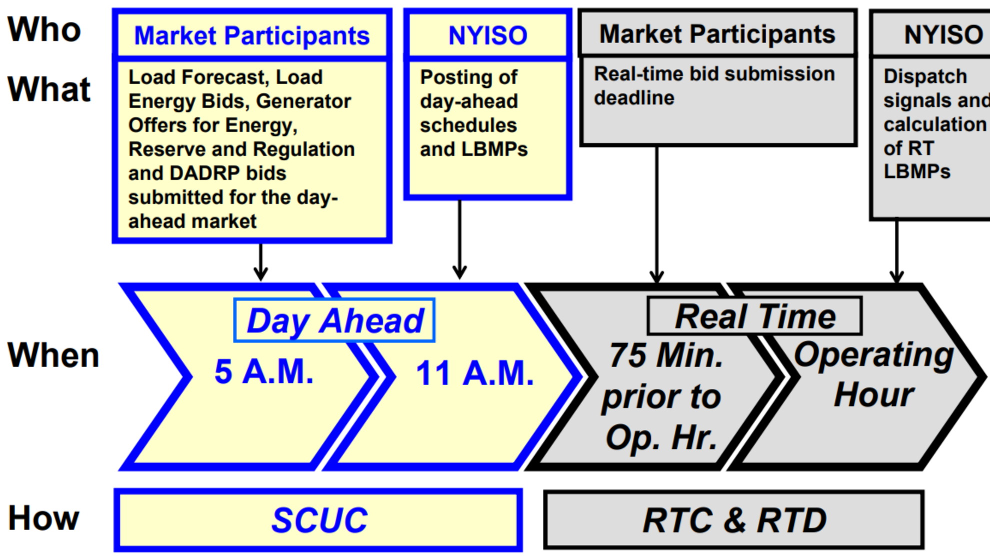

FIGURE 2.2: Energy Market Timeline. Figure from [21]

Chapter 3

Problem outline

This thesis aims at defining strategies of energy supply bidding in NYISO to provide advice to producers to reach their goals on the market. The classical economic goal of market players is increasing profits. However, calculating profit implies that costs are known, which is not the case, as there is no publicly available data about electricity suppliers' production costs. And that is why revenue was chosen as a metric for optimization in this work.

This target can be reached by proposing a strategy for suppliers to set the bidding price as high as possible, but that still be accepted by the operator.

The idea behind this was precisely formulated in [13]: "In a perfectly competitive market, a supplier would maximize his profits by bidding on his true marginal cost function. However, mainly due to the limited number of power companies, the fact that investments are capital intensive, and the constraints of the transmission system, the electricity market is more akin to an oligopoly. Therefore, market power can be exercised to ensure profits well above the ones that would be realized in a perfectly competitive market."

Due to a lack of advanced domain knowledge, and in the absence of clearly stated public information about some exceptional topic of market operations, the next assumptions are stated:

- Assumed that bids were rejected if their price level was above maximum LBMP from Real-Time Market LBMP report, provided by NYISO. And vice versa, all bids below this level were accepted;
- Assumed that all stated volume of Megawatts is accepted if the system accepts the supply bid;
- Assumed that volume of Megawatts produced by the considered supplier is not market-significant, which means that it doesn't influence the maximum price threshold;
- Assumed that all related costs, such as the cost of generator startup or cost of plugging electricity to the grid, are included in a bid, and should not be considered separately;
- Assumed that no competitor supplier possesses significant market power and can not benefit from this in any way;

Chapter 4

Related works

Electricity grid and price management are quite widely discussed topics in literature. Solutions that are used possess rule-based and classical economics approaches, as well as more advanced analytical methodologies, e.g., regressions, deep neural networks, Reinforcement Learning, etc.

As stated in [10], there are three main ways to create a bidding strategy. The first idea for strategy bidding is to estimate a market cleaning price and set the price a bit cheaper. However, at the time this article was released, there was not much historical data available to do this correctly, which was admitted by the authors. This fact, together with the absence of powerful modeling tools, made the task of precise forecasting of prices, congestion, or load levels quite complicated at that time. However, nowadays, ISOs make their operational data publicly available, and powerful modeling tools have arisen, which makes the mentioned problem more feasible. The second approach is an estimation of competitors' behaviors. The third one is the application of game theory approaches. This work presents a combination of the first two mentioned strategy creation approaches.

In [13], bidding strategies for electricity producers in a competitive marketplace were introduced. The goal was to find a way of maximizing profits for the individual company via bidding. The risk aversion of players was taken into account, according to statements of the economic theory. That updates final strategies depending if a market player is risk-neutral, risk lover, or risk-averse. The stochastic Monte Carlo method was used to define the best strategies. Only modeled data, with assumed distributions, was used. And it can be considered as a drawback, as actual data is not structured enough, may not possess any known distribution, and such a method can show inefficiency when applied empirically. Also, quite small grids, 2 and 9 load points, were modeled, which does not represent real modern systems with a much larger load node number. But still, clear bidding strategies for profit maximization were introduced.

Forecasting of electricity marginal price with Artificial Neural Networks was presented in [30]. Such exogenous data as price lags (price level during timestamp -1 and -2), electricity demand level and its lags (t, t-1, t-2), reserves capacity and its lags (t, t-1, t-2) were used as input variables for forecasting. Real data from the Australian Victorian power system was used. Results show Absolute Percentage Error (APE) between actual and predicted data differs from 0.12 to 671.02, depending on the day considered, which is quite a good result. As admitted by the author, other data can be included in the model to improve results.

Authors in [36] propose using the Reinforcement Learning algorithm Q-learning to define long-run profit-maximizing supplier bidding strategy. Only the previous 24 hours of pricing data in the Day-Ahead market was used. Also, the startup costs were taken into account. However, data for testing was created synthetically, which cannot represent the natural state of the market. A comparison with simple rule-based strategies was made in favor of the proposed method.

In more recent work, [37], a deep Q-learning algorithm is used that goals more significant profits for the trader on the German intraday market. Algorithm was trained to effectively deal with actions 'buy', 'sell', and 'hold'. Historical data was used for testing. The algorithm was proved to work better than a random agent. Also, an integrated load forecast algorithm was proven to work better than an approach based only on price data.

Quite a lot of attention was paid to the problem of arbitrage between day-ahead and real-time markets. The idea is to buy electricity on the day-ahead market cheaper and sell it more expensive on the real-time market, making a profit. It is possible because the forecast of the day-ahead market is not precise, and such trades are needed to balance the market. Although it is not the same problem that will be addressed in this work, the arbitrage solution has encapsulated electricity price prediction and data analysis, approaches that can be useful for this work.

In [2] proposed an Online learning method based on dynamic programming algorithms. It was tested on actual data from real-time and day-ahead prices provided by NYISO. Their algorithm's results were compared to rule-based ones and overperformed during most of the time slots. Later on, [33] proposed a machine learning framework, called Mixture Density Network, that beat the Online Learning method to solve the same problem. The latter's competitive advantage was using exogenous factors to model markets, as described in the following citation: "The key features that influence the price spreads between day-ahead and real-time locational market prices can be categorized into three groups. The first group includes all the meteorological variables, such as system-wide/zonal temperature, dew point, cloud cover, and wind speed. The second group includes all the relevant fuel prices for natural gas, coal, and diesel. The third group includes the system variables such as the forecast for system/zonal demand, the available generation capacity by fuel type, and transmission outages."

Another work on the topic is [34], which uses deep reinforcement learning to address the same arbitrage problem for the Netherlands market. The proposed approach (Deep Deterministic Policy Gradient) won the comparison to other strategies, some are rule-based, and some use more straightforward agents. Actual data consisted of forecasted solar and wind generation, forecasted load, actual generation from biomass, gas, nuclear, solar, waste, wind, 24-hour lagged day-ahead electricity prices, 24-hour lagged bid and ask prices, and bidding volumes.

Chapter 5

Methodology

Addressing the problem outlined above involves forecasting of LBMP. Therefore, the time series forecasting models used during this research are listed below. Their choice was dictated by their effective dealing with business time series forecasting in literature [15] [24]. Also, they use different approaches to forecasting (decision trees, deep learning, regressions), which may be beneficial to solving the discussed problem. Realization of models and experiments can be found in the next GitHub repository: `https://github.com/liubas3171/Energy_supply_bidding`

5.1 Linear Regression

As explained in [38], Linear Regression is a standard statistical method for modeling a dependent variable by finding its relationship with one or few independent variables. This model assumes a relationship, given in Eq. 5.1.

$$y = \beta_0 + \beta_1 x + \varepsilon, \tag{5.1}$$

where y is the dependent variable, x is the independent variable (also known as regressor or feature), β_0 and β_1 are coefficients, which show the dependence of variables and are estimated by the algorithm. ε represents the error term, value of y that cannot be explained by x.

β_0 and β_1 are estimated via Ordinary Least Squares (OLS) method. Its main idea is to find such β parameter that the goal function $S(\beta)$ given in Eq. 5.2 is minimized.

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2 \tag{5.2}$$

In other words, there are estimated such β_0 and β_1, that the squared error between actual y and one, modeled by x, is minimized.

Linear Regression can also be expanded to explain dependent variables with multiple regressors. This property is also useful for time series forecasting. In the latter's case, the timestamp is used as the regressor, and external variables can also be included in the model.

A few assumptions about data should be satisfied for the model to work correctly. The first one is that dependent and independent variables possess linear relationships. Second - that error term ε is normally distributed with zero mean and constant variance. Third - features are not correlated, i.e., the data has no multicollinearity.

Implementation from the Python library sklearn was used in this work.

5.2 Random Forest

Random Forest model, widely used for classification and regression problems, was introduced in [3]. It is an ensemble of decision trees, where each tree is fitted on a random subset of the training data and a random subset of the features. The final prediction is obtained as a combination of individual trees' predictions.

The theoretical background of Random Forest can be explained as follows. Given a training set of n observations with p features, the algorithm constructs a forest of decision trees. A random subset of the training data is sampled with replacement to build each tree. This method is also known as bootstrapping. A random subset of the features is also selected at each tree node to determine the best split. This process is ongoing until a stop criterion is met, such as reaching a maximum depth or having a minimum number of samples per leaf node. The goal is to create a tree with nodes with the highest reduction in the variance of the target variable possible.

The total prediction of the Random Forest is obtained by aggregating the predictions of the individual trees. For regression, the aggregation is done by taking the mean of the predicted values of all trees. The Eq. 5.3 shows the predicted value calculation for a new observation.

$$\hat{y} = \frac{1}{T}\sum_{i=1}^{T} h_i(x), \tag{5.3}$$

where, $\hat{y}$ is the predicted value, T is the number of trees in the forest, and $h_i(x)$ is the predicted value of the ith tree for the input vector x.

There are the main parameters of the model that can be tuned to improve its performance:

- The number of decision trees to be created by the model;
- The maximum depth of each tree;
- The maximum number of features to be considered by the model when determining a split;
- The minimum number of observations that are required for splitting an internal node;
- The minimum number of observations that are required to be at a leaf node;

Implementation of the Random Forest Regressor by the Python library sklearn was used in this research.

5.3 Gradient Boosting

XGBoost is a gradient-boosting decision tree model that can be applied to classification and regression tasks. Baseline and explanations were taken from [5].

The XGBoost algorithm uses an ensemble of decision trees to make predictions. It sequentially trains a series of decision trees, where each subsequent tree corrects the errors of the previous tree. The trees are added to the ensemble to minimize the loss function. To get the final prediction, the model sums up the predictions of all the trees from the ensemble. In mathematical terms, the $\hat{y}$ prediction can be shown as Eq. 5.4, where T is the number of trees and f_k is a function on feature x, defined by kth decision tree.

$$\hat{y}_i = \sum_{k=1}^{T} f_k(x_i) \tag{5.4}$$

The loss function, minimized during the model fit, can be expressed in Eq. 5.5, where l is a differentiable convex loss function that measures the difference between actual and predicted y. The classical choice of l is Mean Square Error (MSE). The objective function is optimized using gradient descent, which involves calculating the gradients of the function concerning the model parameters and updating the parameters in the direction opposite to the direction of the gradient.

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{k=1}^{T} \Omega(f_k) \tag{5.5}$$

The second term in the latter equation, $\Omega(f)$, is the regularisation term. It penalizes the complexity of functions f for the model to choose simple ones. This helps avoid overfitting and improves the model's generalization ability to new data. The regularization is controlled by hyperparameters, such as the learning rate, which determines the step size of the gradient descent algorithm.

Other parameters of the model that can be tuned to improve its performance are:

- The number of trees in the ensemble;
- The maximum depth of each tree;
- The ratio of data samples used in each tree;
- The ratio of features used in each tree.

XGBoost Regressor was implemented in the open-source Python library xgboost, and this implementation was used in this work. The library implements parallel computing, making it efficient for dealing with significant amounts of data.

5.4 CNN

A Convolutional Neural Network (CNN) is an artificial neural network usually used for image classification and recognition tasks. However, CNNs can also be applied to time series forecasting problems. The overall description of the model was inspired by [22], and the appliance of the model to time series forecasting was inspired by [23].

The input to a CNN is a two-dimensional array, such as an image or a time series data. When considering time series data, the input data is typically transformed into a two-dimensional format using a sliding window approach. The sliding window approach involves selecting a window size and sliding it along the time series to create a two-dimensional array. Creating a two-dimensional array for time series forecasting with integrated features is more straightforward. It can be a concatenation of feature series and lags of predicted variables, therefore creating a table.

Fig. 5.1 shows a typical architecture of the model. In the convolutional layer, the input array is convolved with a set of learnable filters. The filter is moved across the input sequence one step at a time, producing a feature map for each position. This process allows the CNN to identify patterns in the time series data. The convolutional layer's output is a set of feature maps passed to the fully connected layer for

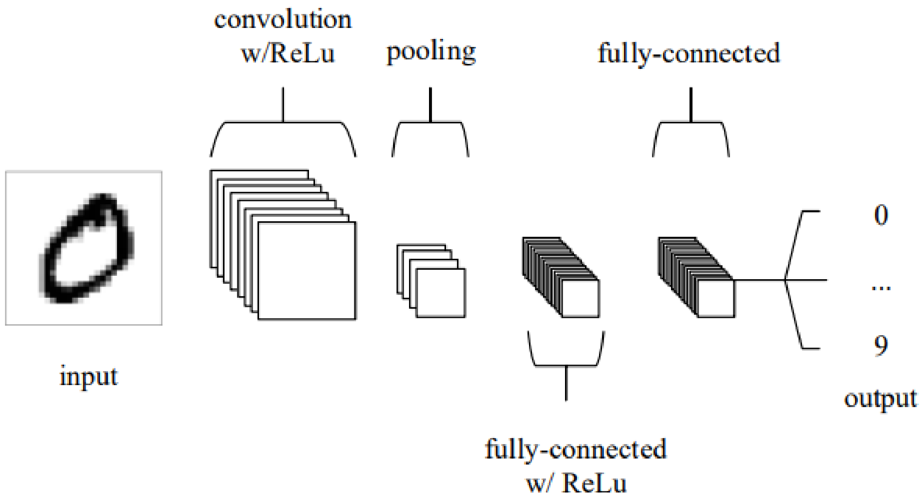

FIGURE 5.1: Typical architecture of CNN architecture. Figure from [22]

further processing. Output from the convolutional layer is compressed in size by the pooling layer.

The fully connected layer applies a set of weights to the feature maps to compute the final output. The output can be a single value, such as the predicted value of the next time step, or a set of values, such as the predicted values for the next several time steps.

By analogy with the models described above, here are the main hyperparameters of the model that can be tuned to improve its performance:

- Size of learnable filters applied;
- Dimensionality of the output space of the first fully-connected layer;
- The number of output filters in the convolution;

For this work, a network was developed using the Python library keras. Architecture is the same as provided in Fig. 5.1.

5.5 Prophet

Prophet is a time-series forecasting model presented in [31] and developed by Facebook based on an additive model that accounts for seasonality, trends, and holidays. The model consists of two main components: trend and seasonality. The trend component is modeled using a piecewise linear function, which captures the data's long-term trend and short-term fluctuations. The seasonality component is modeled using the Fourier series, which captures yearly, weekly, and daily seasonality. The Prophet model also incorporates additional regressors (features) that may affect the forecasted time series.

The mathematical formula for the Prophet model is given in Eq. 5.6.

$$y(t) = g(t) + s(t) + h(t) + e(t), \tag{5.6}$$

where, $y(t)$ is the time series value at time t, $g(t)$ is the trend component, $s(t)$ is the seasonality component, $h(t)$ is the additional regressor component, $e(t)$ is the error component at time t.

The trend component is modeled as a piecewise linear function with changepoints that allow for abrupt changes in the trend. Locations of these changepoints are found by the algorithm (involving a process similar to model regularization) or can be entered by the analyst.

The seasonality component is modeled using the Fourier series:

$$s(t) = \sum [a_n * cos(\frac{2\pi nt}{P}) + b_n * sin(\frac{2\pi nt}{P})] \tag{5.7}$$

Where n is the Fourier series order, a_n and b_n are the Fourier coefficients estimated during fitting, and P is the period of the seasonality component.

The additional regressor component is modeled as a linear function:

$$h(t) = \sum [\beta(j) * x(j,t)] \tag{5.8}$$

Where $\beta(j)$ is the regression coefficient for the j-th regressor and $x(j,t)$ is the value of the j-th regressor at time t.

The error term is assumed to have Gaussian distribution: $e(t) \sim \mathcal{N}(0, \sigma^2)$ where σ^2 is the variance of the error term.

The main parameters that can be tuned for this model are:

- Prior scale of changepoint, which determines how much the trend changes at the changepoint locations. So it should be chosen in a way that trend is neither over- nor under-fitted to changes;

- Prior scale of seasonality, which has the same meaning to seasonality as the prior scale of changepoint to trend, i.e., this scale manages flexibility of seasonality.

Implementation from the Python library prophet, developed by Facebook, was used in this research.

Chapter 6

Experiments and results

6.1 Getting familiar with data

6.1.1 Data preprocessing

The primary data we were working with in this thesis is data provided by NYISO website [20] about historical LBMPs that were paid to 595 generators. Used data describes the period from 1.01.2022 till 16.03.2023 with 3 hours intervals.

First, we removed generators where the number of nans in the series is bigger than 50%. This reduced number of considered generators from 595 to 571.

6.1.2 Outliers removal

Outliers were removed with method Inter-Quartile Range (IQR) [32]. It works as follows: first, the IQR value is calculated by the next formula:

$$IQR = Q3 - Q1 \tag{6.1}$$

Where $Q3$ is the 75th percentile of data distribution, and $Q1$ is the 25th percentile. Then, data points less than the lower bound ($Q1 - 1.5 \times IQR$) or bigger than the upper bound ($Q3 + 1.5 \times IQR$) are deleted from the dataset. This method was applied to each generator's data. Then, deleted values were replaced with the last present values in a previous timestamp. As it is better not to have empty values in analyzed data. This outlier removal method is well suited for skewed data, and that is because it was chosen. And the distribution of LBMPs is indeed skewed, as can be seen from Fig. 6.1, where the distribution of all generator's data is shown.

6.2 Known distributions assumption

The original idea was to find a known distribution that timestamps of competitors' LBMPs possess and forecast the parameters of such distributions for the future. Forecasting should be made with mentioned models and integrated exogenous factors that may affect LBMP. Then, the idea was for the supplier to bid prices within some percentiles of those distributions. For example, if a supplier bids 65th percentile, he will win more trades but sell electricity for a smaller price, so he will not earn extra money. However, if he bids 95th percentile, he will earn more money for each Megawatt sold, but a common situation may occur when his bids will not be accepted. So final profit/revenue can be lost. And so, the idea was to find which percentile of that forecasted distribution is the best to choose for the bidder.

The advantage of such an approach is that it requires only 2-3 models to train and validate. As in most known distributions, there are only a few parameters to predict.

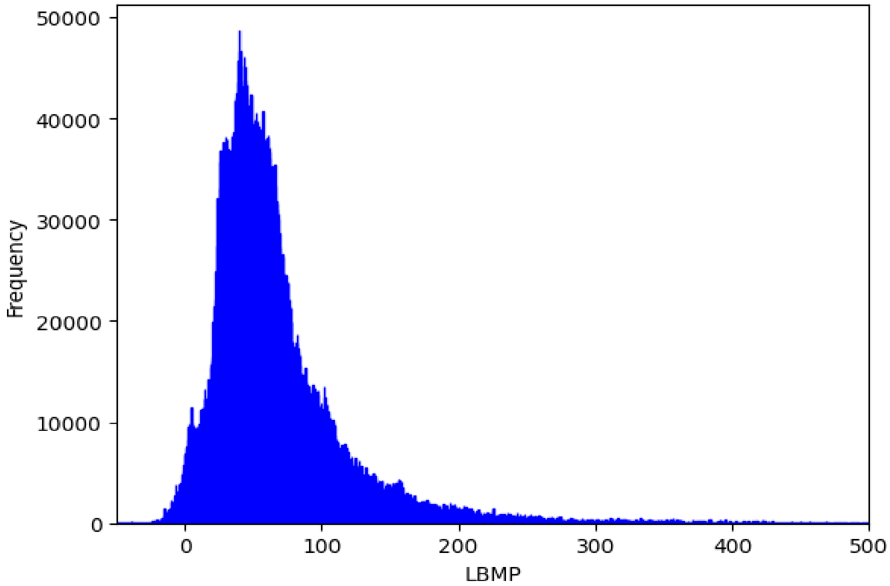

FIGURE 6.1: Distribution of overall LBMPs from the dataset

Means no need for significant computation power. And it shows the straightforward strategy for the bidder.

The main disadvantage, and why this approach was rejected, is that data does not follow any known distributions. First of all, it was decided to picture some distributions. For this, random time stamps among the recent ones were chosen. Results are shown in Fig. 6.2. So, visually, there are no classical distributions among the pictured there. And distributions among different timestamps are not even similar.

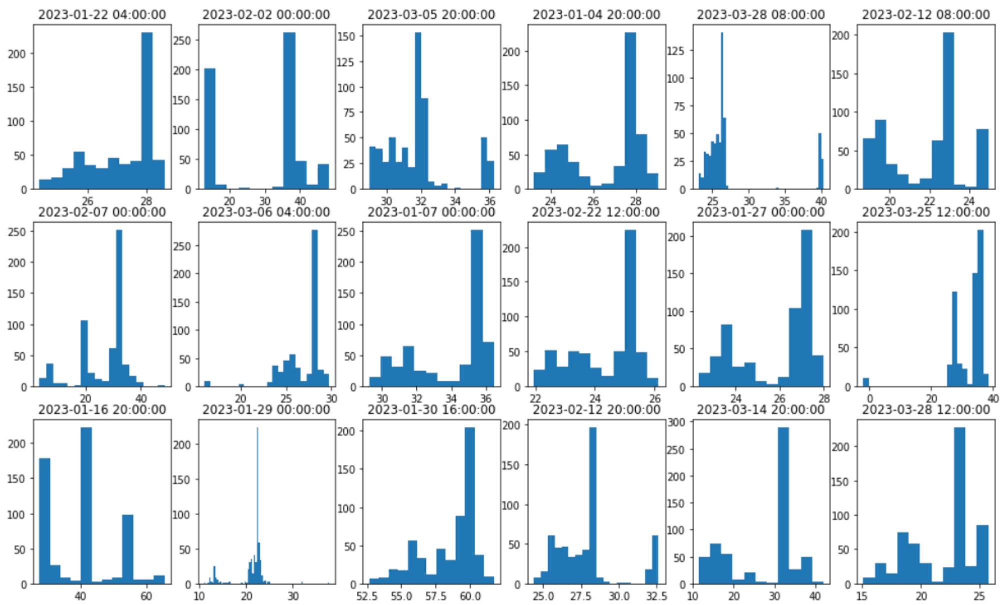

FIGURE 6.2: Distribution of LBMP data from random timestamps

6.3 Modeling competitor's behavior

And that brings us to another method - modeling competitor's behavior. The idea is to create models that predict the bid of each competitor, Then, similarly to as proposed in the method above, to make tests by choosing percentiles of the distribution of those predicted bids. We should come to a solution where the supplier can have a strategy of choosing some percentile.

The disadvantage is, of course, in computation and memory requirements. The market is competitive, and there is a significant number of generators (about 600), and creating a model per each supplier is a complicated task.

6.3.1 Features selection

We want to create a unique model per generator to forecast its behavior. But, as there are 571 time series that need to be forecasted, it is not feasible to do feature engineering per each generator. However, it can be shown that all series are similar, and if we correctly choose features for one of them, they will suit most of the generators. Of course, each has some specifications, and the unique model ideally should be able to capture such things. The Pearson correlation coefficient should be introduced to show that the generator's data is similar. It is calculated for two samples by the next formula:

$$r_{xy} = \frac{\sum\limits_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum\limits_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum\limits_{i=1}^{n}(y_i - \bar{y})^2}} \tag{6.2}$$

Here r_{xy} is the Pearson correlation coefficient between samples x and y, n is the number of observations, x_i and y_i are the i-th values of variables x and y, and $\bar{x}$ and $\bar{y}$ are the sample means of variables x and y, respectively. The coefficient can lay in the range [-1; 1] and measures the two variables' linear relationship.

And, returning to LBMPs of generators, we can calculate their pairwise correlations. Then, calculate the average correlation of every generator data with other ones. After this, we can average the latter values. After such a process, we found that the average correlation of LBMPS of different generators among all series equals 0.74, which is considered high. After this, we found a generator whose average correlation with other ones is close to this value and is equal to 0.69. Therefore, if we take LBMP time series of this unit (generator) as a dependent variable for our feature selection procedure, chosen features should also suit most of the other generators from the dataset.

Based on examples from literature and common sense, feature candidates were chosen. They may be divided into three main categories: energy prices, weather data, ISO operational data. The overall description is provided in Tab. 6.1.

An analysis was conducted to decide if mentioned data should be included as features in models. First, the correlation between these variables and chosen generator's LBMP was calculated, and the correlation among features itself. Results are shown in Fig. 6.3. Although some features are well correlated with LBMP, they are also highly correlated with each other. This issue is known as multicollinearity and may cause unstable models' predictions and problems with interpretations [35]. Therefore, we cannot include correlated variables as features in our models, e.g.,

Name	Description	Source
Load	Historical data about actual electricity consumption	NYISO website
Natural Gas	Historical data about gas prices in USA	Nasdaq website
Coal	Historical data about coal prices in USA	Markets Insider website
Crude Oil	Historical data about crude oil prices in USA	Markets Insider website
Uranium	Historical data about uranium prices in USA	Investing.com
Temperature	Historical data about temperature, measured by a certain station in New York State	Meteostat Python API
Precip	Historical data about precipitation level, measured by a certain station in New York State	Meteostat Python API
Pressure	Historical data about atmospheric pressure, measured by a certain station in New York State	Meteostat Python API
Wind speed	Historical data about wind speed, measured by a certain station in New York State	Meteostat Python API
Dew temperature	Historical data about dew temperature, measured by a certain station in New York State	Meteostat Python API
Relative humidity	Historical data about relative humidity level, measured by a certain station in New York State	Meteostat Python API
Gen Scheduled	Description of variable 2	NYISO website
Wheel Throughs Bid	Historical data about the amount of electricity, bought by retailers in a system	NYISO website
Imports NYISO	Historical data about the amount of electricity imported by NYISO from other states	NYISO website
Exports HQ	Historical data about exports of energy by other state's ISO, namely Hydro-Québec	NYISO website
Exports PJM	Historical data about exports of energy by other state's ISO, namely PJM Interconnection	NYISO website
Is holiday	Indicator whether there is a holiday in the USA on the given day	holidays Python library

TABLE 6.1: Features candidates description

Load and Gen Scheduled, cannot be included simultaneously, as their correlation is significantly high.

Also, such metrics as mutual information score was calculated for each pair of variables in a manner similar to the above. The mutual information score measures the level of nonlinear dependence between two random variables. It takes zero value if and only if two random variables are independent [24]. The higher the score value - the stronger dependence between variables. The score calculation is given in Eq. 6.3, where $p(x,y)$ is the joint probability distribution of X and Y, and $p(x)$ and $p(y)$ are the marginal probability distributions [6].

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) \tag{6.3}$$

Calculation results for considered variables are presented in Fig. 6.4. It shows a similar situation of relationships to the one provided by the correlation matrix.

Another method to decide on feature selection is permutation feature importance [3]. The idea is to train the model with all features, where one of the features was randomly permuted, while the rest were untouched. Then one can see how this will affect performance metrics, MAPE in our case. Therefore, if performance metrics worsen after some variable's permutation, this variable significantly impacts predicted values and should be chosen as an important feature. Prophet was used as a basic model for this method.

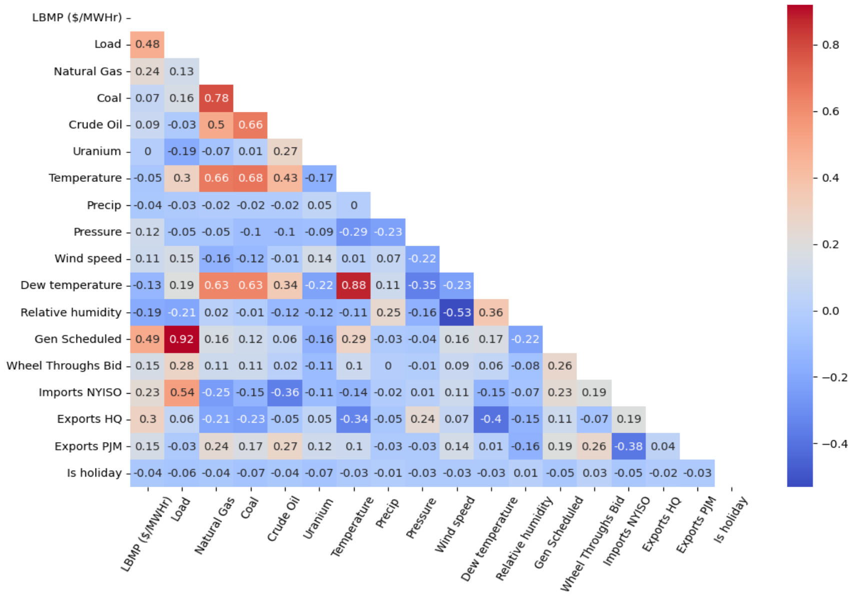

FIGURE 6.3: Pearson correlation matrix of features

Results are provided in Fig. 6.5. Here, among others, is presented the value named BASELINE. It corresponds to the metrics of the model, trained without any variable permuted.

Based on the results of the three methods described above, the following features were chosen for modeling the final prediction: Load, Natural Gas, Dew temperature, and Relative humidity. The load has a high correlation with the goal variable and is important from the permutation method's point of view. Load is also highly correlated with Gen Scheduled and Imports NYISO, and, therefore, the latter variables were not included. Natural Gas is included because it is important from all three perspectives. It is highly correlated with other energy prices, so it was decided not to include them. Dew temperature was chosen because it possesses good mutual info and permutation score results. The temperature indicator was excluded because of the high correlation with the latter feature. The final included variable is Relative humidity, which has good correlation and permutation scores. The rest, not mentioned variables were not included because of their weak relationships with predicted data.

As we work with time series, not only the current value of features may be useful for forecasting the dependent variable, but also values of features on the previous timestamps, called lags [30]. To check this influence, improvement to Partial Autocorrelation Function (PACF) was used, as described in [24]. It shows the effect of lag i of the feature variable on the improvement of the prediction score if we use all previous lags of this variable. To calculate this, we fit the Linear Regression model with $i-1$ lags of the certain feature, where Y is LBMP. Then, to find PACF at ith lag, we calculate a Pearson correlation of this lag with residuals of fitted regression. This metric was calculated for 100 lags per each feature, and results are shown in Fig. 6.6. From here, one can see that lag zero of each feature is significantly more important than all others. Therefore, only zero lag was used for forecasting.

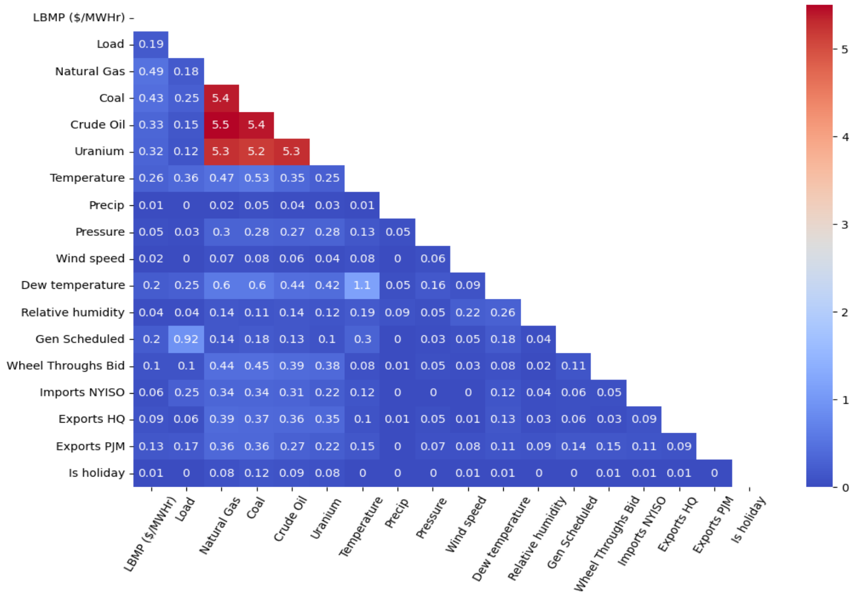

FIGURE 6.4: Mutual info score between features

6.3.2 Features forecasting

We should possess information about the future values of variables we chose as features. There are various ways to receive such information. The first is to forecast these values independently and without exogenous factors. And such problems can be considered as topics for other research works. But there are workarounds. If we take weather indicators, indeed, it is a complicated task that should involve teams of researchers, stations, etc. But luckily, these people share such information for free. Library Meteostat provides API to access forecasts for the next 10 days, which is enough for our task.

Considering the Load forecast, it is quite a simple task, as data possesses strong daily seasonality. Therefore, catching such dependency for a good model is relatively straightforward. And indeed, FB Prophet achieved good results in Mean Absolute Percentage Error (MAPE) 4.37 on the validation set. MAPE metric is introduced below, in section 6.3.3. The model was trained on a set that consisted of 80% of data and validated on a range [80, 97.5]. The parameters described above were validated during this process.

Another situation is with Gas price prediction, as its values have been very complicated recently, and it does not possess any seen seasonality. But still, FB Prophet achieved MAPE on the test result, which is quite good, as for me, and is equal to 51.17.

Results can be seen in Fig. 6.7. The Load is well predicted, both visually and at performance metrics. In contrast, the Gas price is not so great, but we still can use it.

6.3.3 Training

Performance metrics

As the main performance metrics of all trained models, MAPE was used. It is calculated in the next way:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{6.4}$$

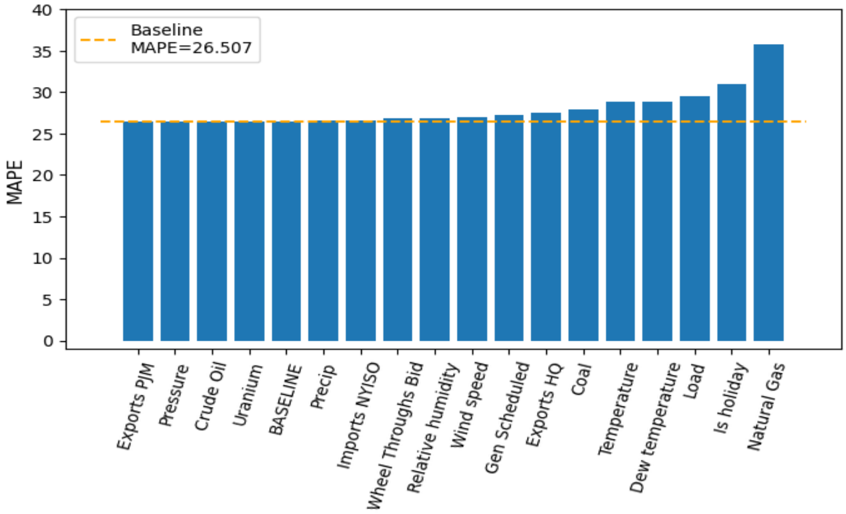

FIGURE 6.5: Permutation feature importance

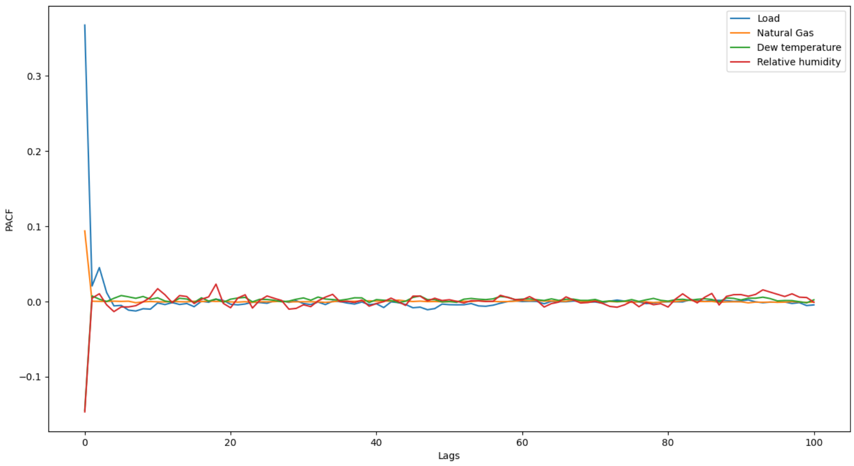

FIGURE 6.6: Lags importance with PACF

Where n is the number of observations, y_i is the actual value of the i-th observation, and $\hat{y}_i$ is the forecasted value of the i-th observation.

Walk-Forward validation

For validating hyperparameters of models and avoiding overfitting, commonly used Walk-Forward validation was applied [26]. Its main idea is to split data into n folds with commonly exclusive validation sets and train the model on all data, located before the validation. As we have the problem of time series forecasting, it makes sense to split and train data with regard to timestamps. n was set to 3 in our experiments. This process can be seen in figure 6.8. At each train-validation step, MAPE is calculated. Then, the average of these values is taken. The less average MAPE we receive - the best model parameters were chosen.

This validation was processed on 97.5% of the data. The rest 2.5% were left for testing purposes so that models do not see this data. The latter percentage was chosen because it corresponds to 10 days. 10 days - it is the time when systems provide relaying weather forecasts. Also, 10 days sound like a reasonable amount of time to plan work with the help of a model. More on the test part of the data - later.

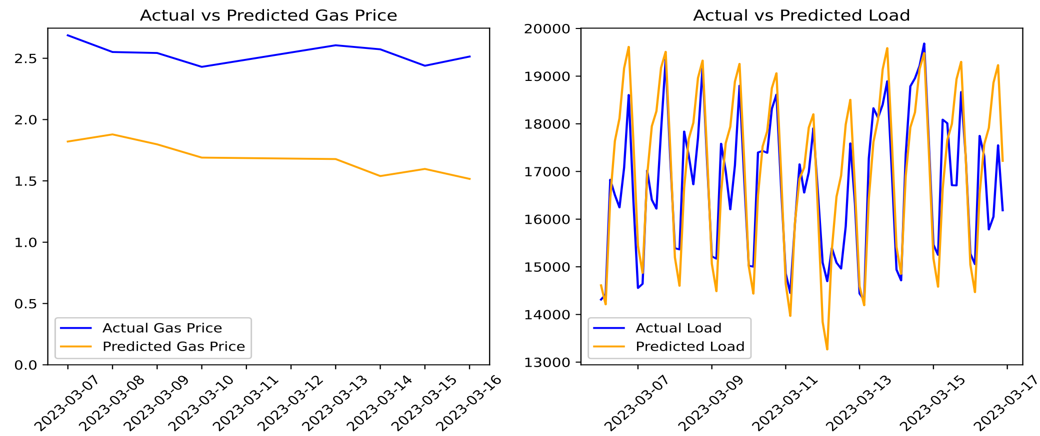

FIGURE 6.7: Forecast of gas prices and load

Parameter sets are created as permutations of all parameters. Then model with each parameter set is created, and Walk-Forward validation is conducted. Each of the 5 models with the best parameters per generator is received as a result.

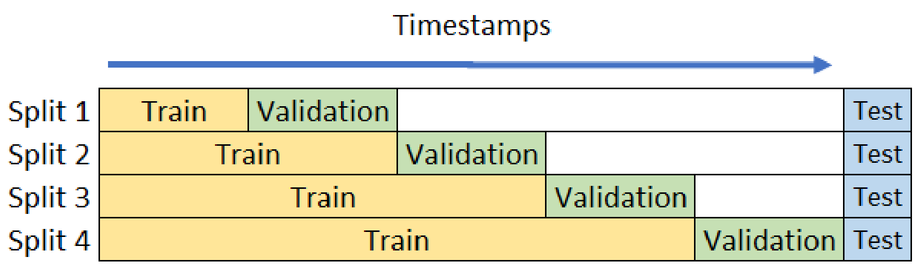

FIGURE 6.8: Walk-Forward validation process

Selection of the best models

Later, 5 models per generator with the best parameters mentioned were trained on 80% of the data. Performance metric was calculated on the validation set of data, which belongs to the range [80%, 97.5%]. A model with the best metrics was chosen to model the generator's behavior and was fitted with merged train and validation data. The names and numbers of models that were chosen as the best are presented in Tab. 6.2.

TABLE 6.2: Best models summary

Linear Regression	**Random Forest**	**Gradient Boosting**	**CNN**	**Prophet**
3	0	11	68	489

The distribution of resulting performance metrics on a validation set of all 571 models is given on Tab. 6.3, where 25%, 50%, and 75% are percentiles of the distribution.

Visualizations of forecasting on the test dataset are given in Fig. 6.9. For visualization, generators were chosen with MAPE values corresponding to those presented in Tab. 6.3. As can be seen, models are good at capturing a daily seasonality. And it must be the reason why Prophet was chosen as the best model so frequently, as shown in Tab. 6.2. That is the only model considered which has seasonality as an

TABLE 6.3: Distribution of MAPEs of final models

Min	25%	50%	75%	Max	Mean
24.70	30.09	36.07	66.73	1965.97	85.75

explicit factor. But models do not deal well with spikes and noise in data, and that causes performance results that are not the best, as one can see from Tab. 6.3.

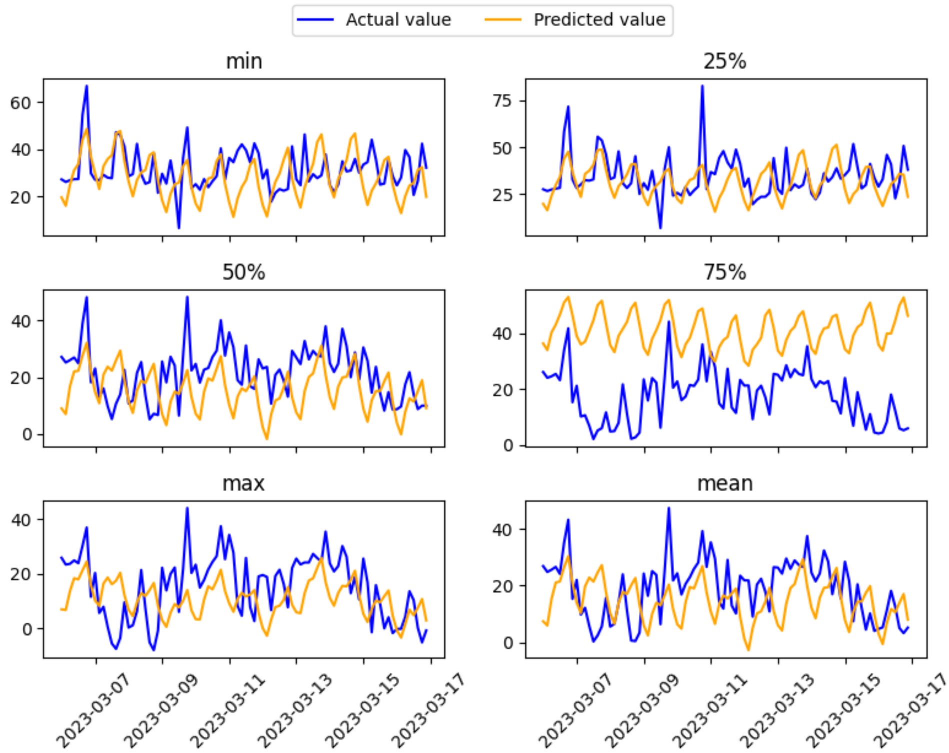

FIGURE 6.9: Forecasting examples on test data

6.4 Resuls

After the final models for each generator were defined, they forecasted bids for the last 10 days (2.5% of the dataset). To define an optimal bidding strategy for the generator, the next method was used. Among forecasted data, we chose certain percentiles from the distribution of forecasted values for each timestamp. Then, if the chosen value was bigger than the maximal real bid from the historical test data, the bid was rejected. This means that money for this bid was not earned. But if the chosen value was less than the maximal value, it was accepted, and the generator received the requested rate per MWHr. Results of such simulation are provided in Fig. 6.10, where earnings are provided as a function of the chosen percentile. From here can be seen, reasonably expected, that maximum earnings (2505 \$/MWHr) are achieved when one chooses a high percentile but not the maximum one, namely, 95th.

After defining the strategy, it was compared to the real earnings of generators for the considered time. For this, the total earning per MWHr of each generator was calculated. And we found out, that, on average generator earned 2611 \$/MWHr, which is more than in the defined strategy. In Fig. 6.11, one can see earnings with

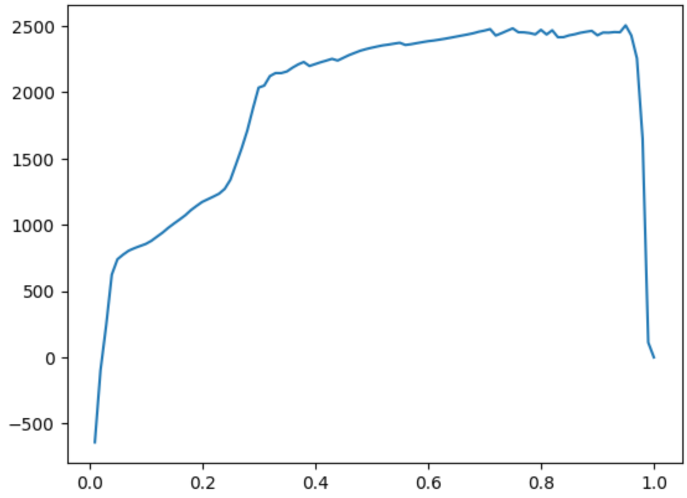

FIGURE 6.10: Earnings as a function of chosen percentile

the proposed strategy, compared with the distribution of actual earnings by the generator. And there can be seen that this strategy overperforms a significant part of real market players.

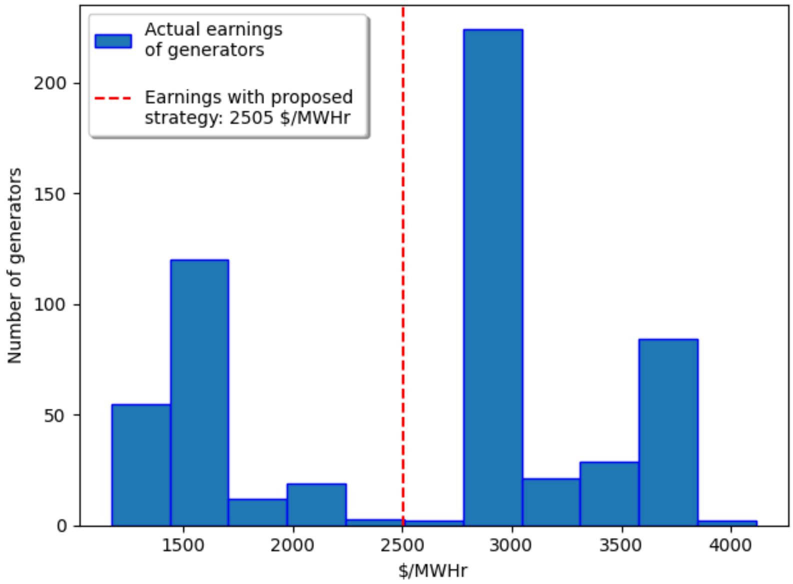

FIGURE 6.11: Distribution of actual earnings by generators

Results are ambiguous. On the one hand, the proposed method does not significantly improve the earnings of the average market participant. Also, training and validating all models is expensive and requires much computational power. On the other hand, despite limited domain knowledge and domain data used, this approach helps receive earnings bigger than the number of the actual market participants that possess mentioned information.

The main problem of the weak performance of considered forecasting models is volatile and spiky data, as shown in Fig. 6.9. Therefore, future works can address it by using models developed for such purposes or using internal market data that is not publicly available.

Chapter 7

Conclusion

In this thesis, deregulated electricity markets generally, and NYISO in particular, were introduced. The problem of optimal electricity supply bidding was addressed and turned into time series forecasting problem. To solve the latter, machine learning and deep learning models were introduced. Feature selection process was conducted, as well as models training and validation. Finally, a strategy for using selected models for effective supply bidding was proposed, and its pros and cons were discussed.

Bibliography

[1] *About MISO*. https://www.misoenergy.org/. Accessed on 10 May 2023. 2023.

[2] Sevi Baltaoglu, Lang Tong, and Qing Zhao. "Online Learning of Optimal Bidding Strategy in Repeated Multi-Commodity Auctions". In: *CoRR* abs/1703.02567 (2017). arXiv: 1703.02567. URL: http://arxiv.org/abs/1703.02567.

[3] Leo Breiman. "Random Forests". In: *Mach. Learn.* 45.1 (2001), pp. 5–32. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324.

[4] *California Independent System Operator - about us*. https://www.caiso.com/about/Pages/default.aspx. Accessed on 10 May 2023. 2023.

[5] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: Association for Computing Machinery, 2016, 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. URL: https://doi.org/10.1145/2939672.2939785.

[6] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. USA: Wiley-Interscience, 2006. ISBN: 0471241954.

[7] Gina E. Craan. *Intermediate Level LBMP Recap*. 2022. URL: https://www.nyiso.com/documents/20142/25467833/LBMP-Intermediate-ReCap.pdf/2490e2f0-54e0-7fe7-ac7e-9b03db3e25f1.

[8] Gina E. Craan. *Loss Price Component*. 2022. URL: https://www.nyiso.com/documents/20142/25467833/LBMP-Loss-Price-Component.pdf/d882794e-619a-2181-d367-475ab0fdf897.

[9] Gina E. Craan. *Market Overview Introduction*. 2020. URL: https://www.nyiso.com/documents/20142/3036629/NYISO-Market-Overview.pdf/45b5844c-02d0-4de3-be5b-abae979246f1.

[10] A.K. David and Fushuan Wen. "Strategic bidding in competitive electricity markets: a literature survey". In: *2000 Power Engineering Society Summer Meeting (Cat. No.00CH37134)*. Vol. 4. 2000, 2168–2173 vol. 4. DOI: 10.1109/PESS.2000.866982.

[11] Fresh Energy. *Negative prices in the MISO market: what's happening and why should we care?* 2018. URL: https://fresh-energy.org/negative-prices-in-the-miso-market-whats-happening-and-why-should-we-care.

[12] Jean-Michel Glachant, Paul L. Joskow, and Michael G. Pollitt. *Handbook on Electricity Markets*. Cheltenham, UK: Edward Elgar Publishing, 2021. ISBN: 9781788979948. DOI: 10.4337/9781788979955. URL: https://www.elgaronline.com/view/edcoll/9781788979948/9781788979948.xml.

[13] V.P. Gountis and A.G. Bakirtzis. "Bidding strategies for electricity producers in a competitive electricity marketplace". In: *IEEE Transactions on Power Systems* 19.1 (2004), pp. 356–365. DOI: 10.1109/TPWRS.2003.821474.

[14] William W. Hogan. "INDEPENDENT SYSTEM OPERATOR : PRICING AND FLEXIBILITY IN A COMPETITIVE ELECTRICITY MARKET". In: 1998.

[15] Journal Ijmer, Ms. Neelam, and Abhinav Jain Er. "Time Series Data Analysis for Forecasting - A Literature Review". In: 2014.

[16] Alexander J. M. Kell, Stephen McGough, and Matthew Forshaw. *Machine learning applications for electricity market agent-based models: A systematic literature review*. 2022. arXiv: 2206.02196 [cs.MA].

[17] Mathangi Srinivasan Kumar. *Locational Based Marginal Pricing*. 2023. URL: https://www.nyiso.com/documents/20142/3037451/3-LMBP.pdf/f7682e03-e921-eaab-09bf-690524b5ade6.

[18] *National Grid - about us*. https://www.nationalgrid.com/about-us. Accessed on 10 May 2023. 2023.

[19] *Nord Pool - about us*. https://www.nordpoolgroup.com/en/About-us/. Accessed on 10 May 2023. 2023.

[20] *NYISO Web Site*. 2023. URL: https://www.nyiso.com.

[21] New York Independent System Operator. *Locational Based Marginal Pricing Infographic*. 2021. URL: https://www.nyiso.com/documents/20142/34827341/LBMP-Infographic.pdf/0d1a9d06-2d3c-42c8-f005-a3c260388de4.

[22] Keiron O'Shea and Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. arXiv: 1511.08458 [cs.NE].

[23] Avinash Kumar Pandey. *Hands-On Stock Price Time Series Forecasting using Deep Convolutional Networks*. https://www.analyticsvidhya.com/blog/2021/08/hands-on-stock-price-time-series-forecasting-using-deep-convolutional-networks/. Accessed: 14.05.2023. 2021.

[24] Andrii Prysiazhnyk. "Dynamic Pricing using Reinforcement Learning for the Amazon marketplace". Ukrainian Catholic University, Department of Computer Sciences, Lviv, 2021, 54p. URL: https://er.ucu.edu.ua/handle/1/2857.

[25] H. Rudnick. "Chile: Pioneer in deregulation of the electric power sector". In: *IEEE Power Engineering Review* 14.6 (1994), pp. 28–. DOI: 10.1109/MPER.1994.286546.

[26] Venishetty Sai Vineeth, Huseyin Kusetogullari, and Alain Boone. "Forecasting Sales of Truck Components: A Machine Learning Approach". In: *2020 IEEE 10th International Conference on Intelligent Systems (IS)*. 2020, pp. 510–516. DOI: 10.1109/IS48319.2020.9200128.

[27] Zach T. Smith. *Congestion Price Component*. 2022. URL: https://www.nyiso.com/documents/20142/25467833/LBMP-Congestion-Price-Component.pdf/6a9c8078-5eea-c717-9099-c0a045ea86ce.

[28] Kelly Stegmann. *Energy Price Component*. 2022. URL: https://www.nyiso.com/documents/20142/25467833/LBMP-Energy-Price-Component.pdf/bc8b62b7-4eb0-3c37-9c60-62141462a451.

[29] Kelly Stegmann. *NYISO Energy Marketplace*. 2023. URL: https://www.nyiso.com/documents/20142/3037451/2-Energy-Marketplace.pdf/5f7d9870-a07f-bb21-19b1-ffb34ded64c7.

[30] B.R. Szkuta, L.A. Sanabria, and T.S. Dillon. "Electricity price short-term forecasting using artificial neural networks". In: *IEEE Transactions on Power Systems* 14.3 (1999), pp. 851–857. DOI: 10.1109/59.780895.

[31] Letham B. Taylor SJ. "Forecasting at scale". In: *PeerJ Preprints 5:e3190v2* (2017). DOI: https://doi.org/10.7287/peerj.preprints.3190v2.

[32] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[33] Wei Wang and Nanpeng Yu. "A Machine Learning Framework for Algorithmic Trading with Virtual Bids in Electricity Markets". In: *2019 IEEE Power Energy Society General Meeting (PESGM)*. 2019, pp. 1–5. DOI: 10.1109/PESGM40551.2019.8973750.

[34] Yuanrong Wang et al. *Deep Reinforcement Learning for Power Trading*. 2023. arXiv: 2301.08360 [q-fin.TR].

[35] Jeffrey Marc Wooldridge. *Introductory Econometrics: A Modern Approach*. ISE - International Student Edition. South-Western, 2009. ISBN: 9780324581621. URL: http://books.google.ch/books?id=64vt5TDBNLwC.

[36] Gaofeng Xiong, T. Hashiyama, and S. Okuma. "An electricity supplier bidding strategy through Q-Learning". In: *IEEE Power Engineering Society Summer Meeting*, vol. 3. 2002, 1516–1521 vol.3. DOI: 10.1109/PESS.2002.1043645.

[37] Zhening Xu and Georg Ritter. "Deep Reinforcement Learning for Intra-Day electricity market trading". In: *Marex Institute* (2021). URL: https://marex-website-media-content.s3.amazonaws.com/uploads/2021/09/Deep-Reinforcement-Learning-Electricity-Market-Trading-final-September-2021.pdf.

[38] Xin Yan and Xiao Gang Su. *Linear Regression Analysis: Theory and Computing*. USA: World Scientific Publishing Co., Inc., 2009. ISBN: 9789812834102.