Delft University of Technology
MSc Sustainable Energy technology

Profit optimized machine learning aided electricity storage; Comparison between privately owned and neighborhood shared

An auto-machine learning solution

Kevin Dankers - 4543416

June 27, 2022

Contact information

Student:
Name: Kevin Dankers
Email k.e.dankers@student.tudelft.nl
Studentnumber 4543416

Supervisor:
Name: Simon Tindemans
Email S.H.tindemans@tudelft.nl

Abstract

Batteries are an essential tool for energy transition. They provide the ability to reduce imbalances by absorbing the forecast errors for consumers and renewable energy sources. This has the added benefit that congestions occurring due to supply and demand mismatches would be prevented. Furthermore, they allow generating a stable power consumption for a longer period by utilizing the available reserves. The electricity markets are all power markets, where the products grant an amount of power for a certain period. Therefore, the introduction of a battery would make it easier to buy power for a portfolio of consumers. Currently, it is not yet feasible to use a battery for the consumer segment as the revenues generated are simply too small to pay the battery back before it reaches the end of its lifetime.

This report aims to describe a machine learning-based battery operating algorithm that can be utilized for the consumer market. The operating algorithm will be trading on the day ahead and the voluntary aFRR market while satisfying the demand of the consumer(s). The trading is done based on the forecast generated by different machine learning models. For the day ahead market, two forecasting models are tested, namely an XGBoost algorithm and a Dense Neural Network. For the imbalance market, a simple Dense Neural Network and a Long Short-Term Memory (LSTM) are compared and for the consumer forecast, an XGBoost model is compared to a Lasso. All forecasts are combined to create a profit-maximizing algorithm. Lastly, a comparison is made between having an individual battery for every homeowner and aggregating a group of consumers together in a larger battery is compared.

The results show that forecasting an aggregated group of consumers has a significantly lower error compared to forecasting individual demand patterns. The large errors in the individual forecast result in a large opportunity cost, based on the Value of Lost Load (VoLL), which renders the benefits of owning a battery useless again. However, the aggregated group of consumers combined into a single large battery is profitable. Considering the current prices (22Q1) the battery generated enormous profits, such that the battery would be break-even in less than 2 years.

Table of Contents

1 Introduction

Electricity grids are starting to reach their capacity limits, as the use of electricity as a power source is growing. Both heat and transportation are opting for electricity as their main source of energy. Together with the increase in the production of renewable energy, more grid congestion starts to appear. In the Netherlands, in roughly 40% of the country, the electricity grid has or nearly has reached its maximum consumption capacity and does not allow for new connections. Furthermore, the production capacity limit has even been reached in roughly 90% of the country (Netbeheer Nederland, 2022). The lack of available grid capacity is determined by the power demand during peak moments, while there is still spare capacity during off-peak moments. The use of home batteries can be helpful because they can absorb part of the peak demand of the consumers as suggested by Cappellen et al. (2022). However, Cappellen et al. (2022) argue that such home batteries would require large amounts of subsidies since they would not be profitable in and of themselves. But allowing the consumer to trade on the electricity market, using their home battery, could potentially increase the profitability of the battery, and, thus, mitigating the need for subsidies. Therefore, the objective of this research is to design auto-machine learning algorithms and a profit-maximizing battery operating algorithm, that can trade on different Dutch electricity markets while still satisfying the consumer's demand, in order to assess whether the use of home batteries is feasible without government subsidies.

1.1. Research Context

A literature review by Baumgarte et al. (2020) shows that the profitability of batteries in energy markets has been researched quite extensively by numerous authors. For example, Hugenholtz (2020) does a techno-economic analysis of electricity arbitrage in the Netherlands and concludes that Lithium-Ion batteries are not profitable when trading for price arbitrage only on the day-ahead market nor on the secondary reserve, also known as the automatic Frequency Restoration Reserve (aFRR). The author concludes that trading on the aFRR is significantly more profitable than trading on the Day-Ahead markets due to the larger price deviations seen on the first. The author greatly simplifies reality by assuming that a battery is fully activated during a Program Time Unit (PTU). In the Netherlands, a PTU equals 15 minutes. However, placed bids are often not fully activated during a PTU, but only partially. Furthermore, the author assumes that you can place bids directly for the next PTU, which does not hold in the case of the aFRR. However, the author does not specify exactly which market is considered in the research, but the focus is likely on the voluntary aFRR. Lastly, the research uses a simplified forecasting method. This thesis covers this gap by addressing those simplifications, and most important, using actual forecasting methods to facilitate the decision-making process.

Similar results are found by Metz and Saraiva (2018) for the German intraday market, which is closely related to the Dutch imbalance market. The authors find that, for single-purpose storage, the prices need to increase 7-fold for a BESS to become profitable. The conclusion that a single-purpose battery is not profitable for

trading in Germany is also shown by Englberger et al. (2020). But Englberger et al. (2020) and Baumgarte et al. (2020) both conclude that when business cases are stacked, operating a battery can become profitable. Baumgarte et al. (2020) show several combinations that have been made profitable, but the combination of self-sufficiency and trading arbitrage has not yet been proven profitable.

Combining a consumer and an imbalance market has been covered by Engels et al. (2017), who combined a home battery for self-consumption and providing balancing services on the primary reserve, also known as the Frequency Containment Reserve (FCR). Engels et al. (2017) use stochastic optimization to assign battery capacity to self-consumption or FCR. The follow-up research by Engels et al. (2019) extends the previous research by incorporating the degradation into the model as well. Engels et al. (2019) find that providing FCR services can be profitable. However, due to this profitability, the FCR market will likely be overrun with batteries providing such service (R. Jansen, personal communication, 14 March 2022).

This research extends the literature on the stacking of battery purposes, by combining the day-ahead market, the consumer(s), and the aFRR, to determine the profitability of home batteries. This is done by using predictive algorithms to optimize the available storage while satisfying the demand of the consumer. However, to buy the required amount of energy for a consumer, it is necessary to know how much the consumer will demand, and to determine the most profitable moments to buy and sell power on the day-ahead market requires knowledge about how prices will develop. Machine learning algorithms have proven themselves remarkably good for forecasting both, such as shown by Kim and Cho (2019) for demand forecasting and Lago et al. (2018) for day-ahead price forecasting.

1.1.1. Research Questions

The main research question that will be answered with this research is as follows:

> How can a household battery in the Netherlands be utilized simultaneously in different markets such that it generates the largest profit?

The sub-questions answered in this research are as following:

1. How profitable is it to operate the battery model only on the day-ahead market under perfect foresight conditions?
2. How can auto-machine-learning be implemented in the forecasting model, to prevent the model from losing its predictive accuracy over time?
3. How do the profits from a naive aFRR bidding strategy compare to a forecast-aided bidding strategy?
4. How do the profits from the final model differ between a single and a group of consumers?
5. How long does it take to repay the capital expenditure of a Tesla Powerwall 2 considering joint bidding on different power markets?

1.2. General approach

This study uses several machine learning models for forecasting and compares their predictive power. For the day-ahead market, an XGBoost and a Dense Neural Network are compared. For the imbalance market, a simple Dense Neural Network and a Long Short-Term Memory (LSTM) are compared. And for the consumer forecast, an XGBoost model is compared to a Lasso. Finally, all forecasts are combined to create a profit-maximizing algorithm.

Many available features could carry explanatory value for the day-ahead and aFRR markets because power grids of countries are more and more connected to each other, and, thus, influence each other's electricity

markets (Van Der Heijden et al., 2021). Furthermore, there is a strong autocorrelation in both the forecasted and the explanatory variables, which asks for the inclusion of their lags in the models, further increasing the number of available features. To filter out the redundant variables, multiple Relief-Based Algorithms (RBAs) are tested for their ability to select the feature set that results in the most accurate predictions.

Furthermore, a model used in the real world is only useful if it can keep performing over time. It is unlikely that a forecasting model would be able to deal with large changes such as the COVID-19 pandemic or the price spike in electricity prices seen at the end of 2021. This non-stationarity is called concept drift in machine learning, and the forecast model needs to be able to deal with this. To do so, the ADWIN algorithm developed by Bifet and Gavalda (2007) is used to monitor the model performance, which will degrade due to concept drift.

Unfortunately, most changes in correlations between the dependent variable and explanatory variables happen gradually over time in the case of electricity prices. This is problematic for drift detection because ADWIN, and most other drift detection algorithms, have difficulty detecting gradual drift. To mitigate this issue in forecasting day-ahead power prices, I propose a simple bias correction algorithm that shifts the predictions based on the historically observed bias. The bias correction aids the model in obtaining a lower forecasting error but does not change the daily pattern.

After obtaining the forecasts, a battery operating algorithm utilizes the predictions to maximize profits. In the battery operating algorithm proposed in this research, the battery acts as the utility company of the consumer(s), who tasks it with participating in the day-ahead market to buy and/or sell power for the consumer(s). It combines the day-ahead price forecast and the energy demand forecast of the consumer to determine the optimal moment to buy and sell power. The optimal buy and sell strategy is approached as a mixed-integer linear programming problem, with the objective to maximize the profit, and the constraints set by the battery specifications and the consumer demand. Furthermore, the battery algorithm is also tasked with preventing any form of imbalance, such that it helps grid stability. Lastly, the battery can provide capacity in the aFRR market, which will be the main source of income. The capacity will not be sold contractually but will be bid on the voluntary aFRR market, such that the capacity can be flexibly altered based on the latest consumer demand forecast. Finally, the profitability of a home battery is compared between ownership by a single consumer and a group of consumers.

The report will be structured as follows. Chapter 2 covers the general Dutch power market. The theory behind the machine learning algorithms applied is covered in chapter 3. Chapter 4 covers the optimization algorithms. The description of the total battery operating algorithm is covered in chapter 5 and the machine learning models used in this algorithm are explained in chapter 6. Chapter 7 covers the operational results of utilizing the battery in the real world. Lastly, the conclusion and discussion of the results are presented in chapter 8.

2

The electricity market in the Netherlands

This chapter covers the general market setup for the Dutch power market. Section 2.1 introduces the general Dutch power system, followed by Section 2.2 covering the general market structure. Lastly, the day-ahead market is covered in Section 2.3 followed by the imbalance market in Section 2.4.

2.1. The Dutch power system

Typically, an electrical power grid is separated into multiple segments, which can be distinguished by the voltage level. The high voltage (HV) grid, also known as the transmission grid, in the Netherlands has a voltage level ranging between 110 kV and 380 kV (TenneT, 2022c) and is operated by the Transmission System Operator (TSO). In some other countries, such as Germany, the transmission grid is operated by multiple parties, but in the Netherlands TenneT, the Dutch TSO, is solely responsible for the transmission grid. The high voltage grid cascades down into the medium voltage grid followed by the low voltage (LV) grid. Both together are called the distribution grid, which is operated by the Distribution System Operator (DSO). Contrary to the single TSO, the Netherlands has 6 DSOs, each responsible for its area.

The distribution system is directly connected to the transmission, schematically shown in Figure 2.1. It can be observed that traditionally the generation units are connected to the HV-grid, while the residential consumption is connected to the LV-grid. This traditional setup allowed that energy simply went from the generation to the residential consumer. The introduction of renewables, but especially solar PV, disrupts this classical hierarchy and introduces the need for energy to flow in both directions, from and towards the residential areas.

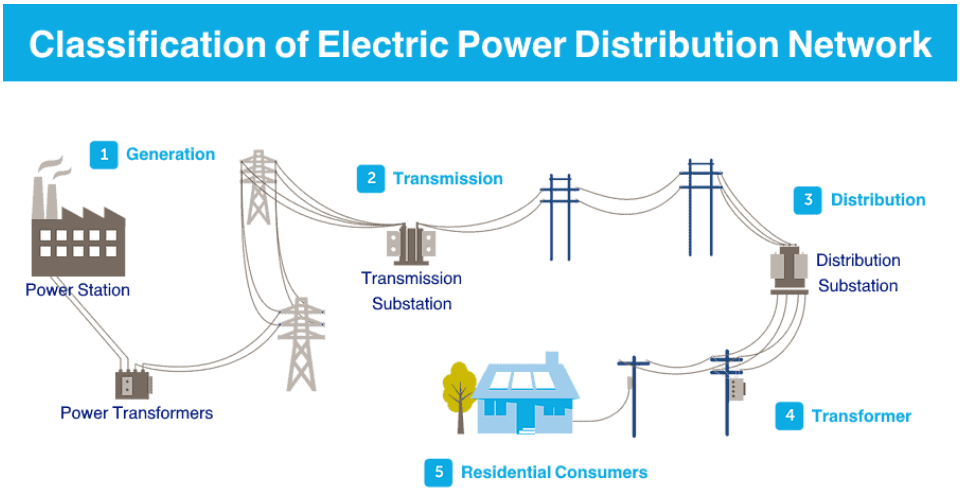


Figure 2.1: A schematic overview of the power distribution network (Electrical Technology, 2021)

Power generation in the Netherlands is mainly produced by gas and coal-fired power plants assisted by a small nuclear station, which contributed 3% to the total produced energy (CBS, 2022). Those power plants are considered the conventional generation part, which is, up to a certain degree, flexible in steering the power output. All turbines are bound to limits for the de- and increase in power over time, the so-called ramp rate. Currently, the ramp rate for a modern gas turbine is roughly 10% of the rated power per minute (Abudu et al., 2021) but differs per power plant. Coal-fired power plants are even slower, which is at most 4.5% of the nominal power per minute, based on the power plants in Germany (Deloitte, 2019).

2.2. Electricity market structure

The Dutch electricity market consists of multiple segments with each their characteristics. The segments and their names are shown in Figure 2.2. The futures market is the long-term market, containing a wide variety of products, such as Years; Seasons; Quarters; Months, and Weeks. Often, the further ahead the less liquid a product is, or not even tradable at all. Future contracts deliver a certain amount of power for every hour of every single day of a period. Thus, a weekly contract would grant the owner the delivery of 24*7=168hours of the predefined amount of power per hour.

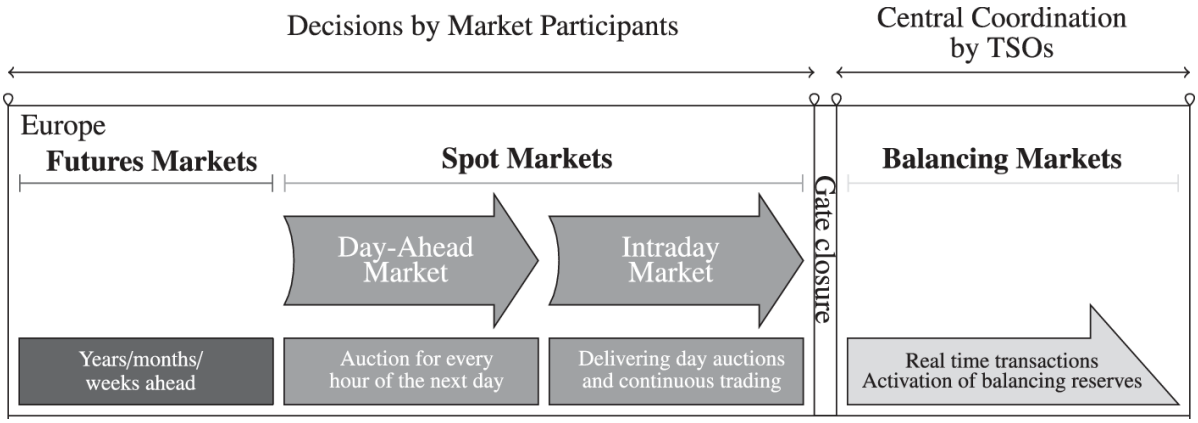


Figure 2.2: The general Dutch market overview as adapted from Bichler et al. (2022)

The downside of this type of delivery is that it is difficult to hedge for a consumer portfolio as consumers often have a variable demand pattern. Besides, the demand differs based on the season, such that the winter

period sees a larger consumption compared to the summer. This seasonality pattern will increase as more consumers are opting for rooftop PV, which reduces the grid consumption in the summer. Furthermore, the changing demand pattern makes it difficult to fully hedge the exposure for a utility company. The risk posed by not being fully hedged, or wrongly hedged, will be transferred indirectly onto the consumers by an increase in fees.

2.3. Spot markets

In Europe, the spot market is considered the day-ahead and intraday market, contrary to the United States, that only knows a single spot market (Bichler et al., 2022). The day-ahead market is the market that resolves the short-term consumption/production forecasts that have not been taken into account in the futures market. The Dutch day-ahead market closes daily at 1200 CET, which is equal to most countries united under ENTSO-e regulations, except for the UK and Swiss which close at 1020 and 1100 respectively (EPEX, 2022). This means that, at most, the production and generation of 36 hours ahead need to be forecasted. However, to deal with this uncertainty the intraday market exists in addition to the day-ahead market.

2.3.1. Price settlement of the day ahead market

The electricity markets are characterized by an auction type of market clearing, where the price for an hour is set based on the most expensive activated power plant. The hourly price is obtained by sorting all the bids in generating volume from low to high based on their ask price, this results in a stairs-shaped figure, such as shown in Figure 2.3. The same is done for the demand, where the demand is sorted from a high price to a low price. The volume where demand and supply intersect is the amount of power that will be dispatched for that hour, moreover, the price on the intersection is the price that is set for that hour. All plants below that bidding price are dispatched and are paid the price that was set. Thus, as shown in Figure 2.3, the blue area is the profit obtained by the activated power plants.

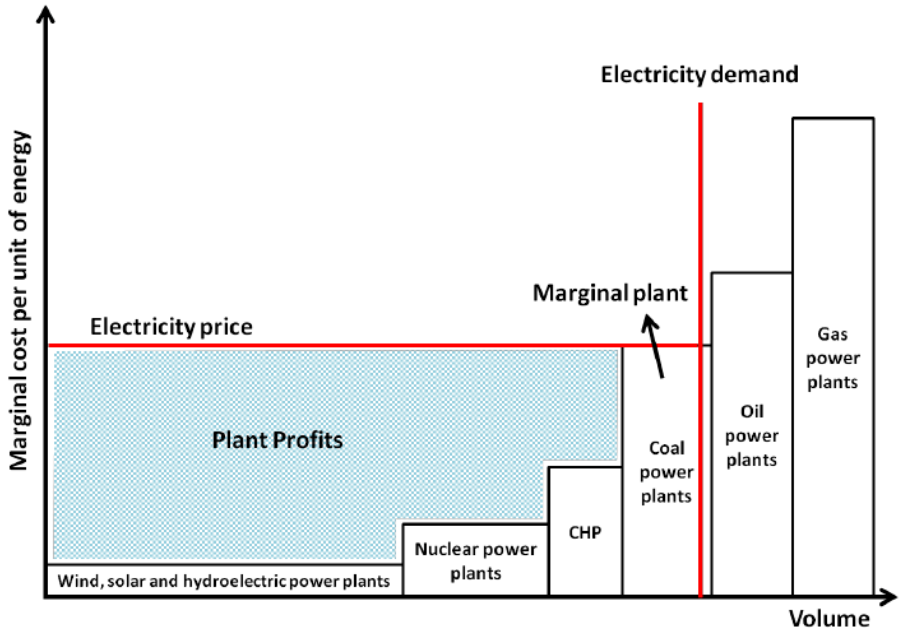


Figure 2.3: Order dispatch in the electricity markets, where the blue area represent the profit obtained by the activated power plants (Sauvage & Bahar, 2013).

Conventionally, when assuming perfect competition, the most optimal bidding price for a power plant is

a price equal to their marginal costs (I. Perez-Arriaga, 2016). which mainly contains the cost of the fuel needed. However, it can be beneficial to place a bid at an extreme price to push the price up in case of shortages. Events of extreme price have never been observed before 2021, however, in 2021 some European countries, such as France and the United Kingdom, have had day-ahead prices above 1000 EUR/MWh.

2.4. Balancing markets in the Netherlands

The balancing system has the purpose of keeping the national grid frequency at 50 Hz. A deviation from the main frequency occurs if production and consumption are not matched, meaning that if the power consumption is larger than the production, the frequency will drop. Vice versa will increase the frequency. If the frequency drops below a certain threshold, generators inside power plants shut themselves off to prevent damage. The result is a larger loss of power and thus a black-out. But also, smaller frequency deviations away from the national 50 Hz could lead to damage to electrical appliances, stressing the importance of a stable frequency.

The imbalance containment reserve consists of a layered structure. As soon as a frequency deviation occurs, the Frequency Containment Reserve (FCR) kicks in to prevent the frequency to deviate too far. After a couple of minutes, the power plants providing the FCR will be covered by the automatic Frequency Restoration Reserve (aFRR). This covers the short to mid-term frequency stability. After a while, if the frequency is still not restored, the manual Frequency Restoration will be activated to deal with the system instability. Graphically the shift between the different reserves is shown in Figure 2.4.

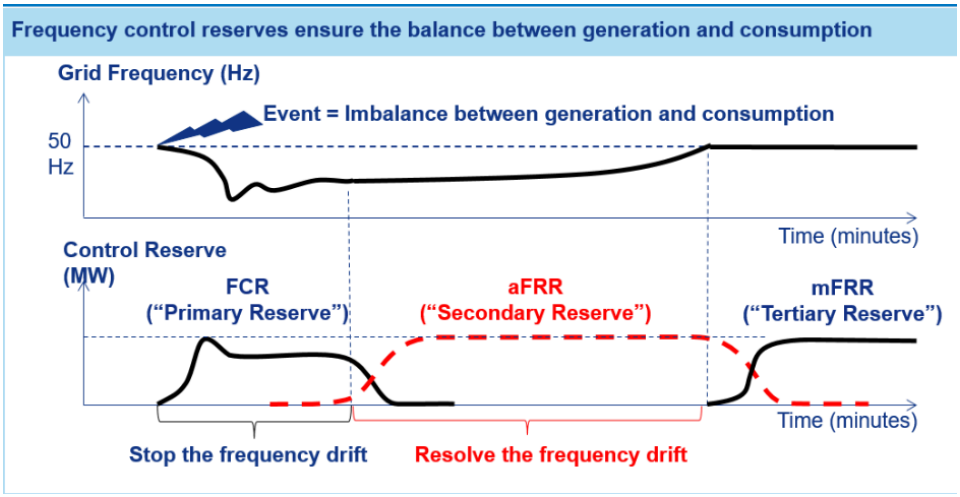


Figure 2.4: After a frequency deviation occurs the FCR kicks in in order to stop the immediate issue, followed by aFRR that slowly takes over from the power plants contributing to the frequency containment (Poirot & Baschet, 2021).

The imbalance markets are also liberalized markets, meaning that the market is encouraged to solve the imbalance itself. This is incentivized by creating a market for imbalances with prices different from the day-ahead or intraday market. In Europe, two different balancing pricing systems exist, single- and dual-pricing. Single pricing can be found in Norway, Germany, and Spain, while dual pricing exists in the U.K., Netherlands, and Italy (Gibescu et al., 2008). The scope of this project is on the Netherlands, thus dual balancing, as known in the Netherlands is considered. Single pricing means that just a single price exists for a single Program Time Unit (PTU), a time unit over which the imbalance is calculated equalling 15 minutes in the Netherlands. In a dual pricing system, two prices can occur in a single PTU as the grid is both regulated upward and downward.

The different regulatory states consist of upward and downward regulation, two-sided balancing, and in-balance. In-balance is straightforward, meaning that the system is in balance and no additional measures need to be taken by the TSO. If there is too much power on the grid, which happens when there is more produced than consumed, we have a **downward regulation**. The other way around, more consumption

than production, is called the **upward regulation**. If during a single PTU both the upward and downward regulation states are used, we have a dual regulation.

2.4.1. Frequency Containment Reserve (FCR)

The Frequency Containment Reserve is a product sold to the TSO, wherein the seller is contractually obliged to regulate the power output to minimize grid-frequency deviations (Engels et al., 2019). In Europe, the allowed frequency deviation is 200 mHz from the nominal value of 50 Hz (ENTSO-E, n.d.). The power should be regulated within a specified time frame, for Continental Europe, in 30s. BESS allow for fast response, which makes them perfect for FCR. However, the problem for a BESS is that the FCR requires continuous activation during the contractual period, which becomes a problem when the BESS is either full or empty. The Dutch TSO allows for batteries to be used for FCR, as long as they fulfill predefined conditions and follow guidelines for (dis)charging during the contracted period (TenneT, 2022b). Utilizing a battery on FCR is not in the scope of this research, thus the specific conditions for a battery are here not covered.

The FCR is, from an asset owner's perspective, an easy product. A frequency measurement device installed on the asset sends a signal to the assets to either reduce production/increase consumption or to increase production/decrease consumption (Next Kraftwerke Benelux B.V., 2022). Both the increase in consumption and the reduction in production lead to the same result, a decrease in the frequency. The increase in production and the reduction of consumption lead to an increase in frequency.

Traditionally a gas-fired power plant would be contracted for FCR as those can flexibly increase and decrease the production, but are limited to their ramp rate. The introduction of batteries introduced nearly instant response, as batteries are not bound to a ramp rate.

FCR is sold in a block of 4 hours. The participating party gets a fixed fee per available MW (TenneT, 2022b). The fee is obtained by creating a merit order for all the bids. Based on the required volume, the assets asking for the lowest price are contracted until the required volume is reached. The system is different than the other markets, where the highest price sets for the whole market. The FCR is a paid-as-bid system, where the asked fee is the premium that the party obtains (Next Kraftwerke Benelux B.V., 2022).

2.4.2. Automatic Frequency Restoration Reserve (aFRR)

The automatic Frequency Restoration Reserve (aFRR) knows two types of participants, contractual and volunteers. Parties committing to the contractual aFRR must place bids for the agreed amount of power for at least the contracted period, which is often longer than 24h. The contracted party obtains a premium based on the contracted power. Parties not participating on a contractual basis, but voluntarily, do not get a premium for their power. The differences between the voluntary aFRR and the contractual aFRR are summarized in Table 2.1.

	aFRR	**Voluntary aFRR**
Minimum order size [MW]	1	1
Contracting method	Obligatory bidding	Voluntary bidding
Financial settlement	Capacity auctioned (EUR/MW/PTU) & Energy supplied (EUR/MWh)	Energy supplied (EUR/MWh)

Table 2.1: The differences between the contractual aFRR and the voluntary aFRR

Every participating party, both contracted and voluntary, places a bid for the available power and their bid/ask price in EUR/MWh. The volume and prices are placed in merit order, such as shown in Figure 2.5. For the downward regulation state, the prices are sometimes negative, this means that TenneT pays the utility operator to reduce their power output or another party to increase their power consumption. For the upward regulation, TenneT pays positive prices and receives negative prices (TenneT, 2020).

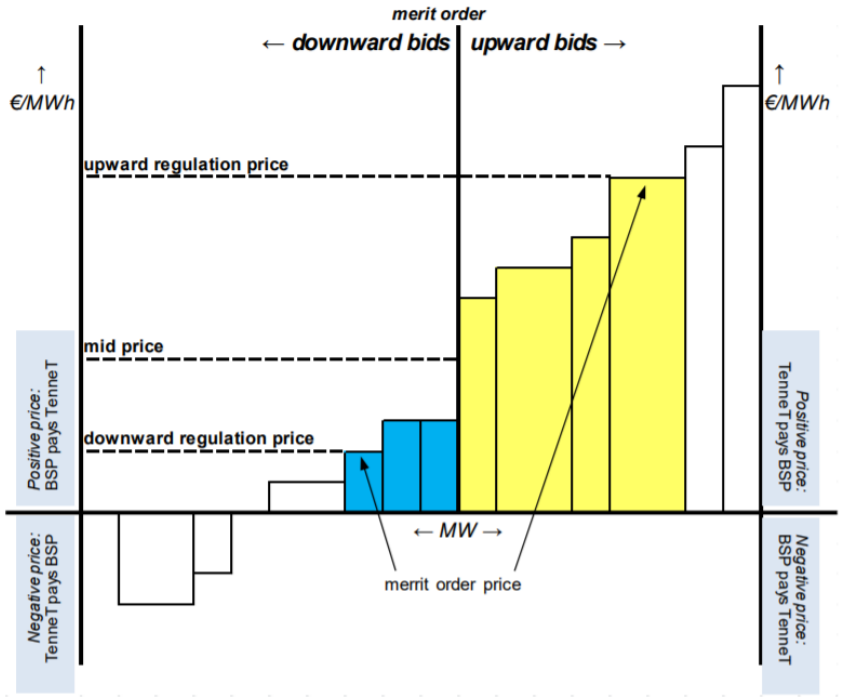


Figure 2.5: The merit order for the FRR is set in the same method as for the spot market. In the case of the FRR Tennet pays the BRP for positive bids during upward regulation and for negative bids in the downward regulation (TenneT, 2020).

In the case of an upward regulation, the cheapest power is activated first and deactivated last. The price for a PTU is set based on the highest activated power bid, however, the revenue obtained is based on the delivered energy and not on the activated power. Therefore, a low ask price would result in the longest activation of power. This is beneficial if during the PTU a high price bid is touched, creating both a long activation period and a high price.

Bids placed for aFRR should comply with the market rules shown in Table 2.2. One of the rules shows the minimum required ramp rate, this is the ramp rate for the bid in volume. Thus if 20 MW is bid-in, the producer should ramp their power plant up to 1.4 MW per minute. This is the reason why sometimes extreme prices occur with relatively small imbalance volumes. For example, consider 5 blocks of 20 MW each, with a ramp rate of 10 %. If the imbalance hits 10 MW within a minute, that means that 10MW of aFRR needs to be activated. The first order of 20 MW can only deliver 2 MW, due to the ramp rate of 10%. Therefore, all 5 bids need to be activated to mitigate the sudden imbalance. The whole PTU gets the price that the 5th bid set, even though the imbalance only reached 10 MW, a fraction of the 100 MW available.

	aFFR
Max price	Spot + 1000 EUR/MWh
Minimum price	-999.9 EUR/MWh
Bidding based	Contractual/ voluntary
Ramp-up/down minimum	7%/min (20% as of 1-07-2022 (TenneT, 2022a))
Allowed volume	{1,999}

Table 2.2: The regulation rules in the Netherlands for the aFFR (TenneT, 2018)

TenneT publishes for10 different power levels the corresponding price, which allows parties to get a feeling for the prices. The prices are published for the lowest price, 100MW, 300MW, 600MW, and the maximum power level for both the upward and downward bids. During a day those prices can vary, as shown in Figure 2.6.

From the image, it is clear that TenneT is not yet adjusted to the extreme prices. The published order book data give, to some extent, insight into the possible price levels during a PTU. However, a one-to-one translation of the total imbalance with a certain price is not possible.

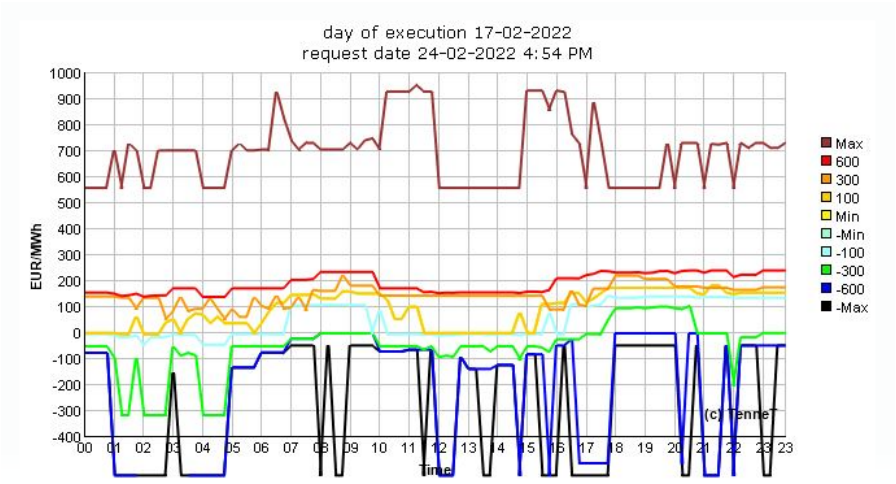


Figure 2.6: The aFRR orderbook for the 17th of february 2022. (TenneT, 2021a)

2.4.3. Manual Frequency Restoration Reserve (mFRR)

The mFRR is the last reserve that takes over from the aFRR providing parties in case of a large or long-lasting frequency deviation. The Netherlands knows two forms of mFRR, namely the mFRRda and the mFRRsa formally known as the incident reserve and reserve power. The mFRRda is the directly activated mFRR, which needs to directly deliver the required power. The upward mFRRda is required to deliver the full power within 15 minutes, while the downward mFRRda provider needs to fully deliver their capacity within 10 minutes after activation (TenneT, 2021b). The mFRRda provider needs to deliver its capacity for at least 60 minutes. mFRRda is contracted by the TSO for at least a week or month. The contracted party is obliged to bid for every single PTU of the contracted period (Vugt, 2020).

Besides the directly activated mFRR, the TSO also contract scheduled activation reserves mFRRsa. The party contracted for mFRRsa gets 1 to 2 PTUs before delivery a message by the TSO that they will need to provide capacity. This will be for at least the whole PTU. Contrary to the mFRRda the mFRRsa does not know any contractual obligation, but all the bids are applied voluntarily. The voluntary bids are merged into the merit order obtained by the mFRRda, which determines the bids that get activated. Another difference between the mFRRda and the mFRRsa is the amount of power required, the first requires bids of at least 20 MW, while the second allows bids of 1 MW. The differences between the two are summarized in Table 2.3

	mFRRda	**mFRRsa**
Minimum order size [MW]	20	1
Contracting method	Contractual bidding	Voluntary bidding
Financial settlement	Capacity auctioned (EUR/MW/PTU) & Energy supplied (EUR/MWh)	Energy supplied (EUR/MWh)

Table 2.3: The differences between the mFRRda and the mFRRsa as adapted from (Vugt, 2020)

3

Machine-learning algorithms

This section covers the main algorithms used in this research, starting with the widely known neural networks in Section 3.1. Section 3.2 covers the working of the XGBoost algorithm, followed by the ADWIN algorithm in Section 3.3. Lastly, the chapter concludes with the Relief Algorithm for feature selection in Section 3.4.

3.1. Neural Networks

Neural networks are widely applied in the machine learning discipline as they are often seen as one of the most advanced forecasting techniques. The idea behind neural networks is quite old, but it was not until computers have become more advanced and especially faster, that using neural networks is becoming an attractive forecasting model.

3.1.1. Mathematical operation

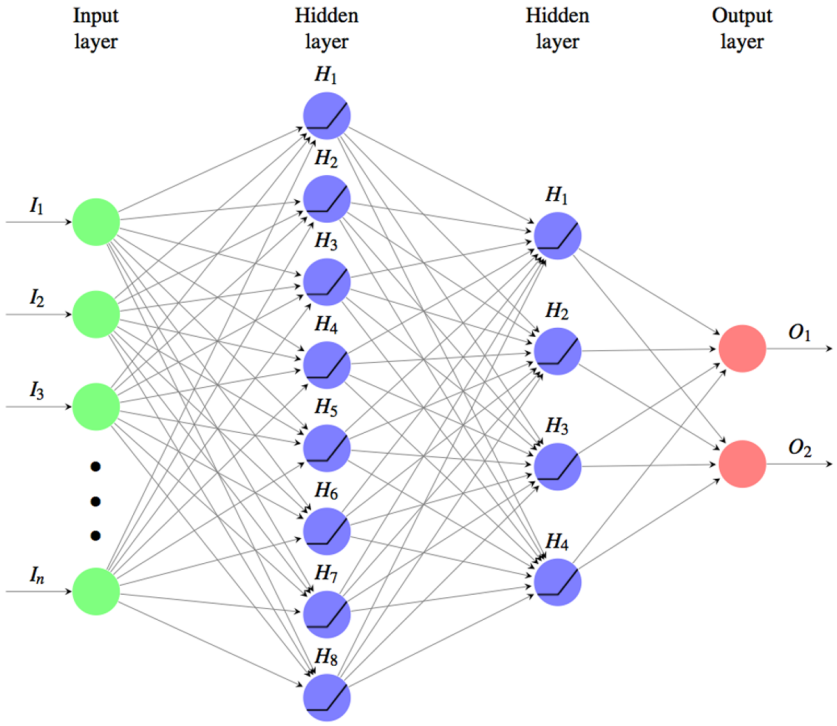


Figure 3.1: The general layout of a neural network containing two hidden layers and two outputs. The input variables flow trough the network to the final output on the right side.

A neural network is based on the idea of a neuron system, which is why the connection points are often called neurons or nodes, visualized in Figure 3.1 as circles. A row of neurons is called a layer, with the first layer being the input layer, visualized in green. The input layer contains the input values required for a prediction. The blue layers are the hidden layers, this is the black box of the algorithm. For an end-user, it is unknown how a decision has been derived from the input variables. Lastly, the output layer, where the number of nodes depends on the number of values required to be forecasted.

In a fully connected network, every neuron inside a layer is connected with all the neurons of the previous and next layers. The connection between two neurons contains a weight that is multiplied by the value that came from the previous node, such that the value for a neuron is the sum of all previous nodes multiplied with the respective weights. This can be written as

$$H_1 = I_1 * w_1 + I_2 * w_2 + ... + I_n * W_n \tag{3.1}$$

or simplified to

$$H_1 = \vec{x}^T \vec{w}, \tag{3.2}$$

where $\vec{x}$ is the vector of all the values in the previous nodes, which can be the input *I*. $\vec{w}$ is the vector containing all the weights connecting the previous nodes to the current node. The weighted sum is extended by adding a bias, before putting the result into an activation function, such as shown in Figure 3.2. The non-linear activation function introduces the non-linearity to the neural network. Without the non-linear activation function, the neural network is just a linear function of chained linear multiplications.

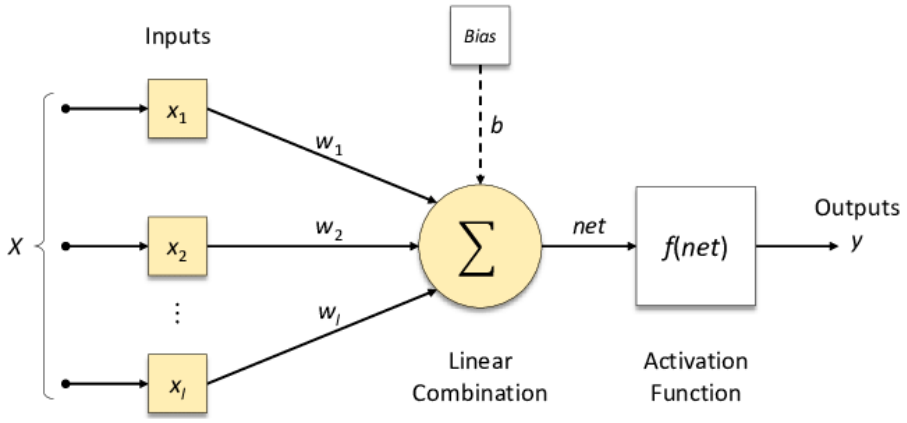


Figure 3.2: The path information follows trough a single node of a layer (Parmezan et al., 2019).

Figure 3.2 uses a activation function f to activate the neuron, such that the output of the node can be denoted as

$$y = f(\vec{x}^T \vec{w} + b). \tag{3.3}$$

The weights inside the neural network are, at first, initialized by a random number, where the random number is drawn from a user-defined normal or uniform distribution (Aurélien, 2017). After which the neural network makes a prediction based on the input. The output is compared to the actual value and based on a loss function the error is calculated. Afterward, the contribution of each output connection towards the error is computed by using the chain rule, such that the contribution of the neuron connected to the output is calculated by $\frac{\partial L}{\partial z}$ where L is the loss that is minimized and z the output of the node, shown in Figure 3.3. The contribution of the next node is again obtained by using the chain rule. Consider the output of the previous node to be y, which means that the contribution is equal to

$$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial y}. \tag{3.4}$$

The minimization of the loss function is done by applying the gradient descent algorithm. Gradient descent tries to find the optimum by iteratively taking a gradient descent step in the direction of the steepest gradient. The algorithm continues doing so until it converges to a local or global minimum. The size of the step is determined by the learning rate of the algorithm and influences the chances of finding the global minimum.

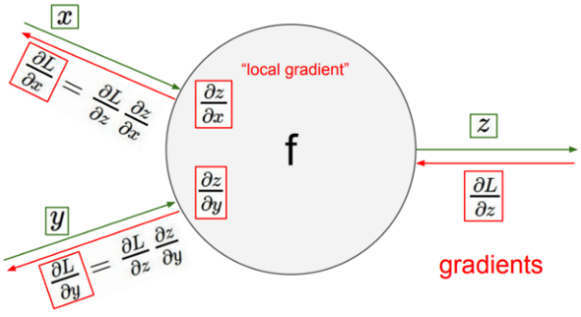


Figure 3.3: A local gradient showing the different connections contributing to the total loss.

The calculation of weight updates is called back-propagation, an algorithm proposed by Rumelhart et al. (1985). Back-propagation computes the gradients by using an automatic differentiation (Aurelien, 2019). Followed by a gradient descent step to update the weights and biases by using the error gradient which the algorithm computed. This process is repeated until the model converges.

3.1.2. Learning rate

The learning determines the step size of the gradient descent and is thus an important factor to monitor to obtain stable results. The loss graph should be a smooth decreasing curve, as shown in Figure 3.4. When the learning rate is set too high the model does not converge to a global minimum but bounces around, resulting in a type of figure shown in Figure 3.5. When the learning rate is too high, the model becomes dependent on the initial seed for good results, as a model with a high learning rate is unlikely to end up in a global minimum. The model would just bounce around near a minimum.

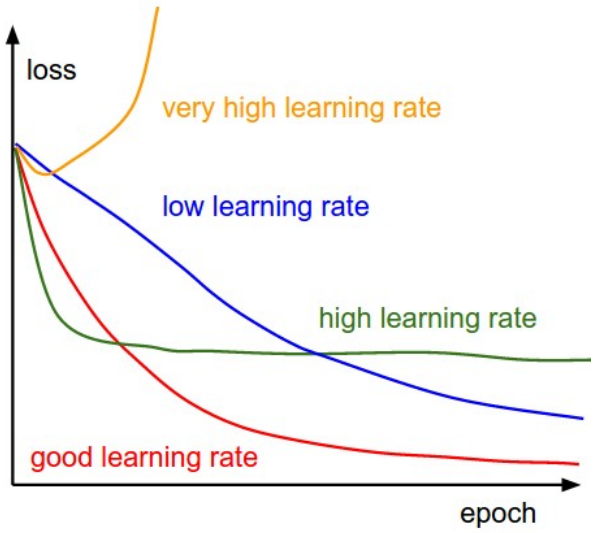


Figure 3.4: The effect of different learning rates on the loss function (CS231n, 2022)

Hypothetically, when using a high learning rate, it would be possible to obtain similar result compared to using the right learning rate, both could find the global minimum. Finding the global minimum with a high learning rate could be achieved by randomly trying different initial seeds, as there could be an occasion where the random initialized state is close to, or exactly at, the global optimum. However, finding the right seed is computational heavy as it adds an extra parameter that requires tuning, therefore it is important to choose the right learning rate.

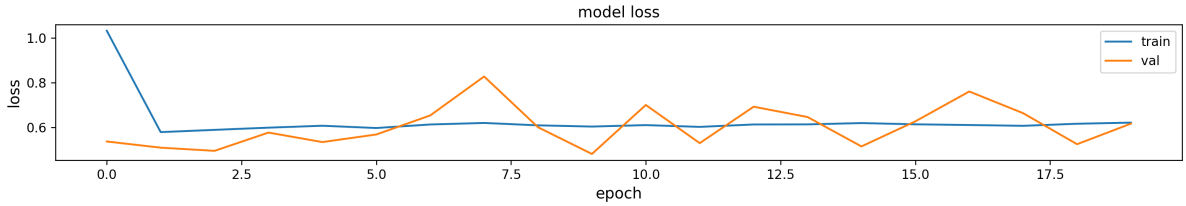


Figure 3.5: A too large learning rate results that the model converges fast, but not in an optimal way. The final result is extremely dependent on the random seed in order to obtain good results.

Just using a small learning rate will not work either, as that could result that the gradient descent getting stuck in local minima. A method to aid the model is by using an adaptive learning rate. Keras allows adaptively changing the learning rate, which can be done by using a time-based decay, which would look as follows (Brownlee, 2017):

$$LearningRate = LearningRate \times \frac{1}{1 + decay \times epoch} \tag{3.5}$$

The effect of the decay on the learning rate over time is shown in Figure 3.6, using a start rate of 0.01 and a decay of 0.0001. Using a decreasing learning rate with time allows the algorithm to run fast to the

optimum and not get stuck in a local minimum, followed by slowly coming closer to the optimum without overshooting the global optimum.

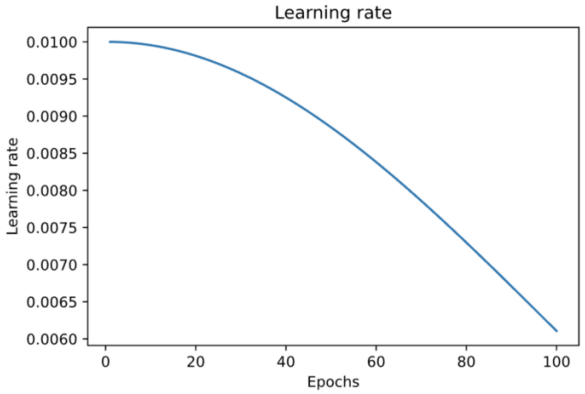


Figure 3.6: The learning rate decreases at every epoch in order to use the large the learning rate early in the training process, while later on fine-tuning the result with a smaller learning rate (B. Chen, 2020).

The adaptive learning rate can be applied in keras by using the callbacks, scuh as shown below

```
from tf.keras.callbacks import LearningRateScheduler


def time_based_decay(epoch, learningrate):
    return learningrate * 1 / (1 + decay * epoch)


history_time_based_decay = model.fit(
    X_train,
    y_train,
    callbacks=[LearningRateScheduler(time_based_decay)]
)
```

Or use one of the predefined Keras function available as shown below.

```
lr_schedule = keras.optimizers.schedules.ExponentialDecay(
    initial_learning_rate=1e-2,
    decay_steps=10000,
    decay_rate=0.9)

optimizer = keras.optimizers.SGD(learning_rate=lr_schedule)
```

3.1.3. Optimization algorithms

Another option, instead of manually applying a learning rate schedule to the Stochastic Gradient Descent (SGD), is choosing an optimizer that has this already built-in. The most common optimizers available within the TensorFlow package are summarized in Table 3.1.

	Strength
Adadelta	More robust version of Adagrad (Peltarion, 2022b)
Adagrad	Sparse gradients (NLP and computer vision) (Brownlee, 2021)
Adam	Commonly the best choice to test first (Peltarion, 2022a)
Ftrl	Shallow models with a large and sparse feature sets (McMahan et al., 2013)
RMSprop	Noisy data and online learning (Brownlee, 2021)

Table 3.1: The most common optimizer with each their own strength.

One of the findings by Choi et al. (2019) and covered by Park (2021), shows that Adam or RMSprop never

underperform SGD or momentum. However, the hyperparameters of the optimizer should be tuned for that to be valid. This is different than the commonly accepted suggestion that the Adam optimizer does not need any tuning.

Selecting the right optimizer can help a great deal in the time it takes to converge, but also in the performance of the model. However, model performance is not only dependent on the optimizer, but also on the ability of the neural network to generalize.

3.1.4. Regularization

To help the neural network generalize, the network architecture can use regularization terms. Regularization is a set of techniques to reduce model complexity, which helps to prevent over-fitting. The most common regularization techniques in neural networks are the L1 and L2 regularization terms and the drop-out layer.

The L1 regularization term is commonly known as Lasso regression. Lasso regression adds the sum of the absolute weights, multiplied by a term α, as a penalty term to the loss function. This penalty encourages the neural network to set the weights to zero and thus reduces the general complexity (Karim, 2018).

L2 regularization, also known as Ridge regression, also penalizes the sum of the weights. However, Ridge uses the square of the weights. This results in weights close to zero, but not zero. Keras applies the regularization on a per-layer basis and knows the following type of regularization. **Kernel regularization**, formerly known as weight regularization, tries to reduce the weights of the layer. The **Bias regularization** penalizes the bias of the layer and lastly, the **Activity regularization** puts a penalty on the output of the layer.

Lastly, the dropout regularization technique. Dropout simply ignores neurons during training. A node has a probability of *p*, defined by the user, of being ignored during a training step. Dropping nodes out forces the other nodes to cover to carry the information forward. This technique helps in preventing the network from overfitting (Brownlee, 2019).

3.1.5. Batch Normalization

Another type of problem known to neural networks is the exploding gradient problem, which simply means that gradients become extremely large compared to the others so that they dominate the network. Using the He-normal initialization in combination with a type of ReLu prevents the problem of exploding gradients at the beginning of training. However, it does not prevent them from returning during training. To tackle the problem of the exploding gradients Ioffe and Szegedy (2015) proposed Batch Normalization.

Batch Normalization zero-centers and normalizes the layer input followed by rescaling and offsetting the values. For a *d*-dimensional input $x = (X^{(1)}...x^{(d)})$, each dimension is normalized by

$$\widehat{x}^{(k)} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{\mathsf{Var}[x^{(k)}]}}, \tag{3.6}$$

where $\widehat{x}_k$ is the zero-centered and normalized values for dimension *k*. This transformation is done on a per-batch basis. For a batch of size *m*, whereby *k* is omitted for clarity, the operation starts by taking the mean of the batch, such that:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i. \tag{3.7}$$

Next, the variance of the batch is taken:

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2. \tag{3.8}$$

Followed by the normalization:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \tag{3.9}$$

where ϵ is a smoothing term that prevents divisions by zero. After the normalization, the output is shifted with the dimension specific variable, such that the final output is equal to:

$$y_i^{(k)} = \gamma^{(k)} \hat{x}_i^{(k)} + \beta^{(k)}. \tag{3.10}$$

γ is a scale parameter, and β is the output shift. Both γ and β are parameters that need to be learned and which increase the number of learnable parameters in the neural network, and thus the learning time for a single epoch. However, due to the normalization, the network as a whole reaches' convergence faster. Thus, the overall training time, wall time, of the network is reduced, and the time a single epoch takes is increased.

The reason that batch normalization works can intuitively be linked to the reason that the normalization of the input data works. Normalization of the input data helps the gradient descent to converge fast and properly. After the multiplications of a layer in the neural network, the output of the layer is not normalized, nor zero centered. The batch normalization layer readjusts this, by again normalizing and zero centering. Graphically, this can be seen as that the input of a layer is the input of the neural network, shown in Figure 3.7.

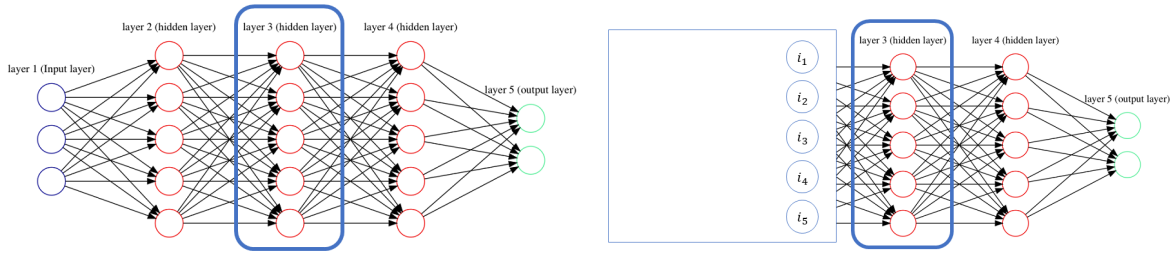


Figure 3.7: Adding a Batch Normalization layer within a neural network aids the neural network in a similar fashion as the normalization of the input data. If a Batch Normalization layer is applied between the second and third layer in a network, the output from the second layer would be normalized before going into the third layer, as would happen with data in the input layer. This normalized data then acts as input to the third layer.

A Batch Normalization layer can even be used as the first layer of the Neural Network, which eliminates the need to normalize (e.g., with Scikit StandardScaler) the training data (Aurelien, 2019) as the Batch Normalization layer will take care of this. Furthermore, Batch Normalization also works as a regularization method, according to Ioffe and Szegedy (2015) the dropout layer in the Neural Network can be reduced in strength or even removed.

Lastly, the Batch Normalization within Keras performs differently in training compared to in testing mode. Within the training mode, the batch normalization layer uses the mean and standard deviation of the current batch. But in testing mode, the model uses the moving mean and standard deviation of the batches seen during training (Keras, 2022a). The Batch Normalization layer has a momentum parameter, which is tunable. The momentum allows controlling how much of the statistics from a previous batch to include in updating the moving mean and standard deviation. A momentum of 0 only considers the mean and standard deviation of the last batch. Typically, a momentum close to 1 is used, such as 0.9, where more 9s are added for larger datasets and smaller batches (Aurelien, 2019).

3.2. XGBoost

The XGBoost algorithm is a widely adopted machine learning algorithm due to its flexibility, speed, and performance. The model is often the winning algorithm in Kaggle competitions, especially those challenges

containing structured data (Ye, 2020). The popularity can partially be contributed to the easiness of running first predictions. The model has a natural feature selection method that filters out features with little predictive value, this is beneficial for users with relatively little knowledge about machine learning.

XGBoost is an ensemble tree model that fits in the family of boosting, just as the name suggests. In an ensemble model, the final prediction is the sum of the predictions of the individual trees, graphically represented in Figure 3.8. The boy, in the first function, obtains the value 2 from the first tree for being both a male and below 15. The second tree puts the boy in the left bucket for using the computer daily, resulting in a score of 0.9. Combining the two scores yields $2 + 0.9 = 2.9$. The same can be done for the older gentleman shown in the second equation.

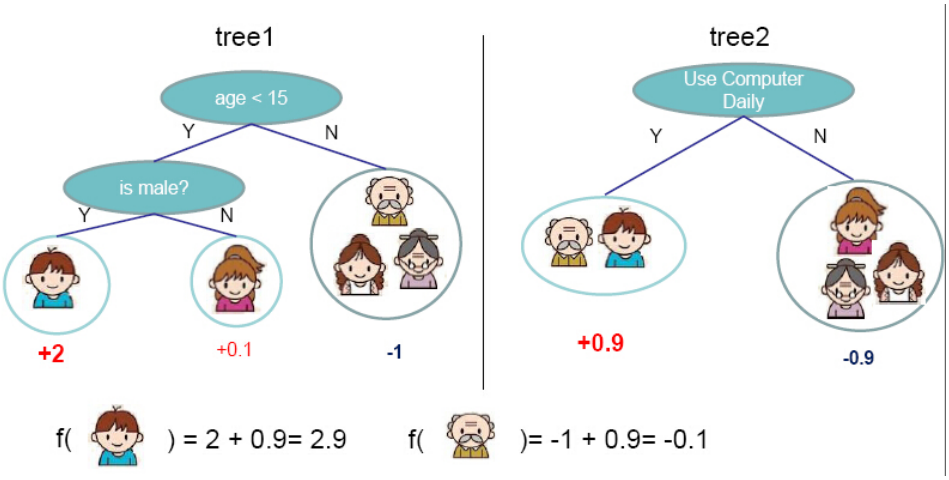


Figure 3.8: Working of a tree ensemble model (T. Chen and Guestrin, 2016)

Figure 3.8 shows how to obtain predictions from a tree-based model. However, to obtain the predictions, the model first needs to be fitted. Just like any other machine learning algorithm, the XGBoost algorithm has an loss function, which we try to minimize. The loss function, at iteration *t*, of the XGBoost model can be written as

$$\zeta^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t), \tag{3.11}$$

where *n* is the number of instances and l is a convex loss function that measures the distance between the actual value of instance *i* and the sum of the prediction of iteration *t*-1 and a tree structure of iteration *t*. y_i is the actual value and $\hat{y}_i$ the corresponding prediction for instance *i*. $f_t(x_i)$ is a tree structure for iteration *t* during instance *i*. Lastly, the Ω is a penalty for model complexity, such that

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2. \tag{3.12}$$

In the equation γ is a regularization parameter that sets the minimum loss reduction required by a leaf. *T* is the number of leaves and *w* the sum of the scores of the leaves. λ is a ridge Regularization term which can be tuned by the user. To show the model complexity penalty with an example, the tree in Figure 3.9 is used to fill Equation 3.12. The tree has 4 leaves, thus $T = 4$. The tree has the three scores, namely 2, 0.5 and 1, therefore $w = [2, 0.5, 1]$. Combining the two into Equation 3.12 gives $\Omega = 4\gamma + \frac{1}{2}\lambda\left(2^2 + 0.5^2 + 1^2\right) = 4\gamma + 2.625\lambda$.

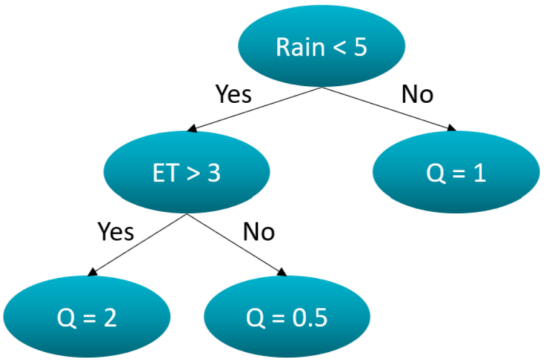


Figure 3.9: Working of the regularization term in the XGBoost model (Van der Munckhof, 2020)

In order to minimize Equation 3.11 a second-order Taylor expansion is used (T. Chen & Guestrin, 2016), such that

$$\zeta^{(t)} = \sum_{i=1}^{n} \left[l\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t), \tag{3.13}$$

where $g_i = \partial_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$ and $h_i = \partial^2_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$, which are the first and second order of the Taylor expansion of the loss of the previous iteration (Van der Munckhof, 2020). Given that $l\left(y_i, \hat{y}_i^{(t-1)}\right)$ is constant, it can be removed from the equation at step *t*. If we define w_j as the score of leaf *j*, then it follows that $f_t(x_i) = w_j$ if x_i belongs to lead node *j*. This can then can be substituted into Equation 3.13, resulting in

$$\begin{aligned} \zeta^{(t)} &= \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \\ &= \sum_{j=1}^{T} \left[\sum_{i \in I_j} (g_i) w_j + \frac{1}{2} \sum_{i \in I_j} (h_i) w_j^2 + \frac{1}{2}\lambda w_j^2 \right] + \gamma T \\ &= \sum_{j=1}^{T} \left[\sum_{i \in I_j} (g_i) w_j + \frac{1}{2} \sum_{i \in I_j} (h_i + \lambda) w_j^2 \right] + \gamma T \end{aligned} \tag{3.14}$$

The optimal weight w_j^* for leaf *j* is given by deriving the derivative of the objective function $\zeta^{(t)}$ with respect to 0, shown by

$$\frac{d\zeta^{(t)}}{dw_j^*} = 0, \tag{3.15}$$

resulting in

$$w_j^* = -\frac{\sum_{i \in I_j} (g_i)}{\sum_{i \in I_j} (h_i + \lambda)}. \tag{3.16}$$

The optimal weight obtained can be substituted into Equation 3.14 to obtain the best model loss for a fixed tree structure, which results in

$$\zeta^t(q) = -\frac{1}{2} \sum_{j=1}^{T} \frac{\left(\sum_{i \in I_j} g_i\right)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T. \tag{3.17}$$

In order to deal with the nearly finite possible tree structures *q*, the XGBoost algorithm utilizes a greedy algorithm which starts with a single leaf and iteratively adds branches to the tree. The gain of a split is

calculated by

$$\zeta_{split} = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma \tag{3.18}$$

Where I_R and I_L are the instances of sets of right and left nodes after the splits, such that $I = I_R \cup I_L$. γ penalizes the extra complexity of adding another split. After finding the optimal tree structure for the first layer, the next layer tree is fit onto the residuals, which is unique for the XGBoost compared with other tree based models, of the first one. The following tree is again fit onto the residuals of the previous, etc.. This process is visualized in Figure 3.10. The left columns of figure shows the tree that is fitted onto the data and the right figure the ensemble of trees. The middle column is fitted onto the residuals of the base estimator after which the result is ensembles again, where the prediction is closer to fitting the actual shape.

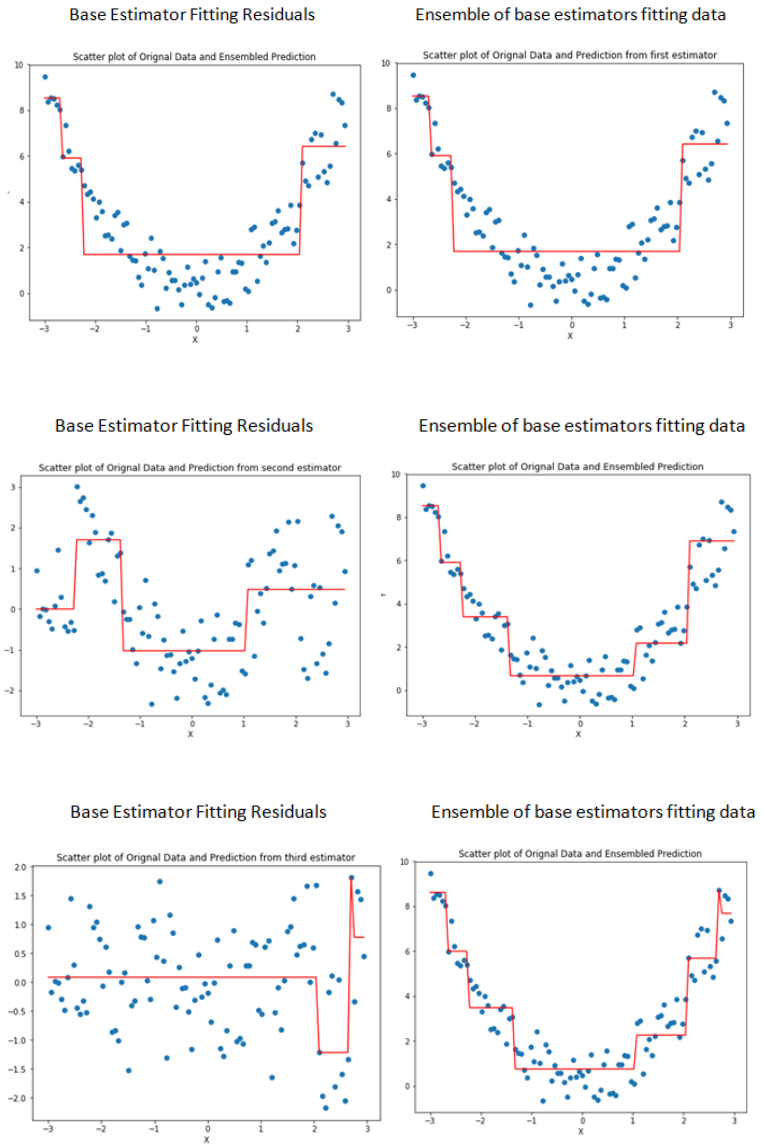


Figure 3.10: The effect of adding a new tree onto the residuals of the previous tree, slowly resulting into a shape representing the actual data (image from Jauhari (2019)).

The process of adding new trees is repeated until the specified number of trees have been reached or until

the loss function does not improve anymore. The last needs to be set by the user as well. As one can image, the more trees are fitted, the closer the estimation gets to the ground truth. However, too many trees results in over-fitting, as shown in Figure 3.11, where 150 and 250 trees are fitted on the example dataset. The issue of over-fitting is also the common pitfall of using the XGBoost algorithm and should be treated with caution.

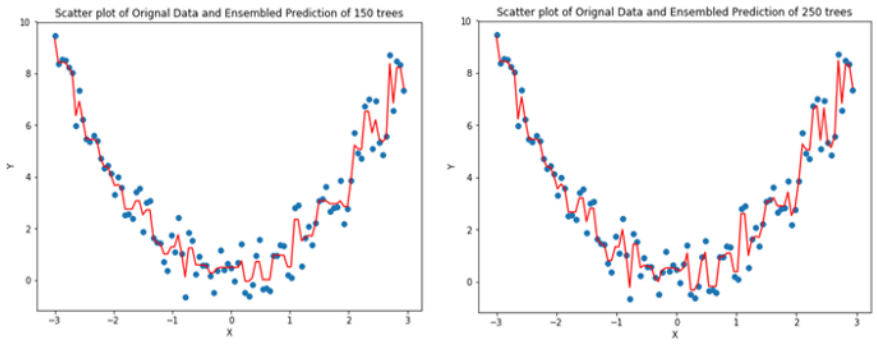


Figure 3.11: Fitting the initial tree onto an example problem (Jauhari, 2019)

3.3. Adaptive Windowing (ADWIN)

A problem found in many real-world data sets is the problem of concept drift. To deal with the concept drift occurring in our dataset the Adaptive Windowing technique, developed by Bifet and Gavalda (2007), is used. This section starts with an explanation of concept drift in subsection 3.3.1, followed by the working of ADWIN in subsection 3.3.2.

3.3.1. Concept drift: data-correlation change over time

The phenomena concept drift is the term used by the Machine Learning discipline, whereas in other domains concept drift is often referred to as nonstationarity, covariate shift, or dataset shift (Žliobaitė et al., 2016). Concept drift is thus a change in the correlation between one or more features with the dependent variable.

For the electricity markets, as covered in this research, it is highly likely that the correlation between the x-values and the dependent variable will change over time. The introduction of more renewable sources of energy, changes in fuel prices and, consumer behavior all affect on the price of electricity (the dependent variable in the forecasting models covered). The introduction of new wind turbines is a gradual shift, this type of drift is called the incremental drift, as visualized in Figure 3.12. Covid-19 can be seen as an example of a sudden drift, the sudden introduction of a lockdown is likely to change the correlations previously found in the dataset. Lock-downs themselves can be seen as a reoccurring drift, as those (in the Netherlands) are only implemented for a limited period, but with a reoccurring pattern.

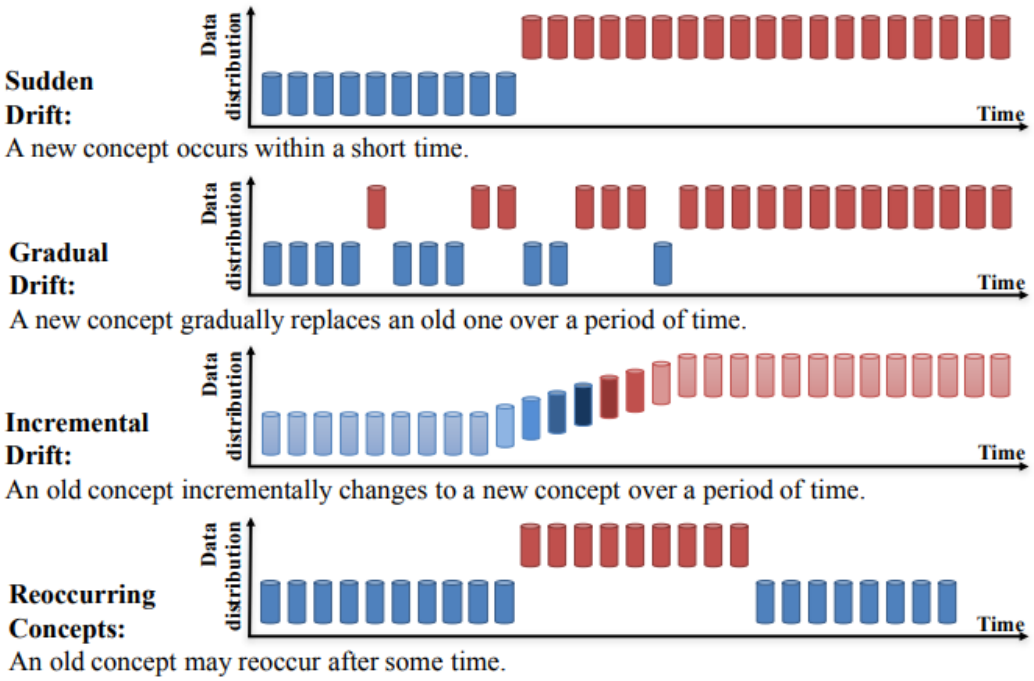


Figure 3.12: Different types of concept drifts occurring over time (Lu et al., 2018)

The obvious problem with such change is that previously trained models are starting to degrade in performance over time and need to be retrained. The options a machine learning engineer has is either regularly retrain or use a change detecting model, such as ADWIN.

3.3.2. ADWIN: Statistically dealing with concept drift

If resources or the application does not allow for a periodic retrain a change detecting algorithm can be implemented, such as the Adaptive Windowing (ADWIN) algorithm. ADWIN is a widely adopted algorithm that not only detects a change in the distribution of a dataset but also gathers a new dataset that can be used to retrain the model, such as shown in Figure 3.13. The figure shows the ADWIN algorithm tracking the model error over time. The algorithm gives a warning after drift arrives in the warning zone and gives another message when a certain drift threshold is breached.

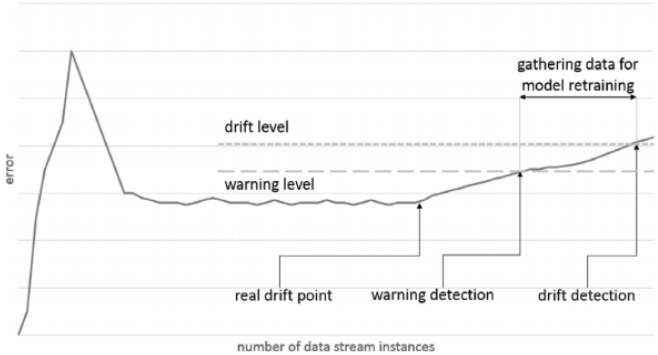


Figure 3.13: The main concept of drift detection (Krawczyk et al., 2017)

The ADWIN algorithm works by keeping a window *W*, which is split into two sub-windows W_1 and W_2. With *n* being the length of *W*. n_1 and n_2 are the length of W_1 and W_2 respectively, such that $n = n_1 + n_2$. For both W_1 and W_2 the mean $\hat{\mu}$ of the window is taken. The algorithm checks, for every possible *n* if the following holds

$$|\hat{\mu}w_1 - \hat{\mu}w_2| \geq \epsilon_{cut}, \tag{3.19}$$

where

$$\epsilon_{cut} = \sqrt{\frac{1}{2m} \ln \frac{4n}{\delta}}. \tag{3.20}$$

in the equation δ is a user defined threshold, set to 2e-3 for a datastream expected to contain gradual drift and to 0.1 for sudden drift (McKay et al., 2020). *m* is the harmonic mean of n_1 and n_2, such that

$$m = \frac{1}{\frac{1}{n_1} + \frac{1}{n_2}} \tag{3.21}$$

The concept of drift detection by ADWIN is based on the knowledge that when the mean of two large enough sub-windows differs from each other that the distribution of the two windows is not equal and thus concept drift has likely occurred (Bifet & Gavalda, 2007). Bifet and Gavalda the authors of the original paper on ADWIN, make predictions based on the adaptive mean. This will not work for the purpose proposed in this research. Therefore, only the warning system incorporated in the ADWIN algorithm is used in this study.

3.4. Interacting feature selection with the Relief algorithm

The Relief algorithm, as developed by Kira, Rendell, et al. in 1992, is an algorithm for feature selection in a space where features interact with each other. This is different compared to a large part of heuristic estimation measures for feature selection as those assume conditional independence of those features with each other (Robnik-Šikonja and Kononenko, 2003). However, for power prices, where the feature sets contain information from neighboring countries, the in-scope features are closely correlated and impact one another.

The original algorithm (Kira, Rendell, et al., 1992) is designed for binary classification problems. The algorithm has later been expanded to a multi-classification version called the ReliefF, developed by Kononenko et al. (1997). The version for regression problems is another adaption of the ReliefF algorithm called the RreliefF, developed by Robnik-Šikonja and Kononenko (1997). Besides the base relief-based algorithms (RBA), many more variants exist.

3.4.1. The original Relief algorithm

To understand the working of any RBA it is best to take a look at the working of the original algorithm by Kira, Rendell, et al. (1992). Their algorithm is considered the relief algorithm, with the purpose of feature selection for binary classification problems. All RBA are essentially feature ranking algorithms, where each feature is ranked between -1 and 1. A ranking lower than zero would mean that the feature does not carry any predictive value for the dependent variable.

Algorithm 1 The relief algorithm for Binary classification

Require: for each training instance a vector of feature values and the class value
$n \leftarrow$ number of training instances
$a \leftarrow$ number of features
$W \leftarrow$ vector of length *a* containing feature weights
Parameter: $m \leftarrow$ number of random training instances out of n used to update W

initialize all feature weights $W[a] = 0.0$

for $i \leftarrow 1, m$ **do**
  randomly select a target instance R_i
  find a nearest hit H and nearest miss M

  for $i \leftarrow 1, a$ **do**
    $W[a] = W[a] - diff(a, R_i, H)/m + diff(a, R_i, M)/m$
  end for
end for
return the vector W of feature scores that estimate the quality of features

Algorithm 1 shows the pseudo-code the relief algorithm as adapted from Urbanowicz, Meeker, et al. (2018). The algorithm requires an array from *a* features by *n* instances. The user defines *m*, the number of training instances to use out of *n* to update the feature weights. The algorithm starts by setting all the feature weights to 0, followed by a *m* number of iterations. During each iteration a target instance R_i is randomly selected. This target instance is an array with features and the corresponding label. For R_i the nearest Hit *H*, the same label, and Miss *m*, opposite label, are selected. The nearest Hit and Miss are determined by finding the smallest Euclidean distance between R_i and the other instances available in *n*. The Euclidean distance used in the original paper by Kira, Rendell, et al. (1992) is later replaced by the Manhattan distance as experiments by Robnik-Šikonja and Kononenko proved that the Manhattan distance is just as good, but reduces computational time. Finding the nearest Hit and Miss is happening using the nearest neighbor theorem, which is shown in Figure 3.14. In the figure, the star is the target R_i having class O. The nearest neighbor of the same class is the black circle, which is the nearest hit *H*. The black cross is the nearest miss *M*, as the black-X is the closest to our target instance R_i.

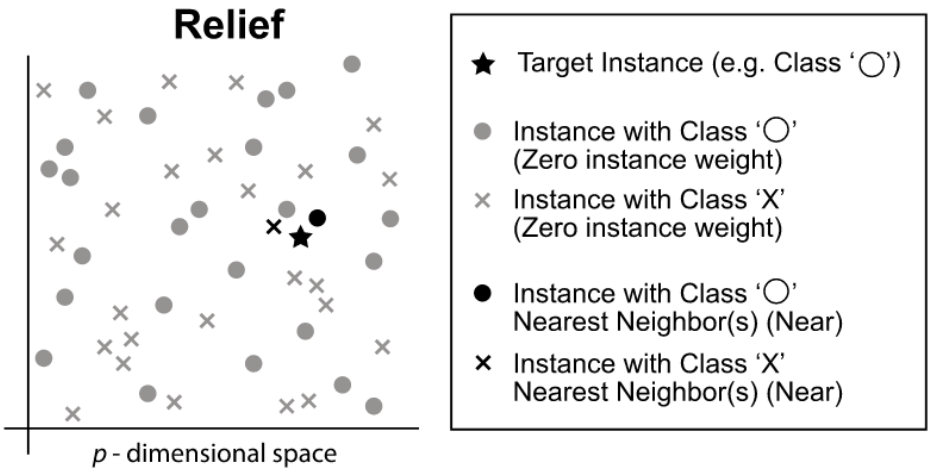


Figure 3.14: Selection of the nearest hit and miss based on the nearest neighbours (Docurbs, 2018)

After finding the nearest Hit *H* and Miss *M* the algorithm enters the weight update phase. Which consists of a for loop running over each feature *a*. For each feature, the diff function calculates a new weight which is normalized by dividing by the number of training instances *m*. The diff function contains an if-else statement

in case of discrete features such that

$$diff(A, I_1, I_2) = \begin{cases} 0, & \text{if } value(A, I_1) = value(A, I_2) \\ 1, & \text{otherwise,} \end{cases} \tag{3.22}$$

where *A* is the feature considered. If the value of *A* in R_i is the same as value of *A* in I_2 the diff function returns a 0. If the value of *A* in R_i is not equal to the value of *A* in I_2, the diff function assigns a 1. The target instance I_1 can be set equal to the target of R_i and I_2 is either the nearest hit *H* or the nearest miss *M*. The updating of the feature weights is visualized in Figure 3.15, where the 6th feature of the nearest Hit *h* is updated with a weight of $-1/m$. The 8the and 15th feature of the nearest Miss *M* are updated with a weight of $1/m$.

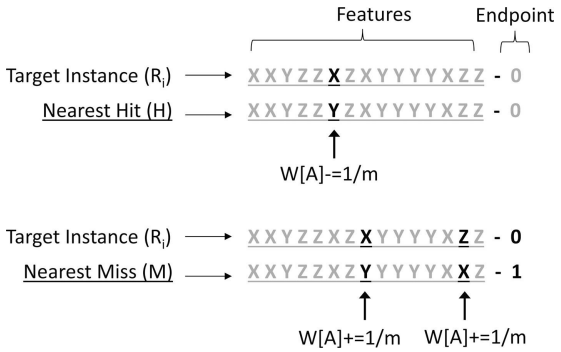


Figure 3.15: Updating feature weights according to the Relief algorithm for discrete features (Urbanowicz, Meeker, et al., 2018)

In the case of continuous features, the diff function is slightly different, shown in Equation 3.23. The main difference between Equation 3.22 and Equation 3.23 is the fact that with discrete variables, the weight is updated with a binary value, while for continuous variables a float is used between zero and one. When a subset contains both discrete and continuous variables the algorithm is likely to underestimate the continuous variables (Kononenko & Sĭkonja, 2007), which requires caution.

$$diff(A, I_1, I_2) = \frac{|value(A, I_1) - value(A, I_2)|}{max(A) - min(A)} \tag{3.23}$$

The weight update function takes the initial feature weight and subtracts the value for the nearest hit *H* while adding the value for the nearest miss *M*, meaning that features that are different in nearest misses get an increase in weight, while features that are different in a nearest hits get penalized. The idea is that a feature that can change the label of the value has a predictive value. While features that can change without affecting the label carry less predictive value. Consequently, features that obtain a negative value weight after running the Relief algorithm can be considered redundant.

3.4.2. Multi classification with ReliefF

The Relief algorithm explained in the previous chapter is only able to deal with binary classification problems. To extend the algorithm to deal with multi-classification, Kononenko (1994) came up with the ReliefF algorithm. Algorithm 2 shows the pseudo code for reliefF, as adapted from Robnik-Šikonja and Kononenko (2003).

Algorithm 2 The reliefF algorithm for multi classification

Require: for each training instance a vector of feature values and the class value
$n \leftarrow$ number of training instances
$a \leftarrow$ number of features
$W \leftarrow$ vector of length *a* containing feature weights
Parameter: $m \leftarrow$ number of random training instances out of n used to update W

initialize all feature weights $W[a] = 0.0$

for $i \leftarrow 1$, m **do**
randomly select a target instance R_i
find k nearest hits H_j
for each class $C \neq class(R_i)$ **do**
from class C find k nearest misses $M_j(C)$
end for

for $i \leftarrow 1$, a **do**
$W[a] = W[a] - \sum_{j=1}^{k} diff(a, R_i, H_j)/(m \cdot k) +$
$\sum_{C \neq class(R_i)} \left[\frac{P(C)}{1-P(class(R_i))} \sum_{j=1}^{k} diff(a, R_i, M_j(C)) \right] /(m \cdot k)$
end for
end for
return the vector W of feature scores that esimtate the quaility of features

The reliefF update introduces the ability to use a k-nearest neighbor for updating the feature weights. This deals with the criticism on the relief algorithm that the second closest hit could also carry important information. Furthermore, the essence of the reliefF algorithm remains the same, but instead of just the nearest hit *H* it searches for the *k* nearest hits H_j and the *k* nearest misses M_j. *k*, the number of neighbors to consider, is a user-defined parameter. The weight update formula is similar to the formula used in the Relief algorithm, but in this case, the contribution of all the hits and misses is weighted with the prior probability of class.

3.4.3. SURF and MultiSurf

The relief algorithm is often used in biomedical research, which is where the latest extensions for the RBA stem from. The Spatially Uniform ReliefF (SURF) developed by Greene et al. (2009) is such an example.

SURF is an extension of the reliefF algorithm but gets rid of the user-defined *k* for the number of nearest neighbors to consider. Instead, it used a fixed distance threshold *T*, which can be seen as a radius in a two-dimensional space. The distance *T* is the average distance between all instances in the training data.

The MultiSURF developed by Urbanowicz, Olson, et al. (2018) is quite similar to the SURF algorithm. However, instead of using the average distance between all instances for *T*, MultiSURF uses a pairwise distance. *T* is instance unique such that we define a T_i, where T_i is the average distance between the instance and all other instances. Furthermore, the MultiSURF algorithm also knows a dead-band zone, from T_{near} defined as $T_{near} = T_i - \sigma_I/2$, to T_i. All instances within the dead-band zone are not taken into account. This addition mainly speeds the algorithm up, as it assigns weight updates to fewer features.

3.4.4. TuRF extension

Lastly, the Tuned Relief (TuRF) extension proposed by Moore and White (2007). The TuRF algorithm is originally an extension of reliefF, but can also be applied to all other RBA. TuRF is an algorithm that deals with noisy data a known issue for reliefF. TuRF is extremely simple, as it just removes the worst scoring

features before refitting a RBA on the new subset. The pseudo-code can be summarized as following

Algorithm 3 The TuRF extension for the relief based algorithms

$n \leftarrow$ the number of loops
$q \leftarrow$ the number of features to remove in a single loop

for $i = 1, n$ **do**
    estimate a RBA
    sort feature scores based on the weight obtained by the RBA
    remove q worst scoring features
end for
return last RBA estimate for each feature

4

Trading with perfect foresight

This chapter covers the implementation of optimization algorithms used for the optimal dispatch calculations. Section 4.1 gives a general description of the battery model. Section 4.2 covers the optimization for the only the day market, followed by a combination of the day ahead and the imbalance market in Section 4.3. Lastly, Section 4.4 covers the optimization when the load of the consumers is taken into account.

4.1. General battery model

The aim of the battery model in this research is to act on different markets to find the best trading opportunities (arbitrage) while satisfying the demand and supply requirements of the consumer(s). The battery setup is schematically shown in Figure 4.1, with the battery as the middle point. The battery combined with the operational algorithm proposed in this research will take over the role of the utility company. The battery algorithm acts on the day ahead, left in the image, and the aFRR, bottom, market. Both markets have a requirement that orders placed need to be at least 1 MW. As this research covers a home battery it is therefore assumed that the market requirements will change in the future or that the battery will act in a virtual power plant (VPP) set-up. Furthermore, in this research the price taker approach is assumed, which means that the prices seen in the market used and that our market presence does not influence the prices seen.

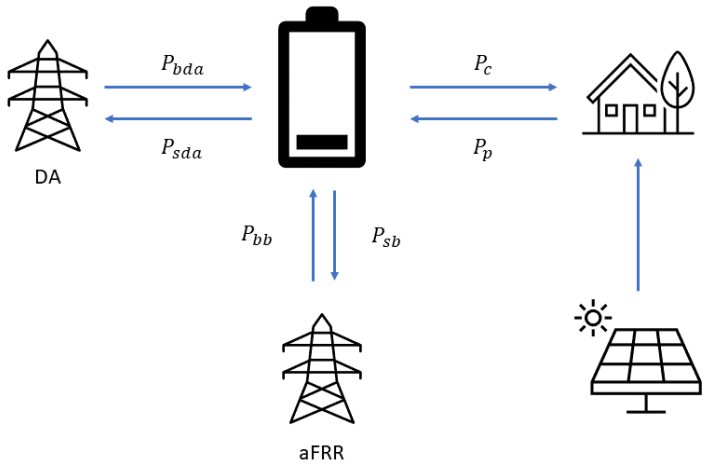


Figure 4.1: The different flows that the connect the battery to different parties. Note that the aFRR is the balancing market in question. In this study, the power flows from the consumer P_p towards the battery are not considered.

The consumer, connected on the right side, is the battery owner and thus the beneficiary of all the financial flows. The consumer could have solar PV which is self-consumed if the production is smaller than their consumption, in the case that the consumer produces more than is consumed, then the energy is delivered to the battery. The battery can decide to directly send the power onto the grid or store it for later use. Sending power onto the grid is only allowed if a position to deliver power is open on either market as the battery is not allowed to create an imbalance. Further research could cover if allowing power to be settled directly on the imbalance is advantageous for the battery owner.

The sign convention used for the battery model is that power delivered by the battery is negative and power that charges the battery is positive, such that it logically follows that energy drawn from the battery is negative and energy added is positive. This allows for summation of all power, such that the total battery power is the sum of the power of all the different segments, which needs to be within the power limits of the batter, such that the following holds:

$$-P_{rated} \leq -P_{bda} + P_{sda} - P_{bb} + P_{sb} - P_p + P_c \leq P_{rated}. \tag{4.1}$$

Not forcing all power to be settled through the battery avoids unnecessary losses and reduced available power. Consumer flows from or to the markets can be directly sent onto the grid without the intervention of the battery. Likewise, positions opened on the day ahead market can be closed on the balancing market. For example, if the algorithm bought a position on the spot market, it can sell the position on the balancing markets during an upward regulation.

4.2. Day-ahead market

First, only the day-ahead market is considered, for this and all following optimization problems, it is assumed that we are a price taker. The working of this market is covered in Section 2.3. The bidding strategy of the day-ahead market for a battery can be approached as a mixed-integer linear programming problem such that

$$
\begin{aligned}
\max \quad & \sum_{h=1}^{24} \Phi_h (P_{sda,h} - P_{bda,h}) \\
\text{s.t.} \quad & E_h = E_{h-1} + \left(P_{bda,h} * \sqrt{\eta} - \frac{P_{sda,h}}{\sqrt{\eta}} \right) * \frac{\delta}{60} \\
& 0 \leq P_{bda,h} \leq P_{rated} * \theta_{in,h} \ \forall \ h = 1, ..., 24 \\
& 0 \leq P_{sda,h} \leq P_{rated} * (1 - \theta_{in,h}) \ \forall \ h = 1, ..., 24 \\
& \bar{\sigma} \geq E_h \geq \underline{\sigma} \ \forall \ h = 1, ..., 24 \\
& E_1 = E_{24} = E_{init},
\end{aligned}
\tag{4.2}
$$

where the objective is to maximize the profit, calculated by the sum of Φ_h, the spot price at hour *h*, multiplied by $P_{sda,h}$, the power sold on the day-ahead market at hour *h*, minus the $P_{bda,h}$, the power bought at hour *h*. δ is the number of minutes in an hour necessary to convert power to energy. The battery is constrained by the energy capacity limits of the battery. E_h, the energy at hour *h*, should be larger than the lower limit, $\underline{\sigma}$, and smaller than the upper limit $\bar{\sigma}$.

E_h is updated every hour by taking the energy level at the prior hour, *h-1*, and either adding or removing the purchased/sold energy. The power is multiplied by the number of minutes activated, in this case 60 minutes, and multiplied or divided by the square root of the round-trip efficiency, η.

The power bought or sold at the day-ahead market is equal to the power delivered or received by the battery, therefore the battery power variable, used in the other models, is not used. Another notable operator in the equation is the boolean operator θ. The battery cannot charge and discharge at the same time, therefore the θ-variable is introduced. When power is bought $P_{bda-s,h} > 0 \Rightarrow P_{rated} * \theta_{in,h} > 0$, which can only be satisfied when $\theta_{in,h} = 1 \Rightarrow 0 \leq P_{sda,h} \leq 0$. Therefore, the θ-operator prevents charging and discharging at the same time.

Moreover, the battery power should always be between 0 and the rated power, P_{rated}. And lastly, the energy level at $h = 1$ should be equal to the final energy level at $h = 24$, which is based on the energy level given by the user, E_{init}. This is due to the fact that only a single day is considered by the algorithm at the time and thus has no future information. To prevent the algorithm from making decisions that impact future dispatch, it is decided to keep make sure the final energy level is equal to the starting energy level.

4.2.1. Start SoC

To find the optimal σ_0, different values for σ_0 have been tried. The result of using a different start energy in the case of perfect foresight is shown in Figure 4.2A. The result clearly shows that it is beneficial to start with an empty battery. This result can be backed-up by Figure 6.6, which shows that the morning hours are often cheaper than the evening hours. Thus, leaving the battery empty for the next morning to charge allows for a larger price delta and thus profit.

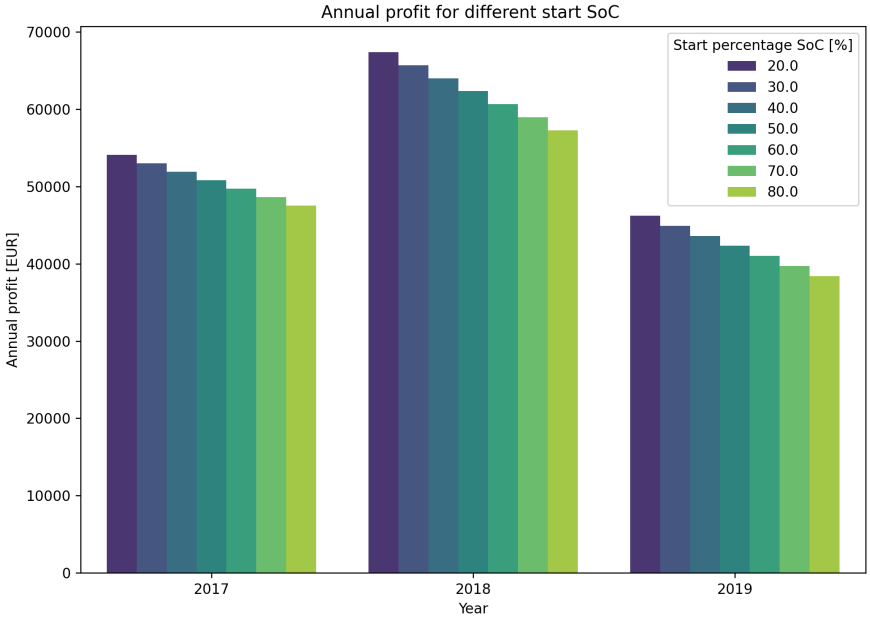


Figure 4.2: Starting the day with an empty battery is more profitable than starting fully charged.

4.2.2. Profitability check

The Rhino battery is used to test the profitability of owning a battery energy storage system (BESS) while purely operating on the spot market. The specifications for the battery are summarized in Table 4.2.2. The Rhino battery is only used to get a feeling for the numbers, the goal is not to optimize the Rhino battery.

	Power [kW]	Energy Capacity [kWh]	Efficiency [%]	CAPEX [thousand EUR]
Rhino	12000	7500	90	6,980
Tesla Powerwall 2	5	14	90	5.5

Table 4.1: Battery specification of the Rhino battery (Giga Storage B.V, 2021) and Tesla Powerwall 2 (TP2) (Tesla Inc., 2018)

The Rhino battery is likely limited to operate to a maximum state of charge (SoC) of 80% and a minimum of 20%, this would correspond to a depth of discharge (DoD) of 60%. The larger the depth-of-discharge the faster the battery degrades (PowerTech, 2021), thus it is physically possible to charge or discharge the battery past those limits. The 20%-80% limit is commonly used as the maximum operating limit. However, it is suggested that a charge above 80% is not an issue if it is discharged fast enough after (F, 2021) to below 80% again. The cost-benefit analysis of operating a battery outside the 20-80% limits is outside the scope of the research and allows for further optimization of the battery model. Therefore, the 20% lower limit and 80% upper limit are modeled as hard limits.

Furthermore, it must be noted that the degradation due to utilization of the battery is not penalized, which simplifies the model. One can imagine that if a battery has a lifetime of an n-number of cycles, every cycle should at least contribute to paying back the battery. A better approach would be to incorporate the non-linear degradation such as done by Maheshwari et al. (2020), extended by adding a penalty to the operational behavior depending on the extent of the degradation. However, building a perfect battery simulation is not within the scope of this research. This simplification of ignoring the effects of battery degradation could lead to an overestimate of the profits obtained. The battery would run more often, obtaining a lot of small profits, while in reality, it might be more beneficial to only be activated if the obtainable profit is larger. This simplification has a minor effect on the day-ahead optimization alone, as those deltas are already quite large.

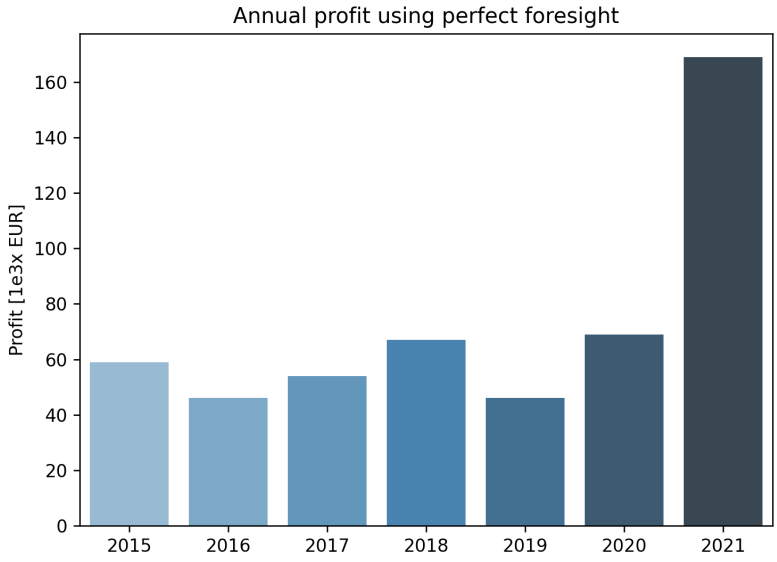


Figure 4.3: The annual profit when trading only on the day-ahead market, considering perfect sight, using the Rhino battery (12MW, 7.5MWh). The table shows deviations between years are quite large, but by far not enough to profitably operate the BESS.

At first, the optimization algorithm assumes perfect foresight of the day-ahead prices in the Netherlands. The results show that the Rhino battery generates little profit in the period between 2015-2020. The results on yearly basis are displayed in Figure 4.3. The average yearly profit comes down to 57,000 EUR/yr. Considering a capital expenditure of roughly 7,000,000 EUR, it would take around 123 years to repay the battery. Even when only considering the elevated prices of 2021 it would still take over 40 years to repay the battery.

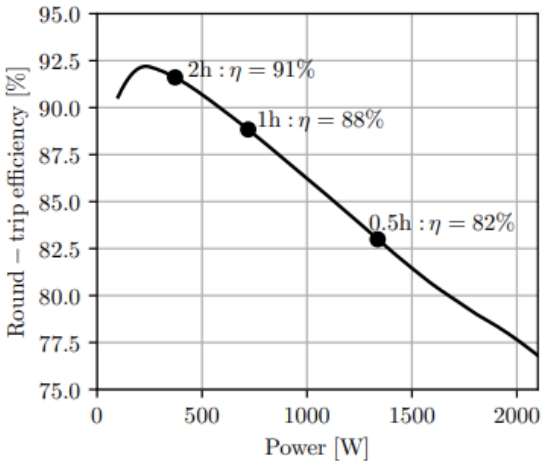


Figure 4.4: The correlation between the c-rate, expressed here in the time it takes to charge (2h corresponds to 0.5C), and the efficiency as adapted from Engels et al. (2019). The figure shows a linear decreasing correlation between the battery c-rate and the efficiency.

To find if the battery utilized by GIGA storage is the most efficient battery type for the spot market, different types of c-rates are used in the optimization function. The c-rate calculated by $textc - rate = P_{rated}/E_{rated}$ in $1/h$, thus a 2c battery is charged in $t = 1/2 = 0.5h$. The efficiency corresponding to the c-rate is obtained through a linear interpolation from the graph shown in Figure 4.4 as used in the research by Engels et al.

(2019). A battery can physically operate at lower c-rates than rated, not higher, but implementing this degree of freedom in the optimization function would over-complicate the problem by creating a non-linear constraint.

It is expected that for the spot market the most optimal c-rate is roughly around $\frac{\text{c-rate}}{\sqrt{\eta}} = 1 * \%DoD$ when for every c-rate the same efficiency is used. Which is due to the shape of the daily price curve, which somewhat has the shape of a double sine wave. That means that there are two local minima and maxima. It would be most optimal to buy during the local minimum and sell at the local maximum. Figure 4.5 shows the general price curve, the local minimums to buy would be at points 1 and 3, while selling would occur at point 4. Therefore, it would be most beneficial to fully charge at point 1 and fully discharge again at point 2, which is only possible with a 1c-rating, thus when the battery can fully (dis)charge in a single hour. In the optimization algorithm, it is assumed that half of the efficiency losses $\sqrt{\eta}$ occur directly during charging, thus that a 1 MWh battery is fully charged with 1.05 MW if the round trip efficiency is 90%. Therefore, the optimal c-rate should be around $\frac{\text{c-rate}}{\sqrt{\eta}} = 1 * \%DoD$.

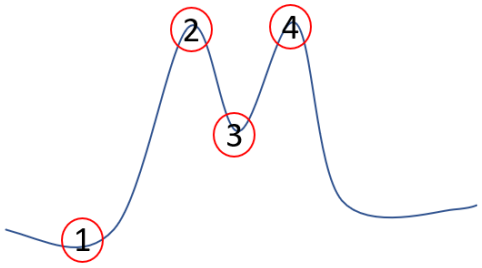


Figure 4.5: The general hourly price curve of the day-ahead market knows 2 points to buy and sell energy from a price arbitrage perspective.

Figure 4.6 shows how the different c-rates result in different profits. For the day-ahead market, the optimal c-rate is the c-rate that charges the battery in a single hour, in this case 0.6c, as the depth of discharge is placed at 60% of the total battery capacity.

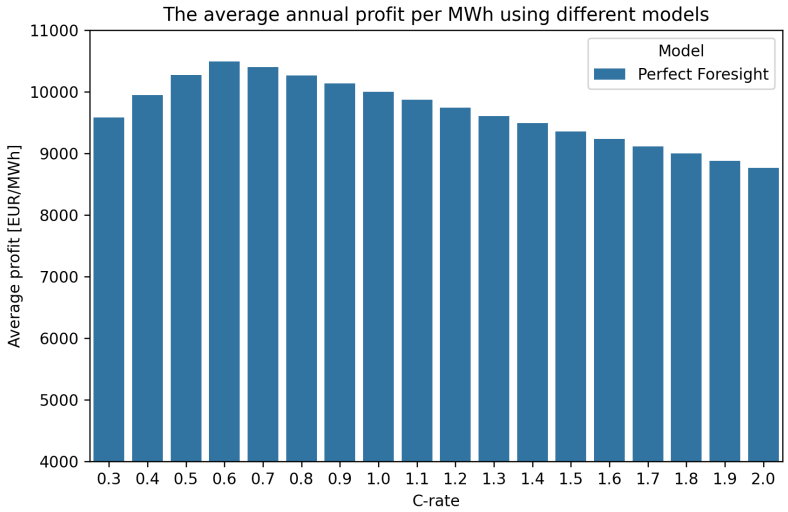


Figure 4.6: Considering a decreasing round-trip efficiency the most profitable BESS for trading on the spot market would be a BESS with a c-rate that fully charges the battery within a hour. A higher c-rate would only result in larger efficiency losses, that result in lower profits.

4.3. Day-ahead combined with the imbalance market

Next, the combination of the day-ahead and the imbalance market is considered. The imbalance market is settled every quarter of an hour in contrast to the day-ahead market which is settled per hour. The goal of this optimization is mainly to compare the final model results with the results obtained by using perfect foresight. The objective function is defined as following,

$$\max \sum_{ptu=1}^{\frac{24*60}{\delta}} \frac{\left(P_{sda,ptu} - P_{bda,ptu}\right)\Phi_{ptu} * \delta + \phi_{up,ptu} * P_{sb,ptu} * \delta_{au} - \phi_{down,ptu} * P_{bb,ptu} * \delta_{ad}}{60} \tag{4.3}$$

$$\text{s.t.} \quad E_{ptu} = E_{ptu-1} + \Delta E_{ptu} \; \forall \; ptu = 1, ..., 96 \tag{4.4}$$

$$E_0 = E_{\frac{24*60}{\delta}} = E_{init} \tag{4.5}$$

$$\overline{\sigma} \geq E_{ptu} \geq \underline{\sigma} \; \forall \; ptu = 1, ..., 96 \tag{4.6}$$

$$P_{ptu} = P_{bb,ptu} + P_{bda,ptu} - P_{sda,ptu} - P_{sb,ptu} \; \forall \; ptu = 1, ..., 96 \tag{4.7}$$

$$-P_{rated} \leq P_{ptu} \leq P_{rated} \; \forall \; ptu = 1, ..., 96 \tag{4.8}$$

$$\Delta E_{e,ptu} = \frac{P_{bb,ptu} * \delta_{ad} - P_{sb,ptu} * \delta_{au} + (P_{bda,ptu} - P_{sda,ptu}) * \delta}{60} \; \forall \; ptu = 1, ..., 96 \tag{4.9}$$

$$-P_{rated}\frac{\delta}{60}(1-\zeta) \leq E_{e,ptu} \leq P_{rated}\zeta\frac{\delta}{60} \; \forall \; ptu = 1, ..., 96 \tag{4.10}$$

$$E_{e,ptu}\sqrt{\eta} - \frac{P_{rated}}{\sqrt{\eta}}\frac{\delta}{60}(1-\zeta) \leq \Delta E_{ptu} \leq E_{e,ptu}\sqrt{\eta} + \frac{P_{rated}}{\sqrt{\eta}}\frac{\delta}{60}(1-\zeta) \; \forall \; ptu = 1, ..., 96 \tag{4.11}$$

$$\frac{E_{e,ptu}}{\sqrt{\eta}} - P_{rated}\frac{\delta}{60}\zeta \leq \Delta E_{ptu} \leq \frac{E_{e,ptu}}{\sqrt{\eta}} + P_{rated}\frac{\delta}{60}\zeta \; \forall \; ptu = 1, ..., 96 \tag{4.12}$$

$$0 \leq P_{bb,ptu} \leq 2 * P_{rated} * \theta_{b,ptu} \; \forall \; ptu = 1, ..., 96 \tag{4.13}$$

$$0 \leq P_{sb,ptu} \leq 2 * P_{rated} * (1 - \theta_{b,ptu}) \; \forall \; ptu = 1, ..., 96 \tag{4.14}$$

$$0 \leq P_{bda,ptu} \leq P_{rated} * \theta_{s,ptu} \; \forall \; ptu = 1, ..., 96 \tag{4.15}$$

$$0 \leq P_{sda,ptu} \leq P_{rated} * (1 - \theta_{s,ptu}) \; \forall \; ptu = 1, ..., 96 \tag{4.16}$$

$$\theta_{up,ptu} * P_{sb,ptu} = P_{sb,ptu} \; \forall \; ptu = 1, ..., 96 \tag{4.17}$$

$$\theta_{down,ptu} * P_{bb,ptu} = P_{bb,ptu} \; \forall \; ptu = 1, ..., 96 \tag{4.18}$$

where Φ_{ptu} is the day-ahead price at time *ptu*. $P_{bda,ptu}$ and $P_{sda,ptu}$ are the power bought and sold at the day-ahead market respectively. The power bought at the imbalance is indicated with $P_{bb,ptu}$ and power sold is $P_{sb,ptu}$. In the dual pricing balancing market it is possible that both regulation states happen during the same *ptu*, therefore a constraint in the form of $\theta_{up,ptu} * P_{bb,ptu} = 0$ does not hold.δ represents the minutes a PTU takes, δ_{au} and δ_{ad} are upward and downward number of minutes the battery is activated. In this scenario a price taker approach is assumed, thus the battery will be directly activated if the TSO sends out a set-point. Furthermore, in the simulation a 100% ramp rate is assumed, which might not be realistic set-points that TenneT would send out. This assumption could lead to a slight overestimation of the actual profits.

The constraint $\theta_{up,ptu} * P_{sb,ptu} = P_{sb,ptu}$, where $\theta_{up,ptu}$ is a binary operator, regulates that the battery can only sell during an upward regulatory state and $\theta_{down,ptu} * P_{bb,ptu} = P_{bb,ptu}$ makes sure to only buy during downward regulation. Equation 4.10 regulates the state of ζ, where ζ is a binary variable. $\Delta E_{e,ptu}$ is the energy outside the battery and before one-directional losses, where E_{ptu} is the internal energy that is already inside the battery. ΔE_{ptu} is the change in internal energy after one-directional losses, such as

shown in Figure 4.7. If $E_{e,ptu} < 0 \Rightarrow \zeta = 0$, and thus if $\Delta E_{e,ptu} \geq 0 \Rightarrow \zeta = 1$. The ζ-operator controls the value for ΔE_{ptu}, if $\zeta = 1 \Rightarrow \Delta E_{ptu} = \Delta E_{e,ptu} * \sqrt{\eta}$ and $\zeta = 0 \Rightarrow \Delta E_{ptu} = \frac{\Delta E_{e,ptu}}{\sqrt{\eta}}$. To prove this, we take $\Delta E_{e,ptu} = P_{rated} \frac{\delta}{60}$. The constraint, given by Equation 4.10, yields $P_{rated} * \frac{\delta}{60}(1-1) \leq E_{ptu} \leq (1 + P_{rated}) * \frac{\delta}{60} * 1 - 1 \Rightarrow \ 0 \leq \Delta E_{e,ptu} \leq P_{rated} * \frac{\delta}{60}$, as ζ is forced to become 1.

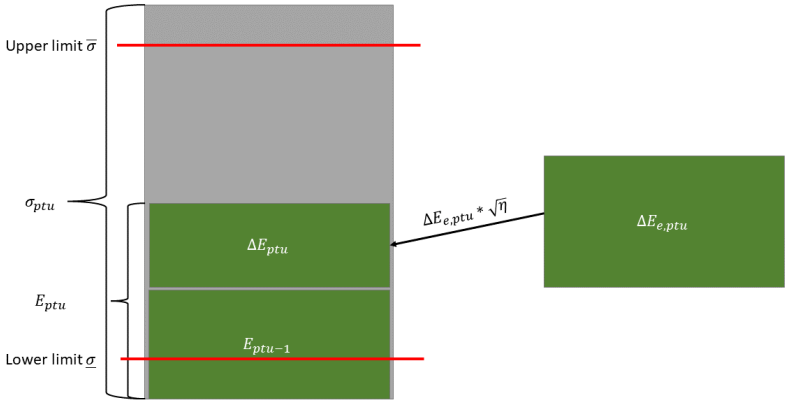


Figure 4.7: The external energy $\Delta E_{e,ptu}$ is reduced by the one-directional losses before they are added to the, in this case empty, battery. σ_{ptu} is the total available capacity, which is fixed in this case.

After obtaining $\zeta = 1$ lets consider the next constraint, given by Equation 4.11. Filling in $\zeta = 1$ and $\Delta E_{e,ptu} = P_{rated} * \frac{\delta}{60}$ results into $\left(\sqrt{\eta} * P_{rated} * \frac{\delta}{60} - 0\right) \leq \Delta E_{ptu} \leq \left(\sqrt{\eta} * P_{rated} * \frac{\delta}{60} + 0\right)$. Followed by filling in, Equation 4.12, which turns into $\left(\frac{P_{rated}}{\sqrt{\eta}} * \frac{\delta}{60} - P_{rated} * \frac{\delta}{60}\right) \leq \Delta E_{ptu} \leq \left(\frac{P_{rated}}{\sqrt{\eta}} * \frac{\delta}{60} + P_{rated} * \frac{\delta}{60}\right)$ $\Rightarrow \left(\frac{1}{\sqrt{\eta}} - 1\right) P_{rated} * \frac{\delta}{60} \leq \Delta E_{ptu} \leq \left(\frac{1}{\sqrt{\eta}} + 1\right) P_{rated} * \frac{\delta}{60}$. To satisfy both constraints $\Delta E_{ptu} = \sqrt{\eta} P_{rated} * \frac{\delta}{60} = E_{ptu} * \sqrt{\eta}$. This is the desired result, as the energy flow is inward the battery the amount of energy that actually is stored is reduced due to the losses occurring on the path inwards.

When we consider the situation where $\Delta E_{e,ptu} = -P_{rated} * \frac{\delta}{60}$, we obtain $-P_{rated} * \frac{\delta}{60}(1-0) \leq -\Delta E_{e,ptu} \leq -1$ by filling in Equation 4.10, where $\zeta = 0$ to satisfy the constraint. Applying $\zeta = 0$ to Equation 4.11 results into $\left(\sqrt{\eta} * -P_{rated} * \frac{\delta}{60} - \frac{P_{rated}}{\sqrt{\eta}} * \frac{\delta}{60}(1-0)\right) \leq \Delta E_{ptu} \leq \left(\sqrt{\eta} * -P_{rated} * \frac{\delta}{60} + \frac{P_{rated}}{\sqrt{\eta}} * \frac{\delta}{60}(1-0)\right) \Rightarrow$ $-P_{rated}\left(\sqrt{\eta} + \frac{1}{\sqrt{\eta}}\right) * \frac{\delta}{60} \leq \Delta E_{ptu} \leq -P_{rated}\left(\sqrt{\eta} - \frac{1}{\sqrt{\eta}}\right) * \frac{\delta}{60}$. As $\eta \in \{0,1\}$, the right hand side will be in the range [0, ∞). Applying the same to Equation 4.12 results into $\left(\frac{-P_{rated}}{\sqrt{\eta}} * \frac{\delta}{60} - 0\right) \leq \Delta\sigma \leq \left(\frac{-P_{rated}}{\sqrt{\eta}} * \frac{\delta}{60} + 0\right)$, forcing ΔE_{ptu} to become $-P_{rated}/\sqrt{\eta} * \frac{\delta}{60} \Rightarrow \Delta E_{ptu} = \Delta E_{e,ptu}/\sqrt{\eta}$, the energy is thus taken out of the battery.

The different ΔE_{ptu} are necessary to accurately model the energy capacity during every *ptu* inside the battery, i.e. if 1MW is sold for a certain *ptu*, the battery needs to drain 1 MW$/\sqrt{0.9} * 0.25 = 0.26$ MWh (considering $\eta = 0.9$ and $\delta = 15$) in order to satisfy the losses inside the battery.

In this optimization, the spot market is first optimized by using Equation 4.2 and the result of the optimization is used in Equation 4.3. The sequential pattern is chosen because the imbalance is a result of a mismatch between the day-ahead market and the actual demand/production. Hypothetically speaking, it would not be possible to know the imbalance prices at the moment of the spot yet, as that would result in an arbitrage. Furthermore, allowing parallel optimization of the imbalance and the day-ahead would likely result in the algorithm buying infinite power at the day-ahead and selling at the same hour on the imbalance or vice-versa due to arbitrage possibilities.

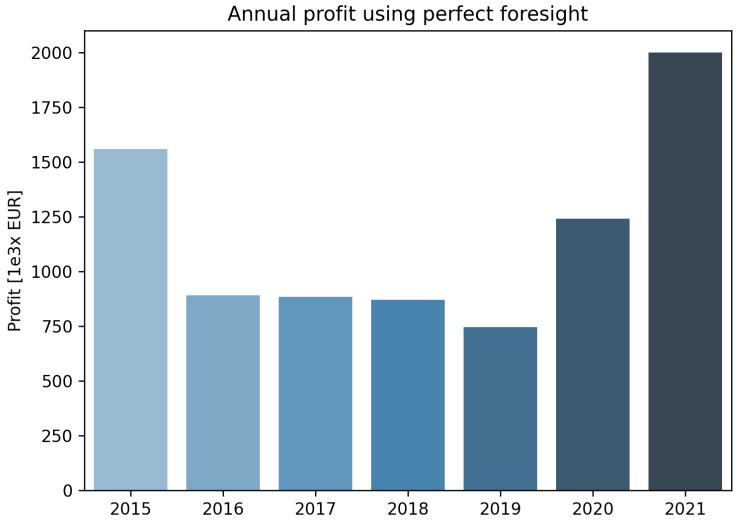


Figure 4.8: The annual profit when trading first on the day-ahead market followed by trading on the balancing market, considering perfect sight. The Rhino battery specs (12MW, 7.5MWh) are used . The graph shows that the combination of balance market and day ahead market resulted in a profit large enough to pay-back a BESS in a timely manner, when perfect foresight is considered.

The results from this optimization, again by using the Rhino battery, are shown in Figure 4.8. The first thing worth noting is the roughly 15-fold in profits compared to only trading on the day-ahead market. Annual profits of roughly a million could create a business case for a BESS costing roughly 7 million. Second, it is interesting to note that between 2016 and 2019 nearly the same profit is realized, which is different compared to the day-ahead market as shown in Figure 4.3. This could indicate that there is barely any change in the merit order for the aFRR in those years, combined with the same events leading to imbalances. To investigate the cause more research is needed. Another interesting observation is the fact that in 2021 the profit for just trading on the day-ahead more than doubled compared with 2020, which does not happen for the aFRR market. The profits do increase, but with around 60% compared to the previous year. This could indicate that the price deltas on the day-ahead have increased more than the price deltas on the aFRR market or that the volatility on the imbalance has been reduced. Again, more research in needed to investigate the cause of this.

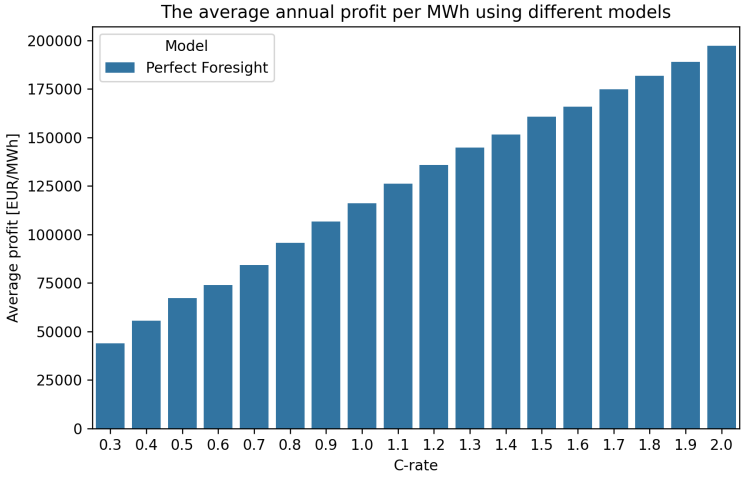


Figure 4.9: The obtainable profit increases steadily with the c-rate.

Figure 4.9 shows the average annual profit for different c-rates. Unsurprisingly, the profit increases with an increase in c-rate. The battery benefits most from a large amount of power if the prices are known up-front. The battery would have activated the maximum amount of power during extreme price events. Besides, the aFRR has typically more small deviations during the day, which, if it was perfectly known how the prices will evolve, would allow for infinite trading on the smaller deviations. Therefore, the results obtained with perfect foresight can never be obtained when applied to a real-world scenario.

4.4. Trading on the day-ahead market and considering the load

So far the consumer has not yet been taken into account, even though it has a crucial part to play in the final model. Therefore, in this section, the consumer is added to the optimization problem introduced in Section 4.2. The relevant consumer information regarding the optimization problem are the consumer's energy demand, $E_{c,h}$, and production, $E_{p,h}$. Those are added to/subtracted from the battery energy at every hour *h*, resulting in the following optimization problem:

$$
\begin{aligned}
\max \quad & \sum_{h=1}^{24} \Phi_h (P_{sda,h} - P_{bda,h}) \\
\text{s.t.} \quad & E_h = E_{h-1} + \Delta E_h \\
& \overline{\sigma} \geq E_h \geq \underline{\sigma} \\
& \\
& P_h = P_{pmax,h} + P_{bda,h} - \left(P_{sda,h} + P_{cmax,h}\right) \ \forall \ h = 1, ..., 24 \\
& -P_{rated} \leq P_h \leq P_{rated} \ \forall \ h = 1, ..., 24 \\
& \Delta E_{e,h} = E_{p,} + P_{bda,h} * \frac{\delta}{60} - \left(P_{sda,h} * \frac{\delta}{60} + E_{c,h}\right) \ \forall \ h = 1, ..., 24 \\
& \\
& -P_{rated}(1-\zeta) * \frac{\delta}{60} \leq \Delta E_{e,h} \leq P_{rated} * \frac{\delta}{60}\zeta \ \forall \ h = 1, ..., 24 \\
& \sqrt{\eta} * \Delta E_{e,h} - \frac{P_{rated}}{\sqrt{\eta}}(1-\zeta) * \frac{\delta}{60} \leq \Delta E_h \leq \sqrt{\eta} * \Delta E_{e,h} + \frac{P_{rated}}{\sqrt{\eta}}(1-\zeta) * \frac{\delta}{60} \ \forall \ h = 1, ..., 24 \\
& \frac{\Delta E_{e,h}}{\sqrt{\eta}} - P_{rated}\zeta * \frac{\delta}{60} \leq \Delta E_h \leq \frac{\Delta E_{e,h}}{\sqrt{\eta}} + P_{rated}\zeta * \frac{\delta}{60} \ \forall \ h = 1, ..., 24 \\
& \\
& 0 \leq P_{bda,h} \leq M * \theta_h \ \forall \ h = 1, ..., 24 \\
& 0 \leq P_{sda,h} \leq M * (1 - \theta_h) \ \forall \ h = 1, ..., 24 \\
& \sigma_0 = \sigma_{24} = \sigma_{init}
\end{aligned}
\tag{4.19}
$$

Here $E_{c,h}$ is the energy consumption of the users, that could not be satisfied with their production (such as solar PV), during hour *h* in kWh. $E_{p,h}$ is the production which is 0 if the user consumes the produced power themselves. $P_{pmax,h}$ and $P_{cmax,h}$ are the maximum power production and consumption seen in the hour, which is thus the maximum power that the battery needs to satisfy for the consumer. *M* is a randomly large number only necessary to prevent dual charging. There is no additional term necessary in the optimization function as the optimal purchase point is during the lowest point of the day, which is already satisfied with the $P_{bda,h}$. Likewise for the best sell point which is satisfied with the $P_{sda,h}$.

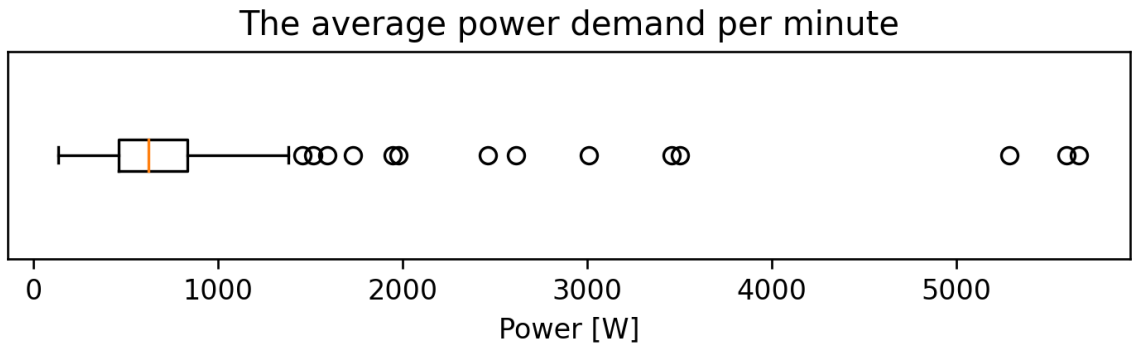
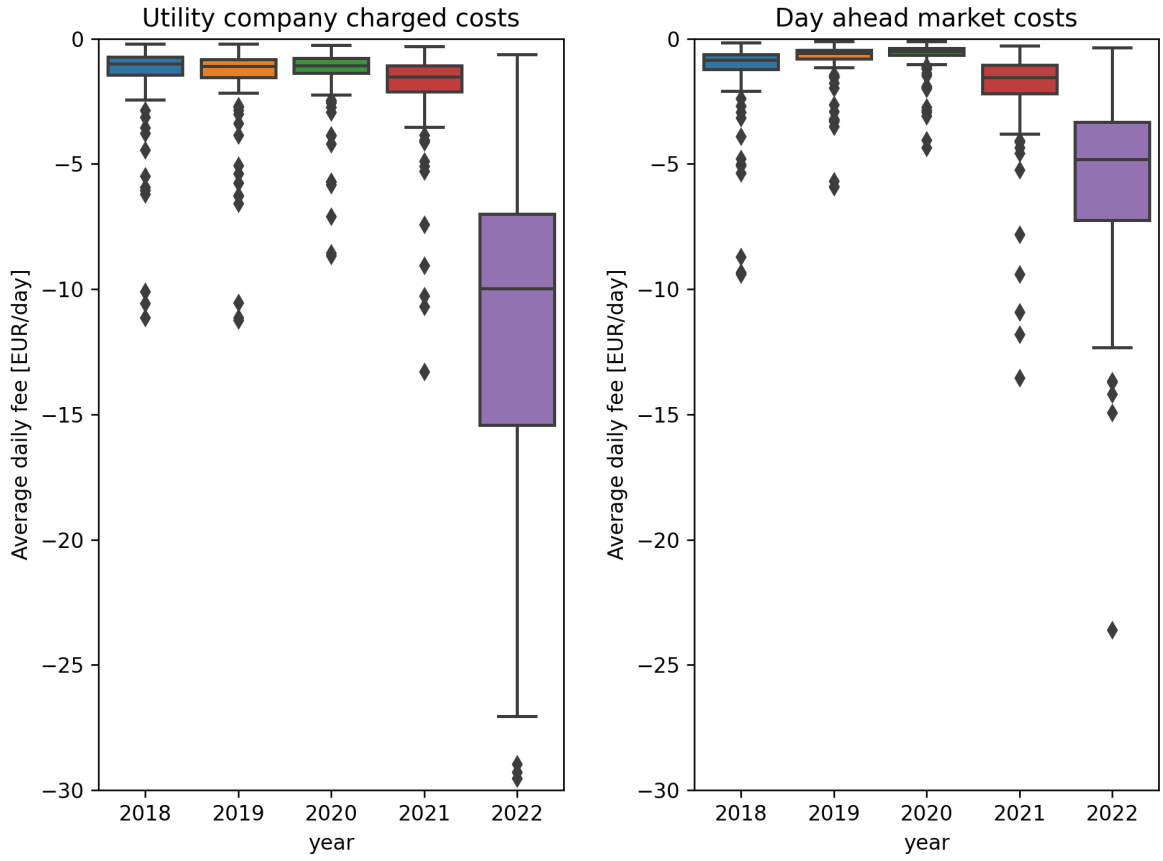


Figure 4.10: The average power per minute for the 100 consumers in scope

For this experiment a Dutch utility company provided data for a group of 100 consumers on individual level to run simulations on. Unfortunately, the data provided did contain redelivery to the grid, therefore it is assumed that all redelivered energy is directly put onto the grid without the intervention of the battery and no penalty is obtained for that. The average power per minute is shown in Figure 4.10. The figure shows that a couple of consumers have relatively high power consumption, those households would require more than one Tesla Powerwall 2 (TP2) in order to benefit from the storage ability.



(a) Utility company charged fee

(b) day-ahead market fee

Figure 4.11: The distribution shows the average daily costs for a consumer out of the group of 100 consumers.

At first, an assessment is made of the current costs of the households. The electricity fees on averagely charged by utility companies are obtained from Statistics Netherlands (2022) for 2018-2021. The data for 2022 was not yet available, thus a quote from Essent (2022) has been used for the price for 2022. The daily average cost of electricity charged to the individual consumers by the utility company, based on a single tariff, is shown in Figure 4.11A. When the charged fees are compared to the price that would have been paid on the day-ahead market instead, shown in Figure 4.11B, we can observe that from 2019 and onward it is generally cheaper to buy the electricity directly on the day-ahead market instead of signing a contract with a utility company. However, this would assume that the consumer would be able to consume the energy with constant power, as the day-ahead market is a power market.

A battery can work as a buffer as it can spread the consumed power such that a constant power signal is sent to the grid. This would require a limitation on the maximum power consumption of the consumer. Therefore the maximum power of the consumer is limited to the maximum power their batteries can deliver. The amount of batteries assigned to a single household is calculated based on the daily average consumption. The number of batteries should be sufficient enough to, on average, completely satisfy the demand when considering a 100% DoD. For example, if someone consumes on average 25kWh per day, then they would be assigned two TP2s. It must be noted that in the Netherlands the largest consumer is on average a household of 4 or more living in an old house, who consumed 12.5 kWh/day (Milieu Centraal, 2022).

Based on the above-mentioned assigned method for batteries, the group of 100 consumers would require a total of 206 TP2, this is roughly 30% more than the required number of batteries when the group is aggregated. This is due to that multiple consumer that only use 5 kWh a day can share the capacity of a single TP2 when aggregated, while if used individually, would all require their own storage.

The battery performance is compared for the two groups by calculating the battery attributed trading profit and loss (PnL). This can be calculated by taking the costs of the consumed electricity if fully bought on the day-ahead market, minus the PnL obtained by the battery. This number is averaged over the number of trading days and divided by the amount of installed capacity in kWh. Figure 4.12 shows the difference between the two groups and the average daily PnL per kWh. It shows that the grouped battery performs slightly better than all the single users, which can likely be attributed to the fact that less power needs to be reserved during a whole hour to capture the peak demand and the fact that less batteries are needed. Therefore, we can conclude that the battery can be operated in a more efficient way.

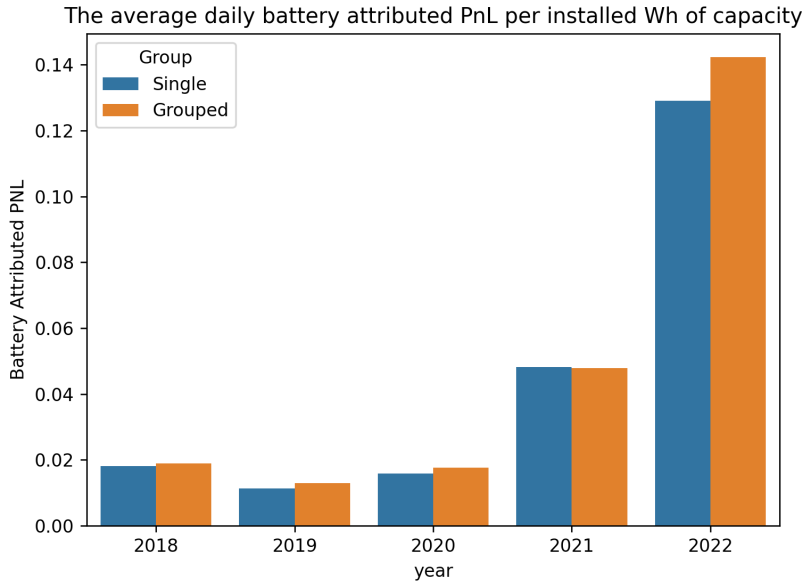


Figure 4.12: The daily trading PnL for using a TP2 is around 0.30EUR/day on average, and increased to 2EUR/day in 2022.

The average daily benefit of owning a battery is shown in Figure 4.13. The benefit takes the advantage that comes from not having a contract with a utility company into account and can be calculated by taking the demand costs, as charged by the utility company and adding the battery PnL on top of it. It can be seen that up and till 2021 the battery benefit is on average 0.04 EUR/day/Installed kWh for a single consumer, which translates to 0.55EUR/day per TP2. This equals 200EUR/yr, for which we can conclude that the 5500 EUR costing TP2 would take 27.5 years to be paid back. Even though this is still longer than the life expectancy of a battery, it is a lot better than the 123 years the Rhino battery would take, as covered in subsection 4.2.2. Furthermore, the large spread between the day-ahead price and the fee charged by the Utility company makes 2022 more lucrative. In this case, a TP2 would earn/save 2000 EUR/yr, already paying back the initial investment in less than 3 years.

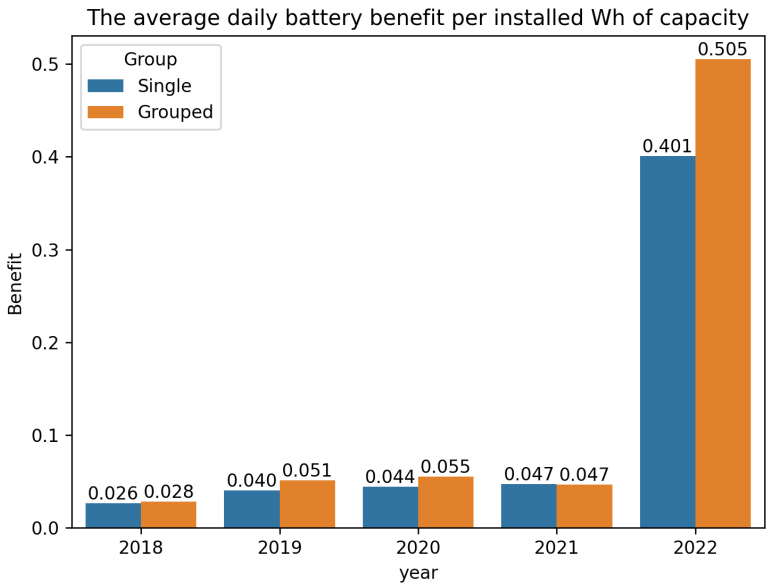


Figure 4.13: The daily average benefit for using a TP2 is around 0.70EUR/day on average, but sharply increases in 2022.

It must be noted that it cannot be concluded that utility companies are overcharging their consumers as they generally do not use the day-ahead market to settle the consumer demand. This is often done through long-term contracts as covered in Section 2.2. Besides, the utility company also makes costs covering the imbalances generated by the consumers.

5

Trading on uncertainty

This chapter covers the model set-up. Section 5.1 covers the initial battery set-up applied within this research. Section 5.2 discusses the operational difficulties and Section 5.3 describes the techniques used to bid the battery in on the aFRR market and the day ahead market. Lastly Section 5.4 gives a description of the machine learning framework used to make different market forecasts.

5.1. Battery operating model

The combination of trading on imbalance and day-ahead markets with consumer demand satisfaction introduces the risk that, due to forecasting errors, the battery cannot meet its obligations. To mitigate those risks, the energy capacity of the battery is split into two, as shown in Figure 5.1. The green part of the battery serves the consumer by opening positions on the day ahead market by both buying and selling power. All consumer demand and supply are settled within this part of the battery. The second part of the battery is dedicated to serving the imbalance market. The capacity dedicated to each part of the battery is not fixed but instead can change daily based on forecasted demand. The battery segments both have their safety limits to mitigate forecast uncertainty, those limits could be optimized based on a risk assessment which would require further research. In this study, those limits are set at 20% and 80% of the respective capacity, which corresponds to the common limits of a lithium-ion battery.

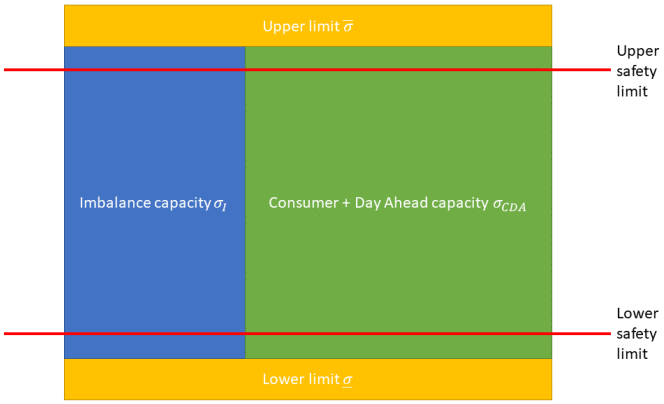


Figure 5.1: The battery is split into two, the green area is the capacity reserved for the consumer and the day ahead market, while the blue area is the capacity reserved for trading on the imbalance market.

Separating the battery into two is similar to having two separate batteries, but with a constantly changing capacity and power based on forecasted needs. Since the goal of the model is to minimize nuisance for the battery owner, meeting the energy demand of the consumer is the most important task of the algorithm. Therefore, a safety mechanism is created, such that there is always energy available to meet consumer demand. This mechanism is designed as follows. When the energy for the consumer and day-ahead market E_{CDA} in σ_{CDA} falls below the lower safety limit, energy is reallocated from the imbalance to the CDA part of the battery. Likewise, energy is reallocated from the CDA part if E_{CDA} reaches above the upper safety limit, to the imbalance part. This is shown in Figure 5.2 with E_{ACL} being the amount of energy that is above the upper safety limit which is relocated to the imbalance part. E_{UCL} is the energy that is under the safety limit at the consumer side.

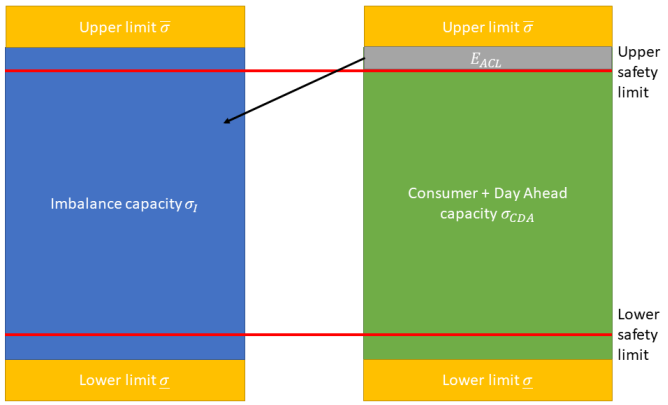


Figure 5.2: A over capacity in the CDA-part of the battery is solved by subtracting the over-capacity on the CDA-side and added to the imbalance side. The same happens for an under-capacity, but vice versa.

However, it must be noted that no physical flows of energy take place between the two parts, energy is only relabeled, which affects the behavior algorithm. In particular, adding or removing energy to the imbalance capacity allows for either more or fewer imbalance operations, which is decided based on the state of the imbalance capacity.

Energy will always be moved even when the safety limits of the imbalance do not allow for it. However, the upper- and lower limits $\overline{\sigma}$ and $\underline{\sigma}$ will never be violated as a result of this relabeling, because the $\overline{\sigma}$ and $\underline{\sigma}$ are shared limits between the two parts. And the relabeling does not affect the total energy in the battery.

On a daily basis, the consumer demand and supply electricity is settled on the spot market. The consumer behaviour is unknown in advance of the market settlement, which is required in order to place optimal bids. Commonly, utility companies use a standardized curve for a large group of consumers (L. Groot-Haar, personal communication, 2021), in order to meet their short-term demand. This approach would only work if the group is large enough to even out all the individual outliers.

However, this battery should also work for a single consumer or a relatively small group of consumers as the research focuses on individual versus neighborhood owned batteries. Therefore, a custom demand forecast, tuned on the consumer base, will be used in the final algorithm.

Moreover, the battery operating algorithm can only make a considered decision on the optimal purchase and sell moments if, next to the demand, the prices of the day-ahead market are known. This is obviously not the case, therefore the day ahead prices need to be forecasted.

The combination of the demand forecast and the day ahead price forecast are used in the optimal dispatch function as described in Equation 4.19. The result of the dispatch optimization leads to the moments on which the battery will either sell or buy electricity, based on capacity and power limits. The demand and

day ahead price forecasts are generated daily at eleven o'clock, half an hour before the bids have to placed on the market, shown in green in Figure 5.3.

Figure 5.3: The events happening on a daily basis by the battery algorithm. Ranging from quarterly forecasts and bidding on the aFFR, to the daily monitoring of the errors and forecasting and bidding for the DA market.

The *CDA* capacity, as shown in Figure 5.1, is based on the demand forecast. The consumer capacity is chosen to be twice the daily maximum forecasted hourly energy consumption, which is done after the new forecasts are made at 11:35. The re-balancing of the safety limits won't happen until the first hour of delivery, which happens at midnight, both shown in orange in Figure 5.3.

Besides the daily forecast for the consumer demand and day ahead prices, the model also forecasts the regulatory state of the imbalance market. The state forecast is given in a probability if the state will be upward regulatory. Furthermore, a probability is forecasted if the upward price will be above the rolling weekly median upward price. This is an indication if the price in the quarter is beneficial to place a bid. Likewise, the probability if the next downward price is beneath the rolling weekly median downward price.

Both the state and the down- and upward price probability reduce the amount of bids that are placed that will not be fruitful. Which is especially beneficial if the available energy capacity is smaller than the available capacity based on the available power.

5.2. Operation difficulties

As stated before, the algorithm is designed to always have enough energy to satisfy the demand of the consumer. Therefore, power needs to be purchased in advance of the consumption, with a guarantee of delivery. Since the imbalance markets do not provide a guarantee of delivery, the required energy for consumption is bought on the day ahead market. The day-ahead market closes for bidding at 12:00 CET, meaning that the forecast for both the day-ahead price and the demand from the consumer needs to be made 12h in advance for the whole next day.

One option is to fully dedicate the battery fully to the day ahead market. However, the results in chapter 4, show that only using the day ahead market for trading activities does not generate enough profit to pay back the initial purchase price of the battery, even considering perfect foresight. The same chapter shows that extending the activities to the imbalance market does result in a significant increase in profits when considering perfect foresight. This is in line with the finding of Baumgarte et al. (2020), who show that multiple business models are required to be profitable.

Therefore, the battery is also used on the imbalance market, more specifically, the aFRR market. As covered in chapter 2, the aFRR knows two structures, the voluntary and contractual. The difficulty from a battery perspective is the limited available capacity. Namely, a 1 MWh battery, can fully charge in one hour at 1 MW, when not considering the depth of discharge. However, the contractual aFRR is currently contracted in 24h contracts, meaning that the 1 MWh battery could only bid 1/24 MW for a downward regulation, to take into account the possibility that the whole day is downward regulated. Any other bid higher than

Algorithm 4 The operation algorithm

Require: for each training instance a vector of feature values and the class value
$PTU \leftarrow$ number of ptus in a hour
$H \leftarrow$ number of hours in a day
$D \leftarrow$ number of days in the simulation

for $d \leftarrow 1$, D **do**
Allocate all energy E_{CDA} to imbalance capacity
Redefine capacity limits for Imbalance, $\overline{\sigma}_I$ and $\underline{\sigma}_I$ and CDA, $\overline{\sigma}_{CDA}$ and $\underline{\sigma}_{CDA}$
$MSE_c \leftarrow$ previous day consumer error
Add MSE_c to $ADWIN_c$
if $ADWIN_C \geq \epsilon_{cut}$ **then**
Retrain consumer forecast model
end if
for $h \leftarrow 1$, H **do**
if $h = 11$ **then**
Forecast next day ahead prices
Forecast next day consumption
Run the dispatch optimization algorithm
Place Bids on the day ahead market
end if
if $h == 13$ **then**
$MSE_{da} \leftarrow$ day ahead error
Add MSE_{da} to $ADWIN_{da}$
if $ADWIN_{da} \geq \epsilon_{cut}$ **then**
Retrain day ahead forecast model
end if
end if
for $ptu \leftarrow 1$, PTU **do**
Forecast the next three balancing states
Forecast if next upward or downward price will be above/below the 20% adjusted median
Place aFRR bids
Update E_{CDA} by the consumer demand
Update E_{CDA} by the bought/sold day-ahead power
Update E_I by activated aFRR energy
if $E_{CDA} > \overline{\sigma}_{CDA}$ **then**
$E_{ACL} \leftarrow E_{CDA} - \overline{\sigma}_{CDA}$
$E_{CDA} \leftarrow E_{CDA} - E_{ACL}$
$E_I \leftarrow E_I + E_{ACL}$
end if
if $E_{CDA} < \underline{\sigma}_{CDA}$ **then**
$E_{UCL} \leftarrow \underline{\sigma}_{CDA} - E_{CDA}$
$E_{CDA} \leftarrow E_{CDA} + E_{UCL}$
$E_I \leftarrow E_I - E_{UCL}$
end if
end for
end for
AUC-PRC$_{upward} \leftarrow$ daily average upward area under the precision-recall curve (AUC-prc)
AUC-PRC$_{downward} \leftarrow$ daily average downward AUC-prc
$Accuracy_{state} \leftarrow$ daily average state forecast accuracy
Add AUC-PRC$_{upward}$ to $ADWIN_{upward}$
Add AUC-PRC$_{downward}$ to $ADWIN_{downward}$
Add $Accuracy_{state}$ to $ADWIN_{state}$
if $ADWIN_{upward} \geq \epsilon_{cut}$ **then**
Retrain upward price forecasting model
end if
if $ADWIN_{downward} \geq \epsilon_{cut}$ **then**
Retrain downward price forecasting model
end if
if $ADWIN_{state} \geq \epsilon_{cut}$ **then**
Retrain state forecasting model
end if
end for
return the vector W of feature scores that estimate the quality of features

the 1/24 MW for a downward regulation come at the risk of non-compliance, if the whole day is downward regulated. Noncompliance could lead to a fee or even expulsion from bidding on the aFRR market by the TSO. Since the voluntary aFRR is more flexible in this regard this study makes use of the voluntary aFRR.

On the voluntary aFRR market, orders need to be placed only half an hour in advance, but it is unknown if and how long the battery is activated until the end of the quarter for which the bid was placed. This uncertainty still requires reservation of capacity, which restricts the available capacity to bid in. For example, consider the prices observed in Figure 5.4 and a 1 MW/1 MWh battery, with half an hour before the upward price of 500 EUR/MWh an available upward capacity of 200 kWh. 200 kWh capacity means that 2 times a bid of 0.4 MW can be placed or 0.8 MW once, assuming that every placed bid is activated for the whole quarter, which would empty the battery completely. It is most beneficial to place the 0.8 MW order during the 500 EUR/MWh, knowing that the price in the next quarter is going to be lower than 500 EUR/MWh. However, that would require foresight of the price development, which is not the case. Moreover, bidding either 0.4 MW twice or 0.8 MW once would mean that there is no capacity left to bid anything in for the PTU that starts after that, because capacity needs to be reserved in case of activation of the bids during the whole period. However, if one or more of the bids did not, or not fully, activate the battery, then bidding no capacity has been a loss of opportunity.

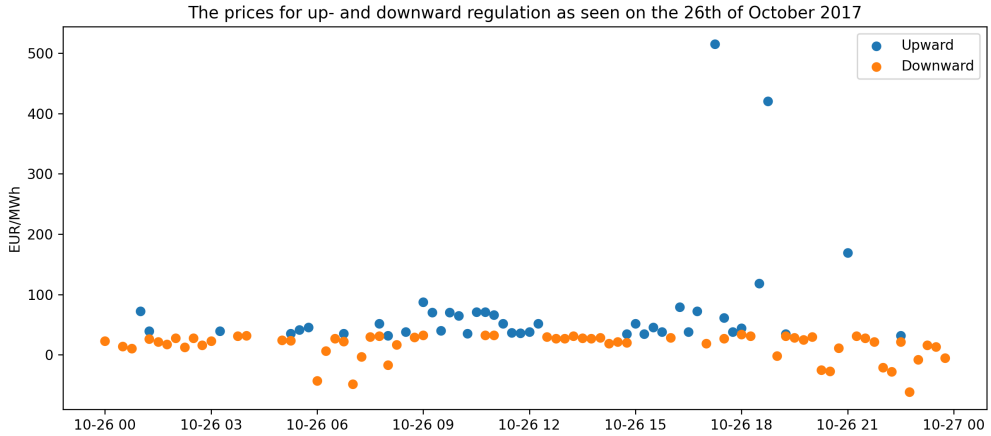


Figure 5.4: A daily pattern of the different regulation states in combination with the observed prices. Some extreme prices can be observed.

5.3. Battery bidding

The algorithm in charge of operating the battery is charged with placing bids on the aFFR imbalance market. This comprises of two tasks, namely placing buy and sell orders, which contain a price and volume. A strategy of just placing bids on extreme prices will not work. When deactivating the activated aFRR, the TSO will start with the most extreme prices first and deactivates everything in the reverse merit order (M. Veenman, personal communication, 2021). Since, the price paid/received is based on the activated energy, it is desirable to be activated as long as possible during an extreme price. Because less extreme bids are activated first, bidding less extreme is likely to increase the time the power plant can produce/consume energy. Therefore, there is a trade-off between revenue per time unit activated, and total time activated. The optimal bid/ask price seeks to balance these two.

The bid price is determined as follows:

$$\gamma_{buy,t+h} = M\vec{\phi}_{up,t,t+96} * 1.2, \tag{5.1}$$

where $\vec{\phi}$ is a vector containing the historical upward prices. The bid price placed by the algorithm uses the median M of the past 96 upward prices, in order to balance revenue per activated unit of time and total time activated. As the market trades in quarter hours, the 96 prices are corresponding to the prices seen in

a daily window. This range is long enough to capture an accurate price, while being short enough to still deal with market changes affecting the bids placed by other parties. An example of an external daily effect altering the imbalance prices is the price of natural gas. Gas turbines are flexible enough to be utilized to deal with imbalances and are thus operated on the imbalance markets. The median price is increased by an arbitrary 20% in order to make sure that the battery is activated for more minutes, as it would place the bid higher on the merit order.

The ask price used is at least the marginal cost of the battery, which are the efficiency losses, but not less than the median upward price, such that the ask price is

$$\gamma_{sell,t+h} = max\left(\frac{\mu_{t,buy}}{\eta}, M\vec{\phi}_{up,t,t+96}\right), \tag{5.2}$$

where $\mu_{t,buy}$ is the weighted average purchase price of all energy bought at the aFRR and still in the battery at time *t*. By taking the maximum of $\mu_{t,buy}$ and the median of the historical prices, the ask price is always at least equal to the marginal costs, but if the current ask-prices seen in the market are higher those prices will be used instead. This prevents from under-pricing the electricity compared to market peers.

5.3.1. Naive model

Figure 5.5 shows a timeline to illustrate how t is defined. If an order is placed for 01:30 to 01:45, under the current regulation, the order needs to be sent to the TSO before 01:00. Therefore, E_t is measured at 00:59, $P_{up,\tilde{t}=t}$ is the power bid in for the PTU that runs from 01:00 up till 01:15.

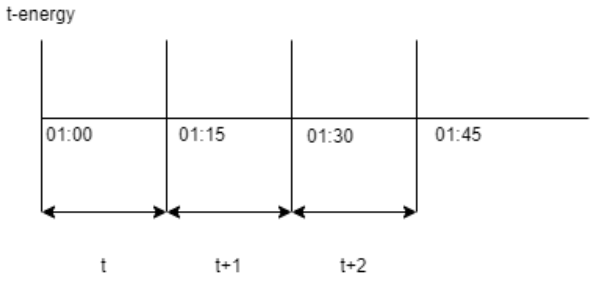


Figure 5.5: The timeline indicating that E_t is measured right before the start of the PTU that runs from 01:00 to 01:15, and that *t* covers the duration of the PTU.

The volume for the order needs to be determined, which is, as a baseline, done by a naive model. This model places an order for the power based on the available energy, such that:

$$P_{up,t+h} = max\left(\left(E_{I,t} - \underline{\sigma}_{I,t} - \sum_{\tilde{t}=t}^{t+h-1} \frac{P_{up,\tilde{t}} * \delta}{60\sqrt{\eta}}\right) * \frac{60}{\delta} - P_{pcd,t+h} - P_{bda,t+h} + P_{sda,t+h}, P_{max}\right) \tag{5.3}$$

and

$$P_{down,t+h} = max\left(\left(\overline{\sigma}_{I,t} - E_{I,t} - \sum_{\tilde{t}=t}^{t+h-1} \frac{P_{down,\tilde{t}} * \delta * \sqrt{\eta}}{60}\right) * \frac{60}{\delta} + P_{pcd,t+h} + P_{bda,t+h} - P_{sda,t+h}, P_{max}\right). \tag{5.4}$$

E_t is the current energy in the imbalance part of the battery, and $\underline{\sigma}_{I,t}$ and $\overline{\sigma}_{I,t}$ are the minimum and maximum allowed energy in the imbalance part of the battery at time *t*, respectively. $P_{up,t+h}$ and $P_{down,t+h}$ are the power bids in *h* PTUs ahead. These power bids are multiplied by δ, which is the number of minutes in a single PTU. It is chosen to use the maximum number of minutes in a PTU because it is unknown if and how long the battery will be activated. It is possible that this is not an optimal choice, and further

research could consider including forecasts of the length of the regulatory state. Furthermore, the power bids are either multiplied or divided by the square root of the efficiency, η, to obtain the actual energy added or removed. $P_{pcd,t+h}$ is the forecast peak consumer power demand at time $t+h$. The maximum is taken of the battery design power and the possible power based on the available capacity. Last, $P_{bda,t+h}$ and $P_{sda,t+h}$ are the bought and sold power on the day-ahead market at time $t+h$.

5.3.2. Forecast aided bidding

Using the naive approach is a risk-averse strategy as the imbalance capacity would never be able to go outside the set limits, when not considering the over-under flow from the consumer part. This also means that the battery often has a sub-optimal use of resources, as the placed bids are often not activated. To increase the available volume, a forecast is made on the coming regulatory states. Knowing the chance that the state will be the opposite of the state we bid in allows us to reduce the volume that needs to be taken into account. For example, if we are 80% sure that the next state will be upward, only 20% of the downward volume that has been bid in is taken into account for calculating the next downward order. Such an approach often works well with longer chains as the errors would average out, however, the chain considered here is relatively short. Therefore, this approach might not be the most optimal solution. Further research could further optimize the short term bidding strategy.

Not completely discarding the placed bid reduces the risk that the prediction was wrong. Furthermore, it is not necessary to use all the forecasted states, such that it is possible to only use the forecast for the next PTU, but not for that one after if it turns out that the forecast accuracy further ahead is too poor. This freedom allows adjusting the risk levels based on the risk appetite and battery specs. For example, for a battery with a high power-energy ratio, it is riskier to increase the number of forecasts to take into account, as the number of consecutive minutes of (dis)charging available is lower.

Furthermore, predictions further away could be of less accuracy compared to those close by. To introduce the state change into the naive power calculation formula $\hat{\theta}_{\tilde{t}}$ is introduced, the probability that the state at time $\tilde{t}$ is an upward state, such that

$$\begin{aligned} P_{up,t+h} =& max((E_{I,t} - \underline{\sigma}_{I,t} - \sum_{\tilde{t}=t}^{t+h-1} \left(\mathbb{1}\{\tilde{t} < t+\nu\}\hat{\theta}_{\tilde{t}} + \mathbb{1}\{\tilde{t} \geq t+\nu\}\right) * \\ & \frac{P_{up,\tilde{t}} * \delta}{60\sqrt{\eta}}) * \frac{60}{\delta} - P_{pcd,t+h} - P_{bda,t+h} + P_{sda,t+h}, P_{max}), \end{aligned} \tag{5.5}$$

and

$$\begin{aligned} P_{down,t+h} =& max((\overline{\sigma}_{I,t} - E_{I,t} - \sum_{\tilde{t}=t}^{t+h-1} \left(\mathbb{1}\{\tilde{t} < t+\nu\}(1-\hat{\theta}_{\tilde{t}}) + \mathbb{1}\{\tilde{t} \geq t+\nu\}\right) \frac{P_{down,\tilde{t}} * \sqrt{\eta} * \delta}{60}) \\ & * \frac{60}{\delta} + P_{pcd,t+h} + P_{bda,t+h} - P_{sda,t+h}, P_{max}, P_{max}). \end{aligned} \tag{5.6}$$

Here $\mathbb{1}\{A\}$ is an indicator function such that $\mathbb{1}\{A\} = \begin{cases} 1 & \text{A is true} \\ 0 & \text{otherwise} \end{cases}$. ν is a constant indicating the number of forecasts to take into account. To even further increase the effectiveness of bidding on the right times, the algorithm is aided with a forecast based rule that withholds from bidding upward if the battery is nearly empty and the chance of the state being upward is slim, the same goes for downward. Mathematically this can be written as

$$P_{up,t+h} = \begin{cases} 0 & \hat{\theta}_{t+h} > \epsilon_{state,up} \text{ and } E_t < \epsilon_{energy,up} \\ P_{up,t+h} & \text{otherwise} \end{cases}, \tag{5.7}$$

and

$$P_{down,t+h} = \begin{cases} 0 & \hat{\theta}_{t+h} > \epsilon_{state,down} \text{ and } E_t > \epsilon_{energy,down} \\ P_{down,t+h} & \text{otherwise} \end{cases}. \tag{5.8}$$

Where ϵ are custom limits, the ϵ_{state} is a limit specifying the certainty of the state forecast and ϵ_{energy} is a limit based on the current energy level on the imbalance side of the battery. By withholding a bid more volume is available for the next PTU.

Next, the price is also adjusted based on a forecast of the imbalance price. If the certainty of an upward price above the moving median is above a set threshold, then the ask price is reduced to 0 in order to be activated for as long as possible. The same goes for the downward chance, if this is above a set threshold, the bid price is increased by multiplying the original bid price with two.

5.4. Machine learning based forecasting

Next, the previously introduced forecasts need to be created, the set-up for simulating the forecasts over a longer period of time is discussed inside this section. Model specific details, such as the specific features or model architecture are covered in their respective sections inside chapter 6.

5.4.1. Auto machine learning literature

To the best of my knowledge, no other research on electricity price forecasting has implemented means to deal with the degradation of forecasting performance over time. Therefore, the literature is reviewed in other segments. However, it must be noted that Gonçalves Jr et al. (2014) is using an electricity price dataset, but the dataset is substantially different, such that it cannot be classified directly as price forecasting. The dataset considered only labels if the price was higher/lower than the previous price (Kolter & Maloof, 2007). Gonçalves Jr et al. (2014) compares different drift detection methods and shows that ADWIN performs the best on real-world datasets and gradual drift, which makes it the perfect choice for this research. To track the distribution Kavikondala et al. (2019) tracks the incoming data distribution and if the model accuracy does not change significantly. Benedetti et al. (2016) uses a fixed thresh hold for forecasting accuracy.

The question of whether to partially or fully retrain a model is covered by Kolcun et al. (2020). The authors show that freezing a part of the neural network slightly reduced the forecasting power, but also the training time. As training time is not an issue in this research, the neural network is not extremely complicated, thus a full retrain is chosen to obtain the best forecasting results.

In Benedetti et al. (2016) the authors use a sliding and an expanding window approach to retrain the model. They call it mobile and growing, but many more names exist for the same approach. The authors show that in their case the expanding and rolling window approaches perform similarly. An overview of retraining techniques is given by Celik and Vanschoren (2021), who also state that auto machine-learning is a relatively new concept and not much covered in the literature. The authors interestingly show that completely restarting the model after drift results in the highest performance, while only retraining or warm-starting see a loss in performance after concept drift. The restart includes a rerun of the hyperparameter optimization, but not a new feature search.

5.4.2. Initial forecasting Model

The initial models are trained on the dataset from 2015-2017, which is split using a 80-20 split without shuffle. This is done in order to validate the model on the most recent data, the initial split is shown in Figure 5.6 for the day ahead forecast.

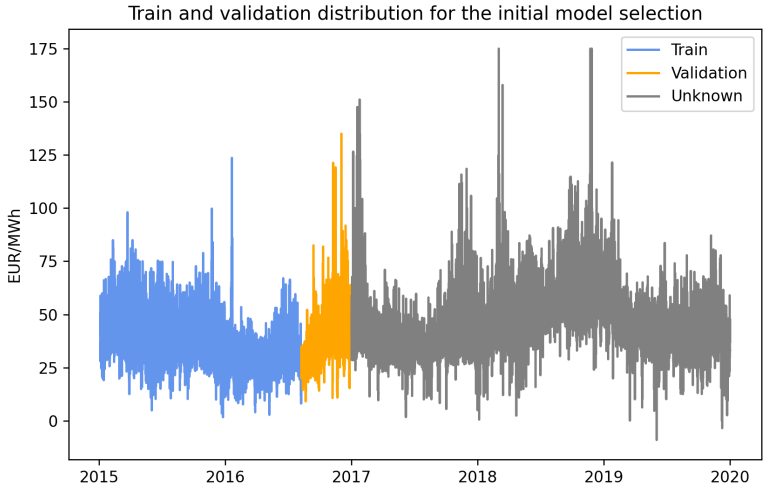
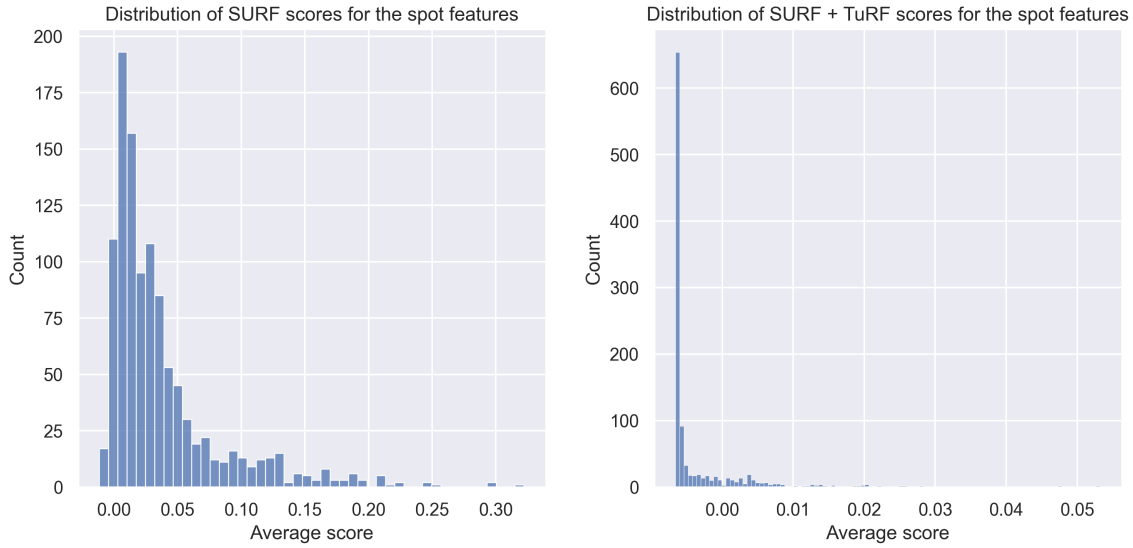


Figure 5.6: The train and validation set used to train the initial model

For the feature selection, it is decided to use an RBA, especially the SURF or MultiSURF. RBAs are feature scoring algorithms, which means that the relative importance of each feature is known, but not how many of those features are necessary for the machine learning algorithm to effectively learn. Therefore, the best cut-off weight needs to be found, meaning that all features equal to or larger than the cut-off weight are used by the machine learning algorithm. This is done by creating evenly spaced cut-off weights, starting from zero. All the features with a score smaller than zero have been eliminated, as those do not contain any predictive value. It must be noted that the feature weight distributions, for both the RBA and the RBA with the TuRF extension, show a right-skewed distribution, shown in Figure 5.7, this results that the cut-off weight bins do not contain equal amounts of features, but decrease with increasing cut-off weight.



(a) Average score distribution for the SURF algorithm

(b) Average score distribution for the SURF + TuRF algorithm

Figure 5.7: The distribution of the average feature scores as obtained by the SURF and the SURF + TuRF algorithm, where the TuRF distribution depends on the number of features returned.

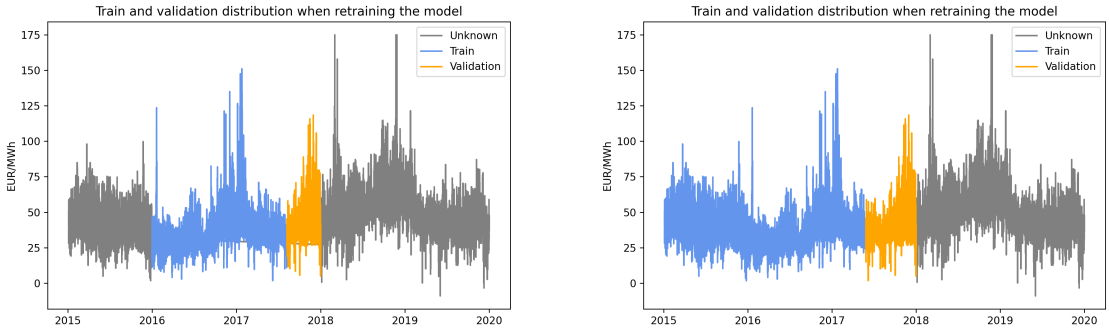
For every selection of features, based on the cut-off weight, the initial model is refitted. The cut-off weight scoring the best on the validation set (last 20% of the 2015-2017 dataset), where the definition of "best" differs per model, is used as the cut-off weight. Next, the best feature set is then used for obtaining the final model, by running a hyperparameter tuning.

5.4.3. Adaptive forecasting model

Furthermore, the model also needs to be updated promptly to deal with changing consumer behavior or markets. To detect those changes the ADWIN algorithm, explained in Section 3.3, is used. The described approach is used for the different machine learning models, wherefore every model is monitored individually and independent from the others.

Hereafter the model starts predicting in a forward-moving manner, meaning that it starts with predicting the first of January 2017. Afterward, the forecast errors are calculated and the model obtains new data from that day making a new prediction for the next day. The errors are fed into the ADWIN algorithm which tracks if a break in the data has been observed, shown in Figure 5.3 in red.

If the ADWIN model detects a change, it triggers the script to retrain the model. The newly seen data is added to the possible train set. Figure 5.8 shows the two different approaches for adding the data, Figure 5.8a shows an expanding window approach where all the past data is added to the train set, while Figure 5.8b uses a fixed window length of two years. In this study, both approaches are tested and compared. The window length selected by ADWIN is not used to train the model, as that would result in too few samples to effectively train the model without having to opt for transfer learning: an approach that could be used in further research to further decrease the errors and reduce the training time.



(a) Fixed window approach

(b) Expanding window approach

Figure 5.8: The fixed window uses a static approach of always taking two years of data, while the expanding window adds all the past data to the train-validation set. The difference between the two is shown on the day-ahead prices.

Furthermore, when a break is found, an RBA is rerun on the instances seen between two breaks. The idea is to slightly overfit the model on the most recent data, as that would be best in describing the next period. Therefore, the validation data is used at the end, without a shuffle, of the window. And the RBA is fit onto the newly seen data since the last retrain, such that if the model was last retrained in December 2018, the RBA is fit on the data from December 2018 until now. The reason is that inside that period the correlation has changed enough to break the model. The minimum window length is equal to the minimum window length of the ADWIN algorithm, which is 5 days by default. Moreover, this subset corresponds with the data captured by the ADWIN algorithm as being within the same distribution range. After refitting the RBA the best cut-off weight is obtained, as described in subsection 5.4.2 and the hyper-parameters of the machine-learning model are re-tuned. This results in a completely new model that should be able to deal with the new data again.

6

The different machine learning models

This chapter covers the different machine learning models used. The chapter starts with the model to forecast Day-ahead prices in Section 6.1, followed by the coverage of the imbalance forecasting in Section 6.2. Lastly, the models to forecast the load for both a single and multiple users are covered in Section 6.3.

6.1. Day-ahead forecasting

In this chapter the process of forecasting the Dutch day-ahead prices is covered. In this section the different features considered are covered, but also a method to obtain the features that carry the most information is discussed. Furthermore, the model architecture, the problem of concept drift on the spot market and finally the results are covered.

6.1.1. General literature

On the subject of forecasting the day-ahead prices, a lot of research has been done. However, the number of researchers focussing on the Netherlands is limited. Visser et al. (2020) compare different machine learning algorithms and shows that a Random Forest performs best. However, the results are difficult to compare as the authors use features that are not available during gate closure, meaning that those features cannot be used in an operational setting, simply because they are not known yet. The same seems to occur more often, for example, Wang et al. (2017) are also using features not yet available.

Furthermore, the result that a Random Forest performs best is surprising as the review by Lago et al. (2018) shows that a DNN is unbeaten on the Belgian spot prices. The differences in results might come due to the different periods, where Lago et al. is using 2010 to 2016, and Visser et al. is using 2019 only. Another interesting finding by Lago et al. is that more advanced algorithms, such as GRU or LSTM do not aid the forecasting above a simpler 2-layer DNN, and that machine learning algorithms outperform statistical methods such as ARIMA-based models.

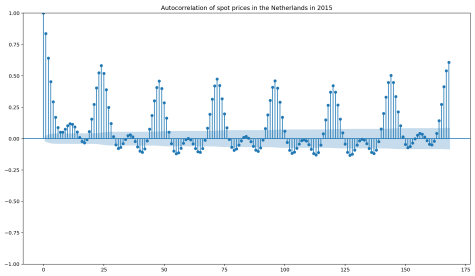
Another research on forecasting Dutch day-ahead prices is done by Van Der Heijden et al. (2021). Van Der Heijden et al. shows the importance of incorporating features of neighboring countries which can be attributed to the fact that power grids are getting more interconnected. Besides, the authors show that a linear model obtains better results compared to a non-linear model such as a DNN. However, the DNN outperforms the linear model during times of high volatility.

Besides the just named authors, many more have researched the topic of forecasting day-ahead prices, but Amjady and Keynia (2009) stands out. They use a modified version of the reliefF algorithm for feature selection, but newer versions already exist, such as covered in Section 3.4, which is used in this study.

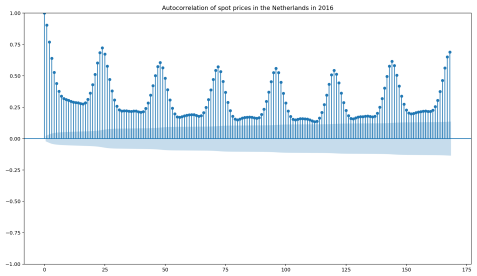
6.1.2. Features

As shown in literature, such as by Wu and Shahidehpour (2010), Jakaša et al. (2011) and others, spot prices show a strong auto-correlation, which also holds for the Netherlands as shown in Figure 6.1. The sub-figures show that the auto-correlation changes over the years. The change can likely be attributed to the expansion of renewable energy in the Dutch grid. The auto-correlation plot shows a strong correlation at the 24th lag, but also every multiple of 24th. This pattern logically follows from the cyclical nature of the day-ahead market. The correlation peaks again at the 168th lag, the price of 7 days prior. For this reason, it is chosen to include 168 lags of the spot price.

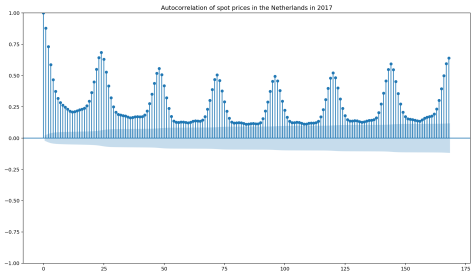
Furthermore Figure 6.1d and Figure 6.1e start to show a new pattern. The peaks at every 12th lag are likely the result of the so-called duck curve (Office of Energy Efficiency and Renewable Energy, 2017). The duck curve is the term describing the demand curve in a power system with a lot of solar PV adaption. The curve can be characterized by a peak demand early in the day and late in the afternoon. In the middle of the day, the demand decreases due to consumers generating their electricity with their solar PV.



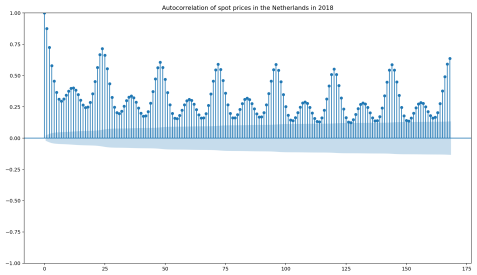
(a) Auto-correlation in 2015



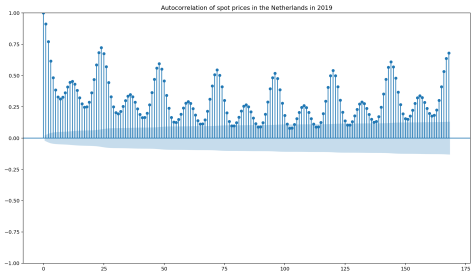
(b) Auto-correlation in 2016



(c) Auto-correlation in 2017



(d) Auto-correlation in 2018



(e) Auto-correlation in 2019

Figure 6.1: The auto-correlation of Dutch spot prices shows a reoccurring pattern every 24th lag. However, the pattern during the day changes a lot.

In all years a stronger auto-correlation can be observed at every 24th lag of the spot price. To increase the usability of this recurrence, the 7-day moving average Dutch spot price of an hour is used as a separate feature, such that:

$$MA_h = \frac{\sum_1^{i=7} \Phi_{(h,d-i)}}{7} \; \forall \; h = 1, ..., 24, \tag{6.1}$$

where Φ_h is the day-ahead price at hour *h* and *d* is the day. When plotting the generated feature against the spot price, shown in Figure 6.2, it can be seen that just using the MA-feature can already result in a decent day-ahead price forecast.

Van Der Heijden et al. (2021) show a strong correlation of the Dutch spot prices with the neighboring prices attributed to an increasing amount of interconnections between countries. Therefore, the feature set is expanded with the historical prices of grid-connected neighbors. The neighbors in question are Belgium, Denmark, Germany, France, and Norway. All of the countries, except France, share a physical interconnector with the Netherlands allowing the purchase/sale of power from each other. Lastly, France is added even though it does not have a physical interconnection with the Netherlands. France is however added on the combined results from Van Der Heijden et al. (2021) and the advice from Veenman (personal communication, 2021, March) a market analyst covering the Dutch day-ahead market.

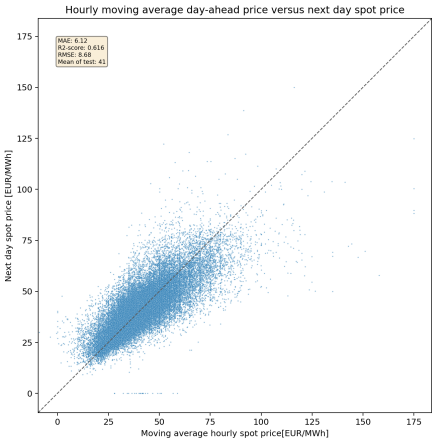


Figure 6.2: A strong linear correlation between the hourly moving average price and the price of the next day can be observed for the whole data set

However, the effects of the neighbors do have a limitation, the spill-over effects are limited by the availability of the inter-connectors. In events where an inter-connector is offline or has a reduced availability, it is unlikely that the country on the other side of the connector is as relevant as before. Therefore, the predicted available power on the inter-connectors, per country, is added as a feature. The prediction is used instead of the actual value as sudden outages are not known at the moment of bidding on the market.

Next, the marginal costs of conventional power plants are used as a feature. The marginal cost of a power plant is seen as the optimal price at which a powerplant is bid in at the spot market (I. J. Perez-Arriaga, 2016). The marginal costs consist of the price for carbon emissions and fuel, which is the minimum required price a power plant would need to cover the variable costs. For forecasting the Dutch day-ahead price only the marginal cost of natural gas and coal-fired power plants are considered as those are the main conventional power plants used in the Netherlands (ENTSO-e, 2022). The marginal costs, in EUR/MWh, are calculated as follows:

$$MC = \frac{\Phi_{fuel}}{\eta} + \Phi_{carbon} * CE, \tag{6.2}$$

where Φ_{fuel} is the price of fuel in EUR/MWh, η the efficiency of the power plant, Φ_{carbon} the price of carbon EUR/ton and *CE* the ton carbon emissions per MWh electricity produced.

The next set of features consists of residual load forecast (RLF) based features of the different countries, suggested by Veenman (personal communication, 2021). The residual load is the amount of power that needs to be satisfied with conventional power plants. According to market theory, power-plant operators will bid their assets against the marginal cost of the power plant. The marginal cost of renewable assets (not considering bio-based plants) is 0 EUR/MWh, so if you subtract those 'free' sources you are left with the power that needs to be generated with paid sources, from which the highest is the price setter. The formula of the residual load forecast P_{RLF}, which is not directly used as feature, is the load forecast P_{LF} minus the sum of the wind-power forecast P_{WF} and solar-power forecast P_{SF} at time *h*, such that:

$$P_{RLF,h} = P_{LF,h} - P_{WF,h} - P_{SF,h} \; \forall \; h = 1, ..., 24. \tag{6.3}$$

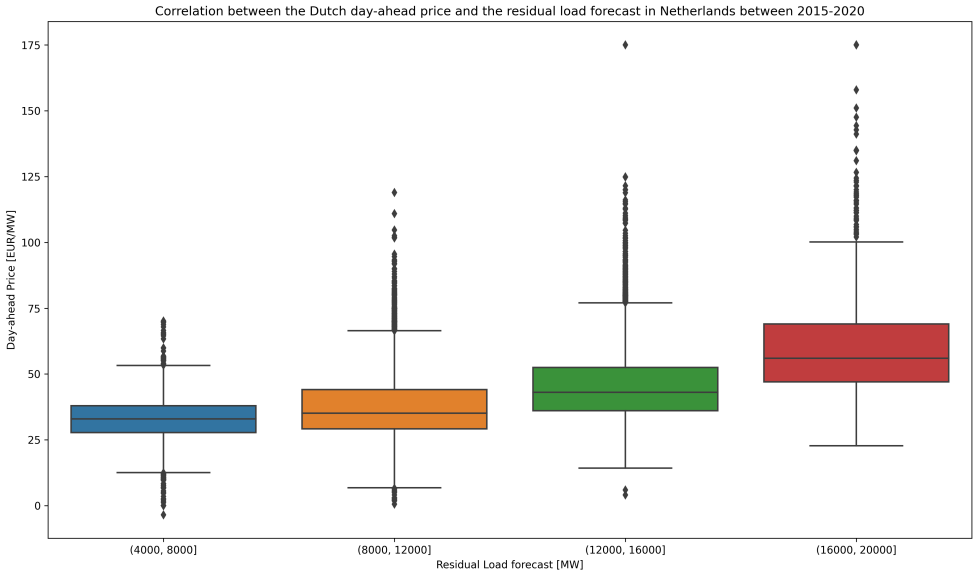
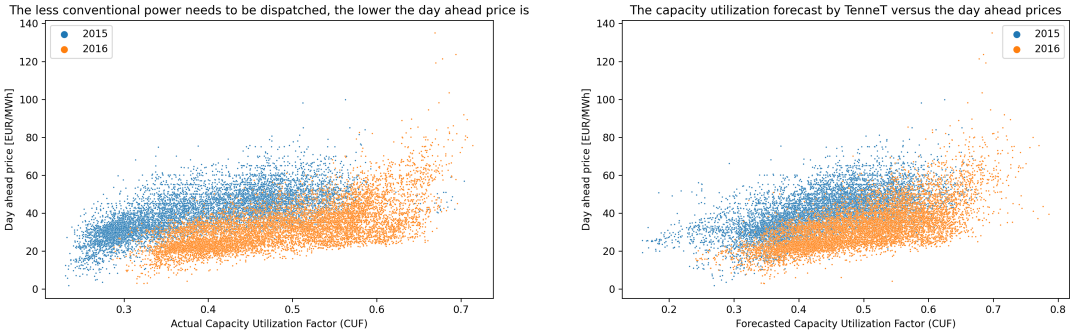


Figure 6.3: The correlation between the residual load forecast for the Netherlands and the spot price shows

The residual load forecast is then used to obtain the Capacity Utilization Factor (CUF) and a Relative Load Indicator (RLI), as proposed by Keles et al. (2016). The CUF is the percentage of conventional sources occupied:

$$CUF_h = \frac{P_{RLF,h}}{AC_h} \; \forall \; h = 1, ..., 24, \tag{6.4}$$

where AC_h is the available conventional capacity at hour *h*. The available capacity does not account for scheduled unavailability of power plants as the data source does not clearly define when the unavailability of the power plant is announced. This is problematic as the full available capacity is thus not captured meaning that there is an over-estimation of the open capacity. Another issue with the CUF is that the country-wide capacity is only updated on a yearly basis, meaning that newly added power plants are not accounted for until the year ends. Nonetheless, the CUF is still a useful factor which deals with the non-stationarity of the data and scales the variable.



(a) Utilizing the actual realized renewable production and load(b) The by TenneT forecasted renewable production and load

Figure 6.4: The Actual CUF versus the Day ahead price compared with the forecasted CUF shows a linear pattern, but the forecasted CUF has a less sharp drop in price below 30%.

Comparing Figure 6.4a it can be observed that the dispatch of little conventional assets, due to a large amount of renewable energy (or little load), reduces the price. While the dispatch of nearly all conventional power plants greatly increases the price. When Figure 6.4a is compared with Figure 6.4b, it can be noticed that the forecast made by TenneT does not fully capture the same pattern. This is because TenneT provides ENTSO-e with poor forecasts as shown by Dankers (2021). Increasing the accuracy of the forecast of renewable energy could potentially increase the accuracy of the forecasting model.

Last, the RLI is an indicator of the change in the residual load forecast compared to a previous timestamp. Just like Keles et al. (2016) the RLI_{24}, RLI_{48}, RLI_{168} have been considered. The RLI is calculated by dividing the current load forecast by the load forecast of *n* hour ago, such that:

$$RLI_n = \frac{P_{RLF}}{P_{RLF,t-n}}. \tag{6.5}$$

6.1.3. Cyclical or one-hot encoded time dependent categorical features

Cyclical encoding of time-dependent features, such as an hour of the day and month of the year, is a common practice in machine-learning (Mahajan et al., 2021). The purpose is to allow the machine learning algorithm to learn that a feature occurs in a cycle (Wyk, 2018), which is lost when using one-hot encoding. The cyclical encoding is obtained by using sine and cosine to alter the feature, such that:

$$x_{cos} = \frac{2 * \pi * x}{x_{max}} \tag{6.6}$$

and

$$x_{sin} = \frac{2 * \pi * x}{x_{max}}. \tag{6.7}$$

Graphically the result of using the sine/cosine representation of a cyclical feature is shown in Figure 6.5.

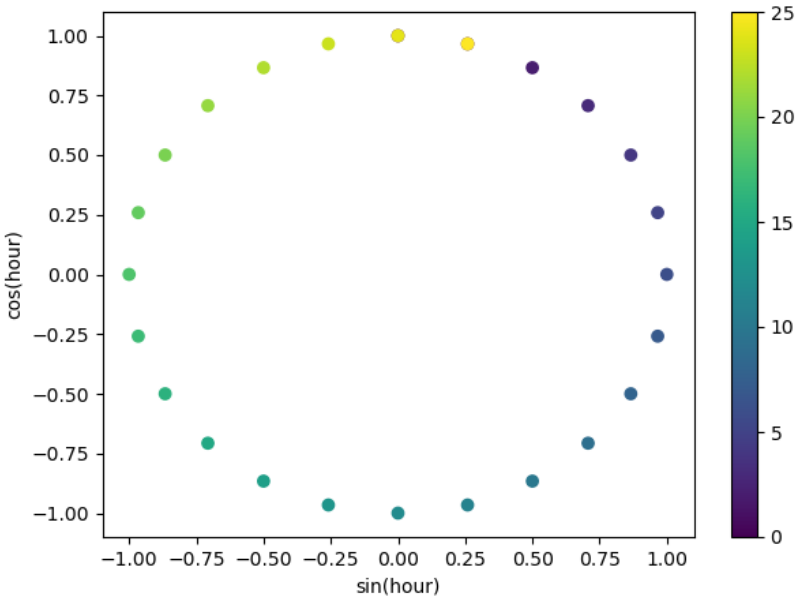


Figure 6.5: Plotting the cyclical representation of the hours of the day (scikit-learn, 2022).

The added benefit of the cyclical representation over the one-hot encoded is the reduction of feature variables. The argument in favor of using one-hot encoding instead of the sine-cosine presentation is when events occur at always the exact time, for example, the Dutch air-raid system, which is tested every first of the month at 1200. In the case of electricity prices, the truth is somewhere in the middle. Peaks happen roughly at the same time, but shift between hours, such as seen in Figure 6.6 where the median of the price-peaks is roughly the same such as for the 9th and 10th hour.

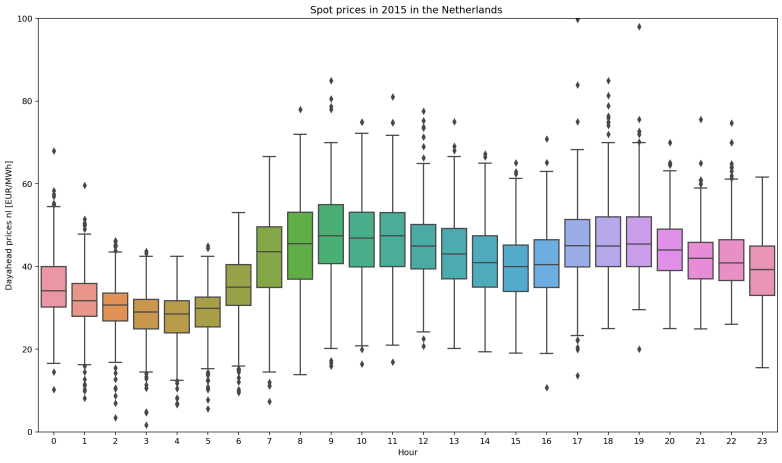


Figure 6.6: Hourly spot prices in the Netherlands in 2015.

Furthermore, it could even be argued that the hour of the day is not needed at all, since the time information is indirectly contained in the historical prices. To find the best solution for this data set, both are tested. The 2015-2017 data set is split, without shuffle, into a train a and validation set (80-20). At first, no feature selection is applied to the feature set, thus the only difference is that in one set the hours, months, and days

of the week are one-hot encoded and in the other, those are implemented as cyclical features. The results, as summarized in Table 6.1, show that the Algorithm benefits from using the one-hot encoded features.

	MAE	RMSE
One-hot Encoded	7.2	11.3
Cyclical	7.5	12.1

Table 6.1: Applying cyclical feature engineering instead of one-hot encoded aids the algorithm when using all the features. The results are obtained by fitting the model using 4 different seed values to account for the stochastic nature of a DNN.

However, the relief algorithm tends to underestimate continuous variables when combined with binary variables (Kononenko & Sĭkonja, 2007), which is the case with one-hot encoded features. Moreover, using the cyclical representation reduces the number of features. Therefore, it is chosen to use the cyclical instead of one-hot encoded features.

6.1.4. Data cleaning

The main source of data used in the models is data published by the different TSOs. Every TSO has its methodology for cleaning and checking the data provided. Most of the TSOs seem to do an accurate job in providing data, however, some data might slip the checks. An obvious example is the load forecast for the UK, shown in Figure 6.7. The load forecast suddenly drops to less than 25% compared with the previous day. To deal with such an event, the actual and forecasted load are compared to the previous day, if the deviation is more than 75%, it is considered an outlier. This may seem like a lot, but an extremely sunny day after a cloudy one, could reduce the load by quite a lot and those days should not be dropped. The outliers are replaced with the data from the previous day.

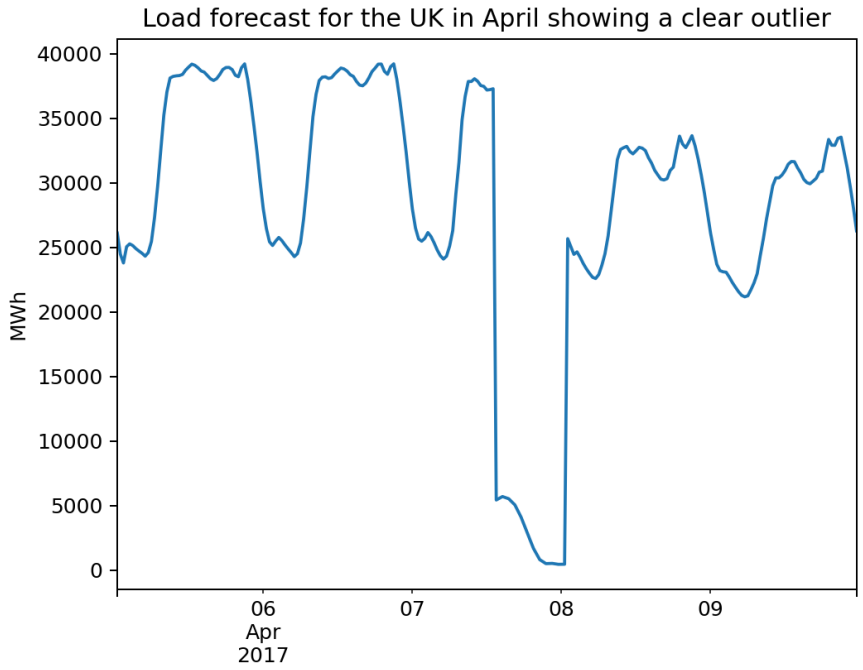


Figure 6.7: The load forecast provided by the TSO of the UK for march, showing a clear outlier

6.1.5. Feature selection

To find the best feature set, the best feature selection method needs to be chosen. The data of 2015-2017 is used for this purpose, where the last 20% of the data set serves as the test set. The dataset contains roughly 1500 features, whereby time lags of a feature are considered as a seperate features. Table 6.2 shows the top 10 highest ranked features by the different feature selection algorithms. The hourly moving average (*Hourly MA*), is a recurring feature in the different feature sets. When observing the results of the SURF and MultiSURF algorithm, it can be seen that the most occurring features are historical price features (>70% of the top 10 features from the SURF and MultiSURF), denoted by *Spot ... t-x*. The TuRF addition reduces the price features to 50%, showing that the TuRF addition detects subtle effects that can change the outcome of the price for the next day. For example, the price increase of gas would increase the peak prices where gas plants are price setters, which is captured in the marginal costs of a gas plant (*MC of gas*).

	SURF	MultiSURF	MultiSURF + TuRF	SURF + TuRF
#1 Feature	Hourly MA-7d NL	Hourly MA-7d NL	CUF DE	Spot NL t-168
#2 Feature	Spot NL t-168	Spot NL t-168	CUF BE	CUF DE
#3 Feature	Spot NL t-24	Spot NL t-24	Spot NL t-24	Hourly MA-7d NL
#4 Feature	Spot NL t-167	Spot NL t-167	Hourly MA-7d NL	CUF BE
#5 Feature	Hourly MA-7d BE	Spot NL t-25	Spot NL t-168	Spot NL t-167
#6 Feature	Spot NL t-25	Hourly MA-7d BE	CUF GB	Spot NL t-24
#7 Feature	Spot NL t-144	CUF DE	Spot NL t-25	CUF GB
#8 Feature	Spot BE t-168	Spot NL t-144	Spot NL t-167	MC of Gas
#9 Feature	Spot BE t-167	Hourly MA-7d DE	CUF NL	Spot NL t-25
#10 Feature	Spot FR t-168	Spot FR t-168	Spot NL t-144	Spot NL t-144

Table 6.2: The top 10 highest ranked features obtain by the different feature selection algorithms. The most important features share a lot of similarities between them.

Another interesting observation that can be made from Table 6.2 is that the forecast of the Capacity Utilization Factor (CUF) is ranked as an important feature. But surprisingly, the CUF forecast for Germany and Great Britain obtains a higher score than the Dutch forecast. Germany is one of the larger European producers of solar and wind energy and can impact the prices of neighboring countries if supply exceeds demand. Further research could extract solar and wind energy as separate features to investigate the effect on the forecasts.

After establishing that the TuRF addition results in a largely different feature set, the performance of the different feature sets are tested. To test which feature selection method results in the lowest errors, the different feature sets are used to fit a model. For this a simple 2-layer dense neural network is used, where the hyper-parameters are tuned for each feature-set. The Hyperband optimization search is set to only run a maximum of 100 trials to keep the run time reasonable. The best cut-off weight, which is selected based on the lowest RMSE, is then used to obtain the best-performing feature selection method. The neural network is initialized with different seeds, to account for the stochastic nature of neural networks. The average values of those four trials are displayed in Table 6.3.

	MAE	RMSE	r^2	Time for a single fit [min]
SURF	4.7	7.0	0.67	2
MultiSURF	4.7	7.0	0.67	1
SURF + TuRF	4.3	6.5	0.72	5
MultiSURF + TuRF	4.3	6.5	0.72	10

Table 6.3: Using the SURF + TuRF feature selection method yields the same errors as the MultiSURF method, but takes half the time to fit.

Table 6.3 shows that the SURF and MultiSURF score similar. However, when implementing the TuRF reduction algorithm on top of the feature selection method the error improves drastically. The drawback is the increased time for a single fit of a thousand instances. For the model implementation, the SURF + TuRF

feature selection algorithm is chosen due to the reduced time it costs to fit a thousand instances, while the errors are the same.

6.1.6. Neural network architecture

To find the best fitting model architecture, different architectures have been tested. All tests consisted of hyperparameter tuning on the same data set. The tuned models are then tested using four different seeds to prevent a model from randomly scoring well instead of having an architectural advantage over the other models.

The main model is an MLP consisting of two layers and a dropout layer after each. The second model is extended by adding a batch normalization (BN) layer in between the dense and the dropout layer. Those two models are the models discussed by Van Der Heijden et al. (2021). In their paper they use the addition of a batch-normalization layer as a tunable feature, meaning that their optimization algorithm decided to add the layer or not, while here both are tested individually to observe the differences between the architectures.

The test results, as summarized in Table 6.4, show that the additional batch normalization reduces the errors. However, the drawback of adding a batch normalization layer is a higher dependency on the seed, as can be seen by the increase in the standard deviation of the errors.

Inspired by He et al. (2016) and their ResNet architecture, which introduced skipping certain layers to increase the performance on image recognition tasks, I try a similar architecture to increase performance on this regression task. The hypothesis behind a skip-layer is that information that does not need to be transformed, can flow freely around those layers.

The architecture proposed in this study is a batch normalization combined with a skip such as shown in Figure 6.8. The image shows how data flows through the BN-layer, followed by a nonlinear activation and a dropout layer, and is then concatenated with the data simply activated by a linear activation function. The result of skipping the nonlinear activation function is that information can also be passed through linearly. Hypothetically this would aid features that have a linear relation with the dependent variable. This could potentially aid the observation found by Van Der Heijden et al. (2021) that a linear model outperforms a non-linear neural network in moments of low volatility. With this architecture, the model should have the best of both worlds. Furthermore, fitting a non-linear model on a linear relation might result in an overfit as the non-linear function tends to look for a more complicated relationship.

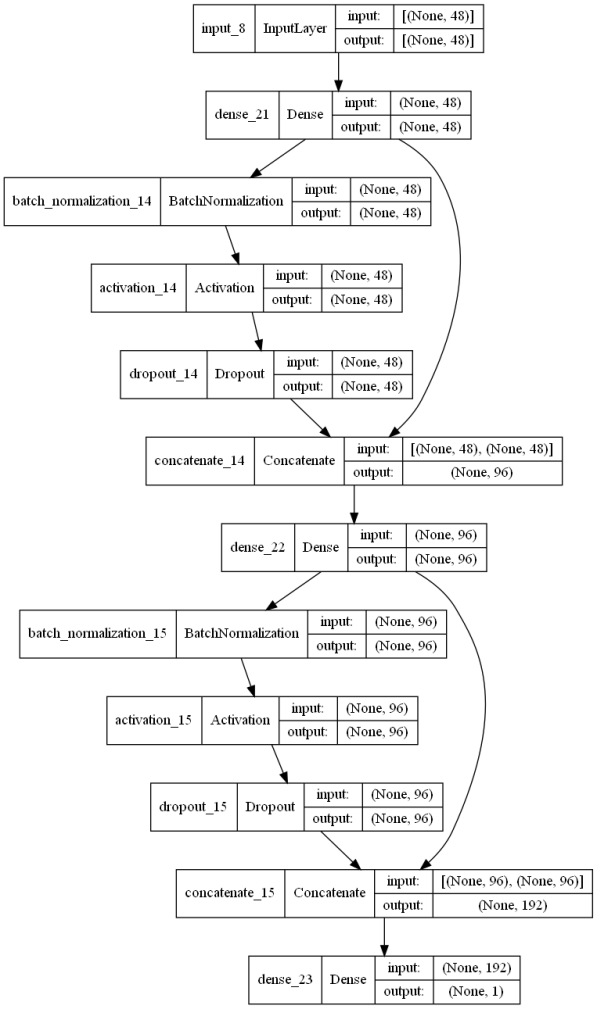


Figure 6.8: Proposed model architecture using a batch normalization and non-linear activation function, but also the ability to pass data forward in a linear fashion.

Every model architecture is initialized by 4 different seeds to account for the stochastic nature of neural networks. Both the errors and the standard deviation of the errors for the different seeds are shown in Table 6.4. The results show that the addition of the BN-layer helps by reducing the error, but, based on the increased standard deviation of the errors, increases the seed dependency. Adding the BN-skip reduced the error even further, while also reducing the dependency on a random seed. It must be noted that this does not prove the hypothesis of the working of the BN-skip layer, it just shows that the architecture is beneficial for this type of problem.

Lastly, a third layer is added consisting of the same architecture as the prior layers. The effect of a third layer on the error is small, as shown in Table 6.4. The extra trainable parameters increase with the introduction of a third layer, therefore the simpler model, of only two layers, has the preference.

	MAE		RMSE		R2	
	Average	Std	Average	Std	Average	Std
2 layers	4.47	0.03	6.6	0.013	0.71	1E-3
2 layers BN	4.28	0.05	6.51	0.05	0.72	4E-3
2 layers BN-Skip	4.09	0.034	6.21	0.018	0.74	1.7E-3
3 layers BN-skip	4.08	0.024	6.19	0.022	0.75	1.9E-3

Table 6.4: Adding Batch Normalization to the model architecture helps in reducing the error, but increases the dependency on the seed. The addition of a skip-layer mitigates this seed dependency. Lastly, introducing the third layer does not reduce the error any further

6.1.7. Walk forward based error monitoring and retraining

After finishing all the individual pieces, the model pipeline is set up as described in chapter 5. All models show a break, determined by the ADWIN algorithm, at roughly the same time, as seen in Table 6.5. Generally, all models show a break roughly twice a year, which seems closely related to the change of the seasons. This could either mean that there is a seasonality at play that the model is not able to capture or that one of the two is easier to forecast. The latter would result in different error distributions for the summer and winter period, and thus in a trigger by the ADWIN algorithm. To distinguish the exact reason, more research is required.

	DNN Fixed	DNN Expanding	XGB Fixed	XGB Expanding
Break 1	2017 - day 128	2017 - day 128	2017 - day 128	2017 - day 128
Break 2	2017 - day 320	2017 - day 320	2017 - day 320	2017 - day 320
Break 3	2018 - day 211	2018 - day 211	2018 - day 211	2018 - day 211
Break 4	2018 - day 275	2018 - day 307	2018 - day 275	2019 - day 6
Break 5	2019 - day 6	2019 - 70	2019 - 134	-
Break 6	2019 - day 198	2021 - 75	2020 - 153	-
Break 7	2020 - day 281	2021 - 299	2021 - 43	-
Break 8	2021 - day 107	-	2021 - 107	-
Break 9	2021 - day 299	-	2021 - 203	-
Break 10	-	-	2021 - 267	-

Table 6.5: Different models show a break a different times, but a general pattern can be observed. Such as that the model roughly breaks twice a year, which could be attributed to the seasonal change of summer to winter and vice versa.

To visualize the concept drift occurring in the data set, the default model, trained on the 2015-2016 data set, is used to predict the whole test set without retraining, this model will be called the fixed model. The actual values are then divided by predictions of the model. A good model has estimations close to one, with the less variance the better. When the factor is plotted over time, shown in Figure 6.9, it shows how the model performance develops. From the figure, it can be seen that the model starts to under-predict starting roughly half of 2018. This loss of performance implies that a form of error monitoring in combination with intervention is necessary to control the model performance.

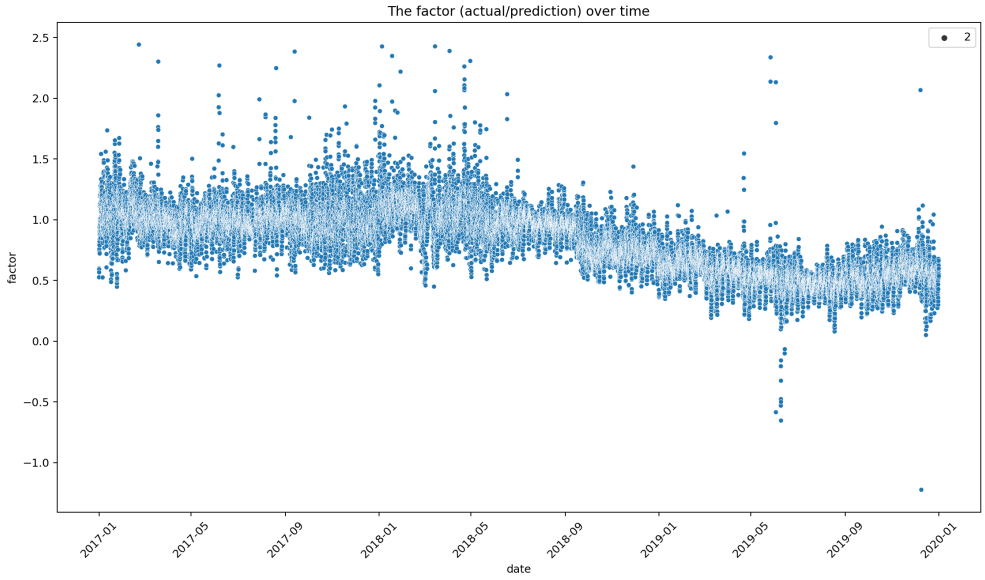


Figure 6.9: Not retraining the model results in an under-fit if used over time, rendering the existing model useless. This shows the need for an adaptive model, which timely deals with concept drift.

The model is applied in a real world set-up which means that it is not possible to correct the predictions after the fact. Such an correction would require knowledge of the actual value. Therefore, an adaptive correction factor is proposed, such that:

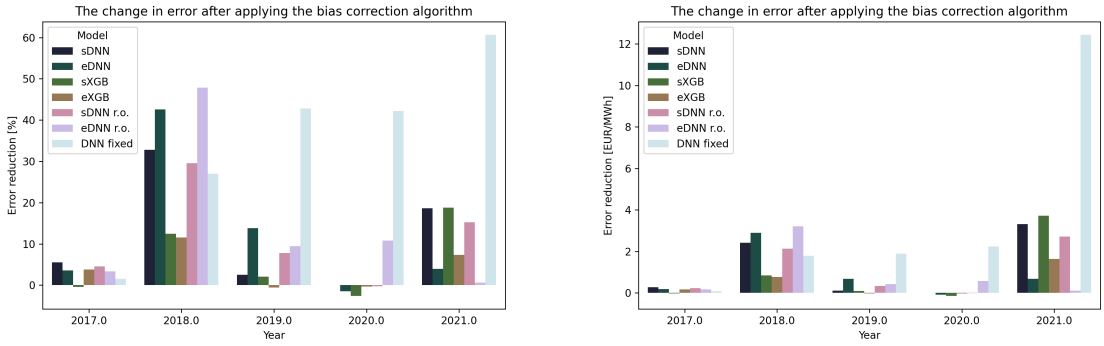
$$\hat{y}_{d,t,updated} = \hat{y}_{d,t} * \alpha_{d-1}, \tag{6.8}$$

where α is a correction factor updated daily, defined as:

$$\alpha_d = \begin{cases} 1 & d \leq 10 \\ \frac{1}{d*t} \sum_{d'=0}^{d} \sum_{t'=0}^{t} \frac{y_{d',t'}}{\hat{y}_{d',t'}} & d > 10. \end{cases} \tag{6.9}$$

Here $\hat{y}_{d,t}$ is the prediction for day *d* on the hour of the day *t*. *d* is the cumulative number of days seen from the moment of training and resets when a model is retrained. The threshold of having at least 10 days worth of prediction samples is to prevent the correction algorithm from learning the bias based on too little data. However, too many observations work counter-productive as well. The information of newly added values is being diluted as the window size grows, therefore, a cap is applied to the coefficient window for 365 days. The 365 day cap is arbitrarily chosen and could be tuned for a better performance.

The prior proposed bias-correction algorithm is likely to aid the algorithm based on the observations from Figure 6.9. The figure shows that the y-predictions only require a multiplication to reduce the bias, as the variance remains quite small and would thus result in more accurate predictions. Figure 6.11 confirms this hypothesis. This figure shows two plots, with the actual values on the x-axes and the predicted values on the y-axes. In the left plot the predictions are not corrected, while the plot on the right shows the corrected predictions. The left plot shows that the uncorrected observations lie to the right of the identity line and are somewhat turned away from the identity line, implying that the uncorrected predictions are too large in most cases and that this difference increases for larger day ahead prices. Therefore, a correction factor can shift and angle the observations towards the identity line, as shown in the right plot of Figure 6.11.

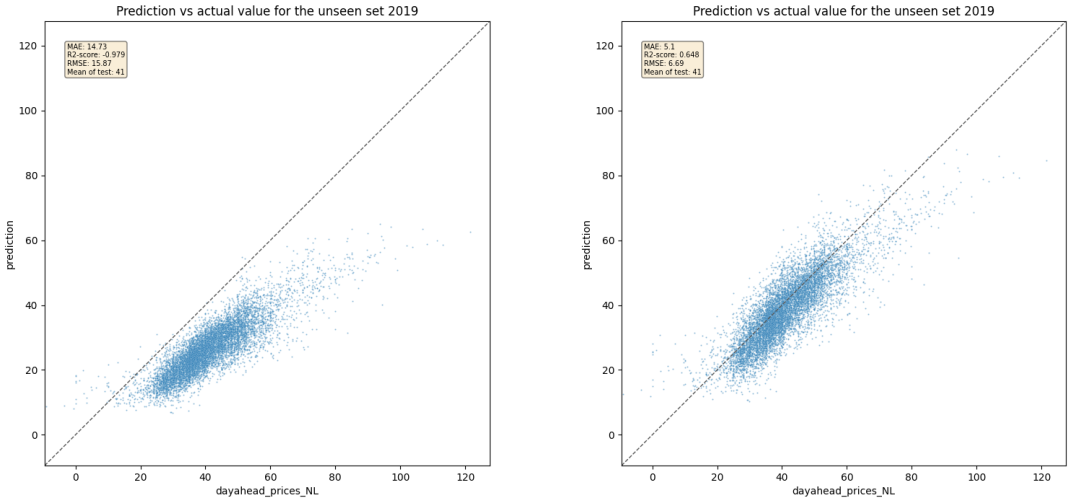


(a) The error reduction as a percentage

(b) The absolute error reduction

Figure 6.10: The correction algorithm was able to reduce the MAE for the different models up to almost 40%. The neural networks showed more bias than the XGBoost. The fixed DNN benefited most from the bias correction, due to having a relatively large bias.

The correction algorithm is also applied in addition to the different machine learning models to help the algorithm. The benefit of this algorithm becomes clear from the comparison of Figure 6.11A, the predictions without correction multiplier, and Figure 6.11B, the predictions with the correction multiplier. Overall the correction algorithm was able to reduce the MAE and RMSE, as shown in Figure 6.11. It is worth noting that the correction algorithm is counter-effective for 2020, where it increases the errors for the refitted models shown in Figure 6.10A. Furthermore, Figure 6.10 shows that the fixed model aids most from the correction, which makes sense, as that model never gets retrained during the process and has thus no way to deal with the data changes. This becomes clear from comparing the absolute error reduction in Figure 6.10B, where the fixed model also has the largest absolute error decrease. However, it does not mean that this model is also the best scoring model, but only which model has the largest bias that can be corrected by the bias correction algorithm.



(a) Plain forecast results

(b) With correction algorithm

Figure 6.11: Comparison of the predictions using an expanding window forecast for 2019 with and without correction factor

Lastly, an XGBoost-model is tested as well. Lago et al. (2018) and Van Der Heijden et al. (2021) shown that an XGBoost algorithm generally performs poorer in forecasting the day ahead prices compared to an

MLP, with the performance especially lacking in the peak hours. However, Van Der Heijden et al. (2021) argued that the XGBoost performed better during off-peak hours. The goal of the spot-forecasting is to find the best trading opportunities, which often is a combination of buying an off-peak hour and selling a peak hour. Therefore, both algorithms are tested to find the best model for this application.

6.1.8. Hyperparameter tuning

The loss function for both the XGB and the DNN is set to the Mean Squared Error. The XGB algorithm is tuned by using the Hyperopt package, for a maximum of 150 evaluation runs. The search space is selected based on a Kaggle guide, (Banerjee, 2020), for XGBoost Hyperparmeter tuning and summarized in Table 6.6. Some of the values resulting in the best validation predictions are at the boundary of the available space, which might suggest that the optimal value is either outside, or very close to the boundary.

	Options	**Best value**
Max Depth	[3, 5, 7, 8, 9, 12]	12
Eta	[0.01, 0.015, 0.025, 0.05, 0.1, 0.3]	0.3
Gamma	[0.05, 0.1, 0.3, 0.5, 0.7, 0.9, 1]	0.1
Reg alpha	[0, 0.1, 0.5, 1]	1
Reg lambda	[0.01, 0.1, 1]	0.1
Colsample by Tree	[0.6, 0.7, 0.8, 0.9, 1]	0.6
Subsample	[0.6, 0.7, 0.8, 0.9, 1]	0.9
Min child weight	[1, 3, 5, 7]	3
Rate drop	[0.1, 0.25, 0.5]	0.5
Skip drop	[0.25, 0.5, 0.75]	0.5
Booster	-	Dart

Table 6.6: The available search space and the best obtained values for the initial XGB model

The DNN is tuned by using the Keras Hyperband tuning algorithm with an early stop after 10 stale iterations of the validation loss. The algorithm is run for 20 iterations and a factor of 20, which is a reduction factor for the number of models and number of epochs for each bracket. The max epochs is set to 150, thus the Hyperband is run for only 420 total epochs per iteration. A deeper search could likely improve the model performance, but that is out of scope for this research. The search space and the best obtained values are displayed in Table 6.7.

	Search Space			**Best value**
	Min/option	Max	Step	
Activation	[Elu, ReLu]			Elu
Units Layer 1	8	160	8	96
kernel regularizer L2	0.0001	0.075	-	0.07
Dropout	0	0.5	-	0.42
Units Layer 2	8	160	8	88
kernel regularizer L2	0.0001	0.075	-	0.07
Learning rate	-	-	-	0.0001

Table 6.7: The search space used for the initial neural network

6.1.9. Model comparison

Finally, the different forecast are used in the optimization algorithm, covered in Section 2.3, using a 12MW, 7.5MWh battery, as specified in Table 4.2.2. The trading results, expressed in Table 6.8, show that the neural networks roughly perform the same, while the XGBoost algorithm performs worst. An other point worth noting is the extremely good performance of the fixed DNN, which is the basic DNN that is only fit onto the initial dataset and aided by the bias correction algorithm.

	Perfect	**DNN**		**XGBoost**		**Only retrain DNN**		**Fixed DNN**
	foresight	Sliding	Expanding	Sliding	Expanding	Sliding	Expanding	
2017	54	35.1	34.9	34.4	33.4	35.6	35.6	36.8
2018	67	47.3	48.2	36.7	41.5	46.7	48.2	47.3
2019	46	35.7	36.0	33.5	32.8	36.7	36.2	35.6
2020	69	44.2	46.7	41.5	37.7	45.7	41.1	47.0
2021	169	124.8	123.7	100.4	99.8	124.7	125.2	124.0
Total	405	287	290	246	245	289	286	291

Table 6.8: Trading results (in thousands of euros) show that the neural networks yield significantly higher returns than the XGBoost algorithms.

When looking into the performance for the different models, as shown in Table 6.9, we can observe a couple of things. First of all, the XGBoost often has the lowest or one of the lowest MAE and RMSE. However, trading-wise, the model always under-performs the DNN. This indicates that the DNN is better at forecasting the general daily shape of the neural network, as for trading it is most important to know the local minima and local maxima. Second, 2018 has been a more difficult year to predict. Table 6.10 shows the number of features necessary to obtain the best forecast, which shows that the model needed six times more variables to obtain the best results. 2018 is the hottest Dutch summer to date (Compendium voor de Leefomgeving, 2020) and in the winter of 2017-2018 the European Gas storage reached its lowest level seen in the past decade (AGSI, 2022). Both might be indicative of extraordinary price forming and throwing the models off, but more research is needed before decisive conclusions can be drawn on this topic.

		DNN		**XGB**		**Only retrain DNN**		**Fixed DNN**
		Sliding	Expanding	Sliding	Expanding	Sliding	Expanding	
2017	MAE	5.1	5.1	4.6	4.7	4.7	5.1	4.8
	RMSE	7.6	7.6	6.9	7.0	7.1	7.7	7.3
	R2	0.65	0.64	0.71	0.7	0.7	0.64	0.67
2018	MAE	7.4	6.8	7.0	6.7	6.8	6.7	6.7
	RMSE	10.5	9.7	10.2	9.8	10.1	9.6	9.3
	R2	0.52	0.60	0.55	0.59	0.56	0.60	0.62
2019	MAE	4.2	5.0	4.8	4.3	4.3	4.6	4.4
	RMSE	5.6	6.5	6.5	5.8	5.7	6.1	5.9
	R2	0.75	0.67	0.67	0.73	0.75	0.71	0.73
2020	MAE	5.3	5.9	6.7	5.7	5.6	5.3	5.3
	RMSE	8.0	8.6	9.0	8.3	8.1	7.9	7.8
	R2	0.73	0.69	0.66	0.71	0.72	0.73	0.74
2021	MAE	17.8	17.3	21.1	22.4	19.9	17.4	20.5
	RMSE	29.0	29.8	33.25	37.5	34.8	29.9	32.8
	R2	0.85	0.84	0.8	0.75	0.78	0.84	0.81

Table 6.9: Error results for the different models shows that different approach score better for different years/situations.

Another observation that can be seen from Table 6.9 is that no model always outperforms the others. There is not a single model that has better errors for all years, therefore the Diebold-Mariano (DM) test is applied to judge which model performs best. The DM-test is a test to determine if a forecast is significantly more accurate than another forecast. The DM-test assumes stationarity, therefore the data is used until 2021. To test this assumption, the Augmented Dickey-Fuller (ADF) test is used to determine if the dataset from 2017-2021 is stationary. The ADF test has the null hypothesis that the time series can be represented by a unit root, thus that the time series is non-stationary (Prabhakaran, 2019). The ADF test yields a p-value of 0.00 on the dataset and thus rejects the null hypothesis. Therefore the DM-test can be used on the dataset to determine the most significant forecast.

The results of the DM-test are shown in Figure 6.12, which shows that the expanding DNN retrain only (eDNN r.o.) is significantly more accurate than most other models. However, the results are inconclusive for

the sliding DNN (sDNN). This shows that the whole process of researching for the most important features and re-tuning of the hyper-parameters does not aid the model, but even decreases the forecasting power. This could indicate the feature importance does not change a lot over time.

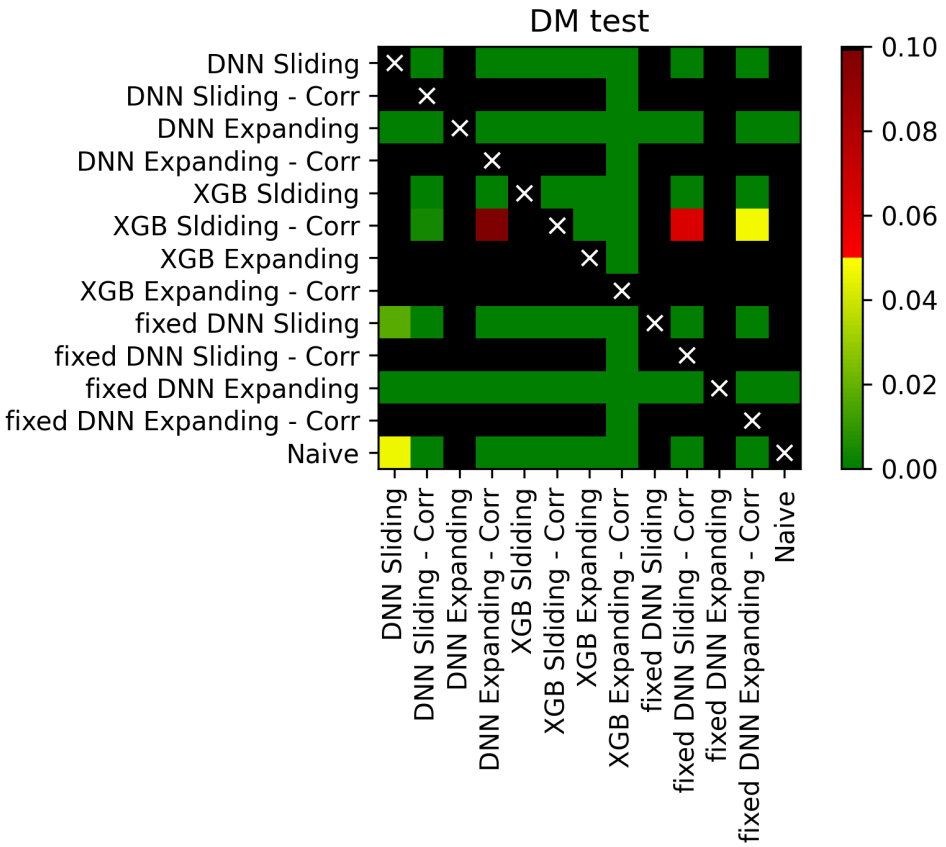


Figure 6.12: The DM-test shows that the model that is only retrained performs overall the best.

Another conclusion that can be drawn based on the DM-test is that the proposed correction algorithm aids the model significantly. The models aided with the correction algorithm, denoted by 'Corr' in the image, are always significantly more accurate than the default model without correction.

Lastly, the different features top ten ranked features are shown in Table 6.10. The total features used are based on the eDNN, which shows that the required features has increased over time. Where 3-11 features were necessary to fully capture the data between 2015-2018, the model required more as of 2018. Still, it is interesting to note that relatively few features are used, especially when compared to the total dataset that contained over 1500 features. This indicates that less than 10% of those features are enough to fully capture the day-ahead market price.

	Initial set	**May - 2017**	**Nov - 2017**	**Jul 2018**	**Nov - 2018**	**Mar - 2019**
#1	$Spot_{168}$ NL	CUF NL	CUF BE	CUF NL	CUF BE	CUF NL
#2	CUF DE	CUF BE	CUF DE	CUF BE	CUF NL	CUF BE
#3	MA_{168} NL	RLI_{24} NL	CUF NL	MA_{168} NL	CUF GB	CUF GB
#4	CUF BE	CUF DE	CUF GB	$Spot_{24}$ NL	$Spot_{24}$ NL	MA_{168} NL
#5	$Spot_{167}$ NL	CUF GB	CUF FR	CUF DE	RLI_{24} NL	RLI_{24} NL
#6	$Spot_{24}$ NL	CUF NO	RLI_{24} NL	RLI_{24} NL	MA_{168} NL	CUF DK
#7	CUF GB	CUF FR	$Spot_{168}$ NL	$Spot_{24}$ BE	$Spot_{168}$ FR	Cosine Hour
#8	MC Gas	MA_{168} GB	CUF NO	$Spot_{168}$ NL	RLI_{168} NL	$Spot_{24}$ NL
#9	$Spot_{25}$ NL	RLI_{168} NL	MA_{168} NL	CUF GB	$Spot_{168}$ NL	CUF FR
#10	$Spot_{144}$ NL	$Spot_{24}$ GB	$Spot_{167}$ NL	RLI_{168} NL	CUF FR	RLI_{48} NL
Total features	11	3	9	58	30	69

Table 6.10: The 10 highest ranked features after a break occurred in the expanding window model

6.2. Imbalance forecasting

In this section the process of forecasting parts of the imbalance markets are covered. For the imbalance market different forecasts are made, a price forecast, a state forecast and an above/under median price forecasts. The methods and results from those forecasts are covered in this chapter.

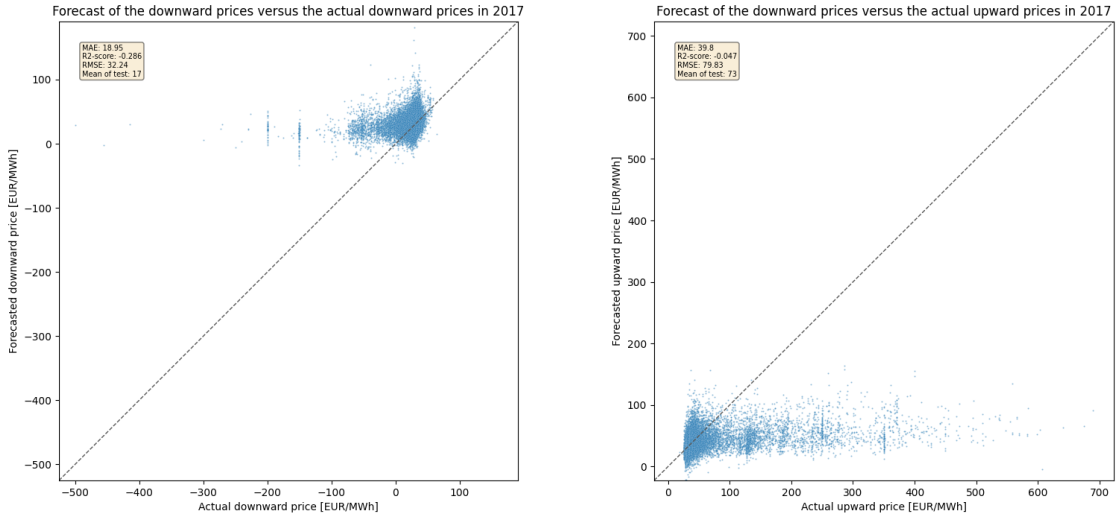
6.2.1. General literature on imbalance forecasting

On the subject of imbalance forecasting, Klæboe et al. (2015) shows that there is not a time-series (ARIMA, ARM, EXO, and ARX) based method that offers any predictive advantage over a naive model for forecasting imbalance prices. Similar results are found by Narajewski and Ziel (2020) for the hourly intraday traded products of the German market. However, the authors show that Lasso is significantly more accurate than the naive model for the Quarterly products, which are more similar to our balancing market. For Belgium, Dumas et al. (2019) shows that up to an hour in advance an MLP outperforms statistical methods in forecasting the imbalance.

Other research on the subject of intraday prices are often focused on the tradeable products, such as works by Monteiro et al. (2016), Ziel (2017), Goodarzi et al. (2019), and Uniejewski et al. (2019). Those products do have characteristics comparable with the Dutch quarterly imbalance price, but the underlying fundamentals are different. As those products are tradable on a quasi-free market the prices are bound to be influenced by market sentiment and traders' opinions. However, an interesting finding by Goodarzi et al. (2019) is that wind forecast errors have a larger effect on the actual imbalance volume compared to solar forecast errors.

6.2.2. Imbalance price forecasting

As discussed in subsection 6.2.1, the literature available on imbalance price forecasting is extremely limited. The main conclusion drawn in the literature is that forecasting imbalance prices is extremely difficult and often not feasible. I have given it a try to forecast the imbalance prices an hour ahead, however, the results turned out extremely poor, such as shown in Figure 6.13.



(a) Downward price prediction for 2017

(b) Upward price prediction for 2017

Figure 6.13: The plots show the price predictions for the downward and upward prices separately. The results obtained are extremely poor and can be considered useless for making trading decisions.

The results are obtained by using a three-layer BN-skip model such as shown in subsection 6.1.6. Different architectures and models have been tried as well, but with the same poor results. As the model predictions are not good enough to be used for this research, it is chosen to leave the further results out of this study. Therefore, a different path is chosen. Instead of using a forecast of the price, the operating algorithm will use the available energy capacity to place bids based on the available capacity and a forecast of the coming imbalance state. For this use-case, it is chosen to opt solely for a neural network, as those have the option to output multiple values efficiently. Another option would be to opt for the Skforcast library (Rodrigo, n.d.), a Python library to create time series forecast with multi-step outputs, using Scikit regressors. To evaluate the performance of those regressors, more research is needed.

6.2.3. Deduced states

As covered in subsection 2.4.2 the imbalance market knows 4 states. The four states are upward, downward, dual-balancing, and no-balancing. For the model, we are mostly interested in forecasting downward and upward regulatory states. The reason is that in the case of dual-balancing either/both bids are activated anyway. The no-balancing state happens roughly 10% of the time per year, which is quite a large chunk. But to simplify the problem, it is decided to leave the no-balancing state out. As the goal is to simplify the problem, the y-values will be transformed into binary coding, where 0 corresponds to downward and 1 to upward. The dual-balancing states will be transformed to either 0 or 1 depending on what the dual-balancing state corresponds most with.

TenneT publishes 1-minute data with the activated upward and downward volume and the best price for each state. By recreating the logic used by TenneT to assign a state, we can convert the dual-balancing to the state that corresponds most with the TenneT logic. The state is not determined by the total volume activated during a PTU, but by the ramping of power plants. If the amount of upward power is increased and the amount of downward power is decreased at the end of the period, then we label it as an upward state (1). The other way around would be labeled as a downward state (0). Therefore, for every minute i inside the PTU, the upward power derivative $P_{ud,i}$and downward power derivative $P_{dd,i}$ are calculated. Next, the minute data is resampled to quarters, in the case of Dutch PTUs, whereby the last non-zero value for upward P_{RU} and downward P_{RD} are taken. If there is no non-zero value, which would either mean that

the amount of power required is stationary or that the power is. The latter is unlikely, as an imbalance is not likely to be stable. In those cases, the value of P_{RU} or P_{RD} is set to -999, such that the non-dual states are also assigned correctly, even when the power is reduced during the PTU.

To deduce state based on the minute data, the following code is used

Algorithm 5 Obtaining the balance state

$P_d \leftarrow$ downward volume
$P_u \leftarrow$ upward volumes
$m \leftarrow$ number of instances

for $i = 1, n$ **do**
  $P_{dd,i} \leftarrow P_{d,i-1} - P_{d,i-2}$
  $P_{ud,i} \leftarrow P_{u,i-1} - P_{u,i-2}$
end for

$P_{RD} \leftarrow$ Resample the downward instances to the corresponding minutes in a PTU by taking the last non zero value of P_{dd}
$P_{RU} \leftarrow$ Resample the upward instances to the corresponding minutes in a PTU by taking the last non zero value P_{ud}

for $q = 1, \text{length}(P_{RD})$ **do**
  if $P_{RD,q} = 0$ **then**
    $P_{RD,q} \leftarrow -999$
  end if
  if $P_{RU,q} = 0$ **then**
    $P_{RU,q} \leftarrow -999$
  end if
  if $P_{RU,q} < P_{RD,q}$ **then**
    $\mathrm{K}_q \leftarrow 1$
  else
    $\mathrm{K}_q \leftarrow 0$
  end if
end for
return Deduced states K

The algorithm shows a 98% overlap when comparing the actual upward and downward states with the deduced states K. The same logic is then applied to the situation with a dual-balancing state to obtain the state that corresponds most.

6.2.4. Above/under median price forecast

While it has been established that forecasting the prices directly is not yielding satisfactory results, forecasting the change that a price will be either 20% above or below the median could be beneficial, whereby 20% is chosen arbitrarily. To test this, two new y-values have been generated, namely

$$y_{u,t} = \begin{cases} 1 & \phi_t > M\vec{\phi}_{u,t-96,t} * 1.2 \\ 0 & \text{otherwise,} \end{cases} \tag{6.10}$$

and

$$y_{d,t} = \begin{cases} 1 & \phi_t < M\vec{\phi}_{d,t-96,t}/1.2 \\ 0 & \text{otherwise.} \end{cases} \tag{6.11}$$

$y_{u,t}$ indicates if the price will be 20% above the median upward price seen for the 96 upward prices. If we assume that 50% of the time the regulatory state is upward and 50% downward, then the $y_{d,t}$ would be calculated over the prices of the past two days. The prices for the downward state are ignored for calculating $y_{u,t}$. $y_{d,t}$ is an indication of the price at the time t will be 20% below the median of the 96 past downward prices. For $y_{d,t}$ it is assumed that the median of the prices is greater than zero.

6.2.5. Features

The features used for forecasting the imbalance state are similar to the features used in the day ahead model described in subsection 6.1.2 and are extended with new information that comes available during the day. Such an example is actual generation per source. For example, the actual wind and solar production up until now are known and the previous day's forecast for wind and solar power as predicted by the TSO is known. Those two variables are combined in a single volume deviation, where the actual generation is subtracted from the forecasted power, which we will call the Surprise Factor (SF). It must be noted that research by Dankers (2021) has shown that the actual values for solar and wind production, as provided by TenneT, are extremely poor. To increase the effectiveness of the algorithm used, further research could look into methods to obtain more accurate actual renewable power generation.

Another set of new variables are the Spark spread and the Dark spread. The Spark spread is the day-ahead prices minus the marginal costs of a gas-fired power plant. The Spark shows if gas-fired power plants are turning a profit. If the Spark is positive, likely, a lot of gas-fired power plants are fully generating power, and thus less steering capacity is available for the imbalance markets. The same goes for the Dark spread, this is the day ahead price minus the marginal costs of a coal-fired power plant. Other possibilities also exist, such as the Quark spread for nuclear power plants, but those are less relevant for the Netherlands.

Furthermore, the dataset is extended by adding the order book for the aFRR market, as introduced in subsection 2.4.2, weather data, and power plant unavailability. For the weather data, the temperature, wind speed, direction, global radiance, pressure, and humidity are incorporated. The wind speed and direction are combined into vector components as that could aid the model, for the same reason, that time-based features are modeled to a cyclical representation. The power plant unavailability is expressed as a percentage of the total installed capacity.

Lastly, the number of minutes aFRR has been activated in a previous PTU is included as a feature, combined with the derivative of the activated power per minute. If the derivative is positive, that would mean that the TSO is ramping up the power and thus that the next state is likely to be the same as the previous. If the derivative is zero it indicates that the balancing state has ended stable, however, it is unknown if the stability occurred at 0 MW imbalance or a higher/lower value. Therefore the last activated amount of power is added as well.

Again, the same neighboring countries are included as in the day ahead dataset. For the neighbors and the target country, the historical prices of up to 96 PTUs in the past are included. All other features, for which history is included, go 12 PTUs back, which (Bottieau et al., 2019) obtained to be the relevant amount of quarters to take into account for the Belgium market. Besides, the current dataset already contains over 2500 features, when considering every historical value included as a separate feature. Adding more information only increases the feature selection time.

The SuRF + TuRF algorithm is fitted onto the feature set for the three different y-values. The features that occurred for more than a single step of history are summarized in Table 6.11. Thus, it is likely that those time-series sequences do carry determining information about the imbalance state and price. Table 6.11 does not include the upward and downward price and the imbalance state, as those are obvious predictors for the y-values.

State	Upward	Downward
RLIa (DE, FR)	CUFa DE	RLIa 168 NL
PHPa DE	$RLIa_{24}$ (NL, DE, FR)	SF solar DK
HWRPa DE	$RLIa_{48}$ GB	SF Wind (BE, NL)
SF RLI (GB, DK)	SF solar (GB, DK, FR)	PHPa GB
Imbalance (DK)	SF-TPF BE	Imbalance (FR, BE, NL)
Solar actual (DK)	SF wind DE	Actual FOP DK
	PHPa (BE, DE)	
	HWRPa DE	
	Imbalance (NL, DE)	

Table 6.11: The most important features that are only known up and till the moment of forecasting.

In the table, the Pumped Hydro Power is denoted as PHP, Hydro Water Reservoir Power is HWRP and lastly, the FOP is fossil oil production. Other features, that are not time series, and important for the forecast, are summarized in Table 6.12.

State	Upward	Downward
$MA_{down,192}$ NL	Spark (BE, DE, DK, FR, NL)	Min300 NL
$MA_{up,192}$ GB	Pos100 NL	MinMax NL
$RLIf_{24}$ (DE, DK)	Pos300 NL	$RLIf_{168}$ NL
$RLIf_{168}$ DE	PosMax NL	Wu (NL, BE)
Absolute humidity FR	Dark (BE, DE, DK, FR, NL)	Wv NL
	$RLIf_{24}$ (DE, NL)	Cosine Hour
	$RLIf_{168}$ DE	Cosine Quarter
	$MA_{up,672}$ (NL, GB)	Pressure UK
	$MA_{down,672}$ NL	Total wind f NL
	CUFf (BE, DE)	

Table 6.12: The most important future known features for the different forecasts.

It is striking that most features are features from neighboring countries instead of from the Netherlands as one would expect. A possible explanation could be found in the data quality. All data used in this research is open-source and delivered by the TSO to the ENTSO-e platform. As mentioned before, the reporting quality by TenneT is extremely poor for renewable energy production. The quality is likely so bad, that the information on features of neighboring countries carries more information than the information provided by the TSO. Besides, it is likely that a wind forecast error in Germany also has an effect here, as both forecasts are made by the same European weather model. Further research could look into increasing the model accuracy by using better data quality.

6.2.6. Model architecture

The proposed architectures for this research compose of a 2-layer dense network and a simple LSTM with an additional input for exogenous variables, such that the layout has the structure as shown in Figure 6.14. The left arm of the neural network adds the variables known about the next states. Such variables could consist of the PTU number but also order book info, information that is known on forehand.

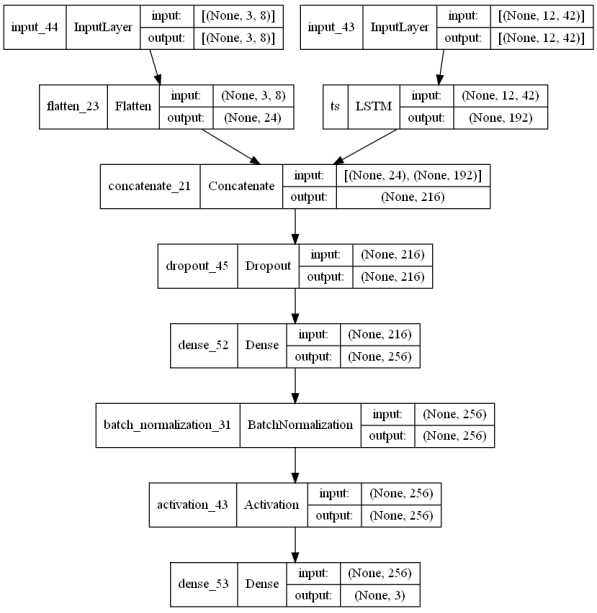


Figure 6.14: The LSTM model architecture combines the future unknown features after going through an LSTM and the future known features, before feeding the data into a simple Dense layer.

The LSTM parameters that have been tuned are summarized in Table 6.13, ϵ, β_1 and β_2 are tuned based on the findings by Choi et al. (2019). Literature does not define a specific search space for the different components. Kingma and Ba (2014) use the values of 0.99, 0.999 and 0.9999 for the β_2 component with the explanation that values closer to 1 are needed for sparse gradients. This optimization problem does not contain a sparse gradient, thus a range between 0.95 and 0.9999 is selected for the search space. ϵ is set by Keras to a default of 1e-7, but the tool-tip suggests a value of 1 for ImageNet (Keras, 2022b). Therefore, the search space is selected between 1e-7 and 1 for ϵ. For the other values, no real one-model fits all search space exists, which is thus selected based on experience.

	Search space			Optimal Values		
	Min	Max	Step	State	Upward price	Downward price
LSTM units	32	320	32	224	224	224
LSTM kernel regularizer (L1)	0.0001	0.075	-	0.0055	0.0055	0.0055
LSTM bias regularizer (L2)	0.0001	0.05	-	0.029	0.029	0.029
Dropout	0	0.5	-	0.24	0.24	0.24
Dense units	32	320	32	128	96	96
kernel initializer	-	-	-	HeNormal	HeNormal	HeNormal
Activation	-	-	-	Elu	elu	elu
Dense kernel regularizer (L1)	0.0001	0.075	-	0.0002	0.0002	0.0002
Dense bias regularizer (L2)	0.0001	0.075	-	0.058	0.058	0.058
Adam Beta-1	0.9	0.99	-	0.95	0.95	0.95
Adam Beta-2	0.95	0.9999	-	0.98	0.98	0.98
Adam Epsilon	1e-7	1	-	0.9	0.9	0.9

Table 6.13: The search space and optimal values for the LSTM deployed on the imbalance data

The hyperparameter tuning has been initialized with the same seed, for the three different sets (state, upward price and downward price). The upward and downward price forecast use the same feature set, while the state forecast uses slightly different features, as covered in subsection 6.2.5.

	Search space			Best values		
	Min	Max	Step	State	Upward price	Downward price
Dense-1 units	32	250	32	144	64	64
Dense-1 kernel regularizer (L1)	1e-4	0.075	-	0.04	0.0022	0.0022
Dense-1 bias regularizer (L2)	1e-4	0.1	-	0.037	0.096	0.096
Dropout	0	0.5	-	0.3	0.4	0.4
Dense units	32	300	32	96	160	160
kernel initializer	-	-	-	HeNormal	HeNormal	HeNormal
Activation	-	-	-	Elu	ReLu	ReLu
Dense-2 kernel regularizer (L1)	0.0001	0.075	-	0.07	0.011	0.011
Dense-2 bias regularizer (L2)	0.0001	0.075	-	0.028	0.028	0.028
Adam Beta-1	0.9	0.99	-	0.91	0.91	0.91
Adam Beta-2	0.95	0.9999	-	0.9992	0.95	0.95
Adam Epsilon	1e-7	1	-	0.035	0.9	0.9

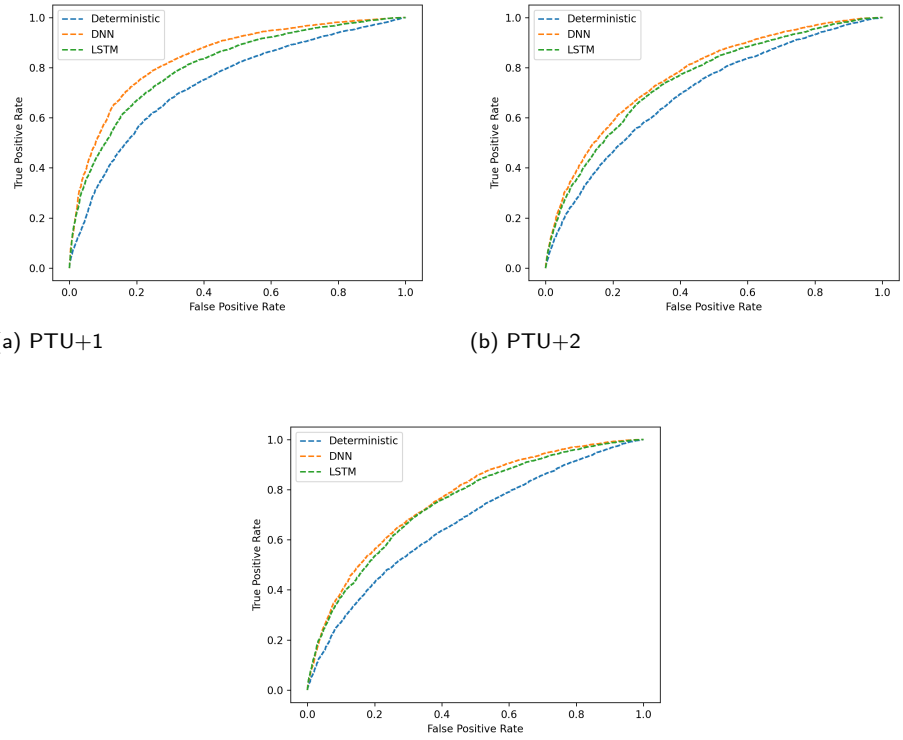
Table 6.14: The search space for the different Dense models shows the same values for the upward and downward above median price forecast.

The performance of the state forecasting model is summarized in Table 6.15. The state forecast uses an early stop of 20 epochs on the validation accuracy. whereby the validation set contained the last 20% of the 2015-2017 data. The deterministic approach already showed quite good results, especially for the next PTU. The models perform slightly higher on the PTU hereafter, but the performance can still be improved. Likely, the features do not carry enough information, as hypothesized in subsection 6.2.5.

	AUC - ROC			Accuracy [%]		
	+1	+2	+3	+1	+2	+3
naive	0.76	0.66	0.65	76	66	65
2-layer Dense	0.84	0.77	0.76	77	70	69
LSTM	0.81	0.75	0.75	74	69	69

Table 6.15: Forecasting the imbalance state using a LSTM does not aid the model over a simpler DNN.

Lastly, Figure 6.15 shows the ROC-curve for the forecast of the three next PTUs. The graph shows that the DNN lies more to the top left corners, implying that the DNN has larger true positive rates for equal false-positive rates. This holds across all probability thresholds.



(a) PTU+1

(b) PTU+2

(c) PTU+3

Figure 6.15: The ROC-curves show that the DNN especially outperforms the LSTM on the next PTU, while the PTU's after perform equally.

The results for the above-median upward price forecast 3-PTUs ahead are displayed in Table 6.16 and for the below downward price forecast 3-PTUs ahead is summarized in Table 6.17. In this case, the dataset is not evenly distributed, therefore the area under the PRC is used instead of the area under the ROC curve.

	Precision [%]	Recall [%]	Accuracy [%]	AUC - PRC
naive	49	49	61	0.44
2-layer Dense	50	70	62	0.57
LSTM	54	53	65	0.56

Table 6.16: For forecasting the above median price 3-PTUs ahead a DNN scores the saem AUC-PRC as an LSTM, and both outperform a naive approach on the AUC-PRC.

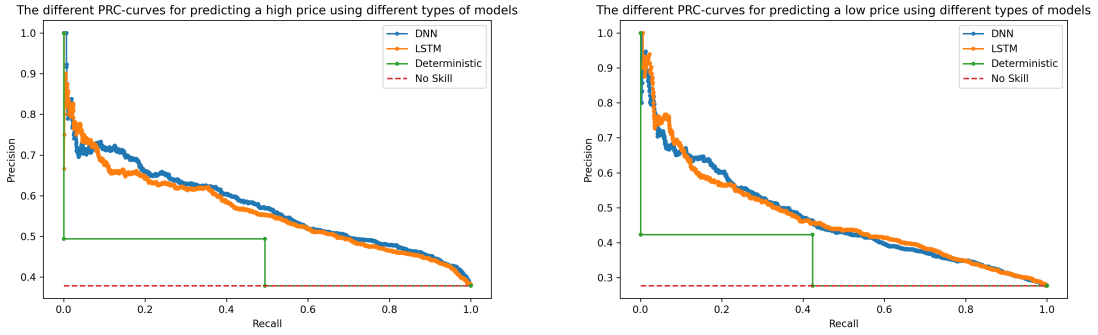
Table 6.16 shows that the accuracy of the LSTM is higher, but the recall is lower for estimating if a price will be 20% above the median price ($y_{u,t}$). The same pattern can be observed for the estimate of values 20% below the median downward price ($y_{d,t}$)., as shown in Table 6.17.

	Precision [%]	Recall [%]	Accuracy [%]	AUC - PRC
naive	42	42	68	0.34
2-layer Dense	36	73	57	0.47
LSTM	36	70	58	0.47

Table 6.17: For forecasting the below median price 3-PTUs ahead a DNN obtains a similair AUC-PRC as the LSTM, but both score better than the naive approach

However, the tables are not aiding the model decision. The forecasts will be used to maximize the profit

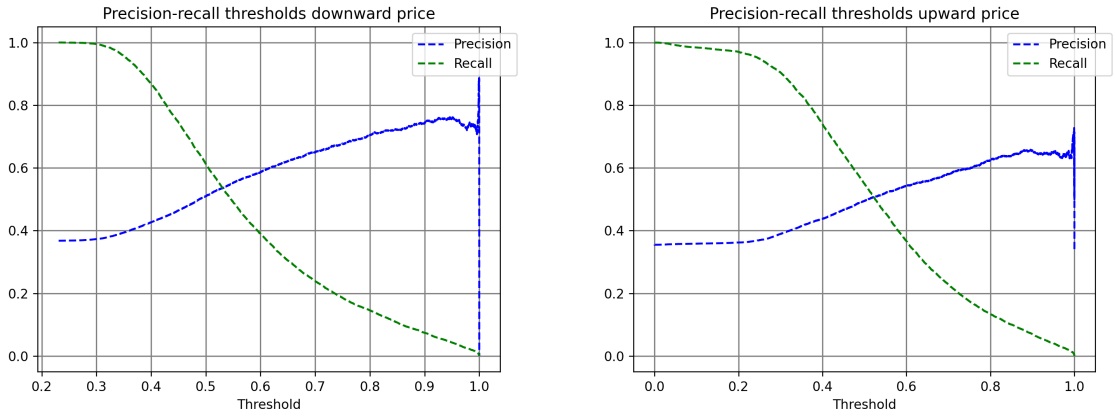
which makes it important to catch as many occasions as possible, with the highest possible precision. This means that a plot of the PRC needs to be evaluated, which is shown in Figure 6.16. The plots show that the forecast of the downward prices is less accurate compared to the forecast for the upward prices. This could indicate that the right features are not inside the dataset, or that the process is more prone to randomness.



(a) 20% above moving median upward price forecast (b) 20% below moving median downward price forecast

Figure 6.16: The different PR-curves show that the neural network outperforms the naive and deterministic approach. However, a lot of room for improvement is still available.

Furthermore, Figure 6.16a shows that for most regions the DNN curve is above the LSTM and both are above the deterministic and the probability approach. This shows that the models can capture more information than the simpler approaches. The same holds for the downward forecast in Figure 6.16b. However, the curve is closer to the deterministic approach. In both cases, the DNN either slightly outperforms or performs equally to the LSTM. Therefore the simpler DNN is chosen to be used for forecasting the state probabilities.



(a) The threshold curves for forecasting if the price will be 20% below the median downward price (b) The threshold curves for forecasting if the price will be 20% above the median upward price

Figure 6.17: The precision of the 20% above median upward price forecast barely reaches above 60%, likely rending the forecast quality too poor to be used.

Lastly, the threshold-precision and threshold-recall curves are plotted in Figure 6.17. The curve for the upward price prediction, plotted in Figure 6.17B shows that the highest obtainable precision is roughly at 60%. The downward outlier price predictions can be predicted with a higher precision of up to 75%. In both cases, the threshold for the operating algorithm is set to 0.8.

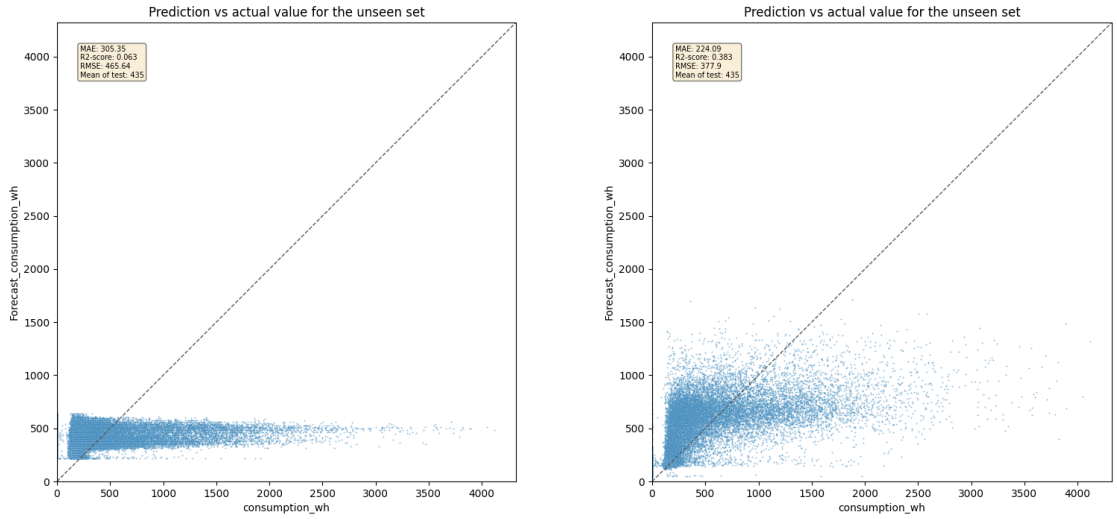
6.3. Load forecasting

6.3.1. General literature

Lastly, forecasting consumer demand has been researched to quite some extent with state-of-the-art results. However, as time is limited for this research, it has been chosen to not implement the best-performing forecasting algorithm but choose the simplest second best. Kim and Cho (2019) has shown that combining an LSTM with a CNN drastically reduces the forecasting error, but also shows that a linear regression model performs second best. As it is unlikely that a simple linear regression model would be able to capture any of the randomness induced by the consumer, it is chosen to also test an XGboost model.

6.3.2. consumer forecasts

For the consumer, four forecasts are required, two long-term and two short-term forecasts. For the long-term forecast, the hourly demand in Wh and the hourly maximum peak power of the next day are forecasted. The short-term forecast is a forecast of the 15-minute consumption and peak power for the quarter starting in half an hour. The consumer data is only available as of July 2018. This means that the initial training set is a lot smaller compared to the imbalance and day-ahead forecast. To maximize the usable forecasts, the model is initially trained on the 5 months of data from 2018. Afterward, the model is put into nearly the same framework as the other models, where the ADWIN algorithm monitors the errors. One of the differences is that the consumer forecast model is not tuned at every break, nor is a new feature search done as the feature set is limited. Instead, this model has 2 manually defined retrain points, once after 3 months into 2019 and once a half year into 2019.



(a) Forecast results using Lasso

(b) Forecast results using XGBoost

Figure 6.18: The forecast results for forecasting the demand for the next day of a random single consumer shows that both the Lasso and XGboost do perform quite poor

But, before we can run this algorithm for all 100 consumers, a model needs to be picked first. This is done by comparing a Lasso, which performed well in the research of Kim and Cho (2019) and an XGBoost algorithm that can deal with the non-linearity's likely present in the data. An indication of such non-linearity's can be observed in the results of the Lasso, plotted in Figure 6.18A. The lack of variance in the predictions shows that the model is not complex enough. Therefore, the XGBoost, with the results shown in Figure 6.18B, is chosen for the consumer forecasts.

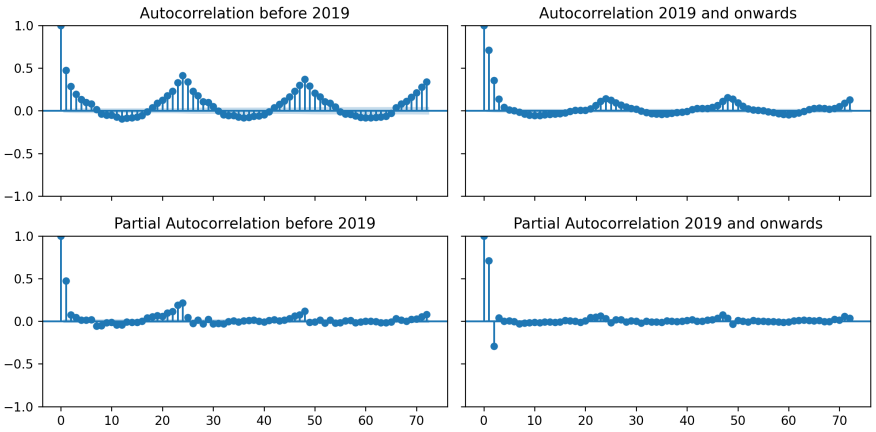


Figure 6.19: The autocorrelation and partial autocorrelation plot of a random consumer in the dataset show a change in the pattern likely attributed to COVID-19.

From the autocorrelation plot of a random user, shown in Figure 6.19, we can observe that the autocorrelation indicates that there is seasonality in the consumer data in 2019. However, the lockdown likely adjusted the consumer pattern and thus the recurring seasonal patterns. The partial autocorrelation plots show that only the first two lags are important for an ARIMA-based model. The patterns occurring will likely differ from consumer to the consumer depending on consumer behavior and types of electrical appliances and such.

When forecasting the consumer demand the models used were not able to pick up much of the patterns. Comparing the auto-correlation plot of the random consumer, with the aggregated group of all 100 consumers, as shown in Figure 6.20, a stronger indication of seasonality in the data can be observed. Also, the partial autocorrelation plot shows more lags that carry information, especially during the first 24 hours. The combination of a strong auto-correlation and recurring daily seasonal patterns likely aids the forecasting models.

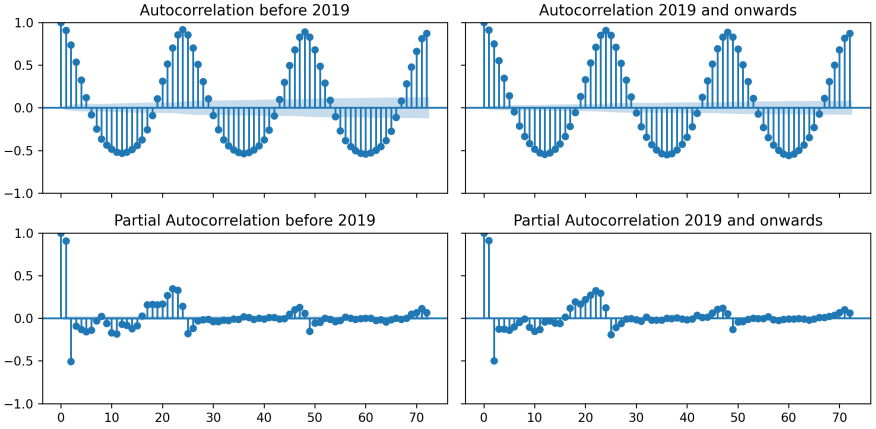
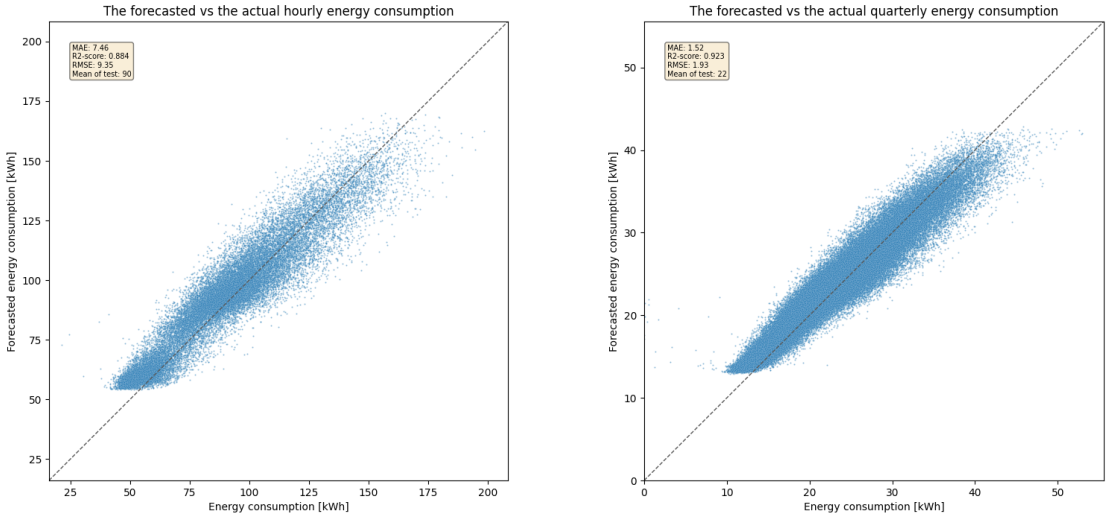


Figure 6.20: The (Partial) Autocorrelation plot of all the consumers aggregated

The simple aggregated model for both the short term and long term performs quite well, as shown in Figure 6.21. Figure 6.21A shows the long-term forecast results, the R^2 of 0.88 indicates that the model can explain most of the variance in the next-day consumer demand. Unsurprisingly, the short-term model, shown in Figure 6.21B has an even high performance, with a R^2 of 0.92.



(a) Long term forecast

(b) Short term forecast

Figure 6.21: The long-term and short-term forecasts of the aggregated consumers show that the simple model can predict relatively well.

It must be noted that the models are the nearly vanilla XGboost models, with default features as found in the python package. The variables changed based on manual observations are summarized in Table 6.18

Variable	**Value**
ETA	0.05
number of estimators	50
Maximum depth	5
Regularization Alpha	0.1
Regularization Lambda	0.75
Tree method	Hist
Enable Categorical	True

Table 6.18: The XGboost hyperparameters specifically changed for the consumer forecasts

7

Battery operations

This chapter starts with an overview of the best model setup in Section 7.1. Followed by a comparison of a naive aFRR bidding strategy and a forecast-based bidding strategy in Section 7.2. lastly, Section 7.3 covers the comparison of a neighborhood shared and an individually owned battery.

7.1. Model setup

At first, the naive model, that does not use any forecasts on the imbalance market, is assessed. At first a search for the optimal C-rate for the 2170 kWh stack of 155 TP2s is carried out. In reality, the c-rate is fixed for off-the-shelf batteries, but custom build batteries allow some degrees of freedom. Afterwards the optimal C-rate is implemented and used in the battery operations.

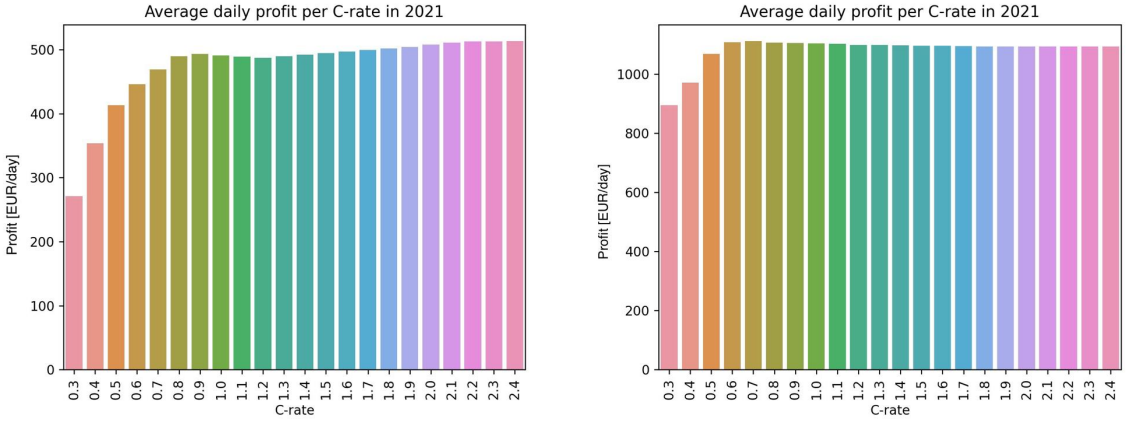
Furthermore, there could be occasions when the required power cannot be delivered by the battery. In those situations the power demand of the consumer is reduced. The implication of reducing the power for the consumer is that one of the electrical appliances will be shut off or at least reduced in power. Depending on the consumer, and the severity of the forecast error, that could implicate that an electrical stove cannot produce full power or that an electrical heater will temporally be lowered. The costs of this inconvenience is calculated based on the Value of Lost Load (opportunity costs) concept. The opportunity costs for the Netherlands is estimated to be 22.94EUR/kWh on average (Cambridge Economic Policy Associates, 2018). The opportunity cost (OC) is used to asses if use of a battery outweighs the required reduction of energy consumption. The opportunity cost is calculated based on the peak power that cannot be delivered, such that for a PTU the opportunity cost is

$$OC_{ptu} = (P_{MPD,ptu} - P_{MP}) * \delta/60 * 22.94, \tag{7.1}$$

where $P_{MPD,ptu}$ is the maximum power demand during a single *PTU*. P_{MP} is the maximum power that the battery can deliver and δ is the number of minutes in a single PTU. The opportunity cost calculated here is the maximum possible opportunity cost, but in reality the opportunity cost will be lower. The simulation does not account for the moment of power drawn but only looks at the peak powers during a single PTU. For example, if the aFRR requires full power 5 minutes into the PTU, and the peak demand of the consumer occurs at the first two minutes, the simulation will still assign an opportunity cost for the whole PTU, even though the maximum power requirement of the battery is never actually reached. This simplification has an effect that the opportunity costs will be overestimated.

7.1.1. Optimal battery configuration

Figure 7.1 shows a comparison of the average daily profits in 2021 for different c-rates for the aggregated group of consumers. Figure 7.1A shows a strong increase in the daily profit up to where the power and energy are equal, roughly at a profit of 500 EUR/day. For the battery including the consumer we calculate the battery benefit again, defined as the sum of the avoided costs, the Profit and Loss (PnL), defined as the revenues minus the incurred costs, of the imbalance and the PnL of the day-ahead operations.



(a) Not considering the consumers

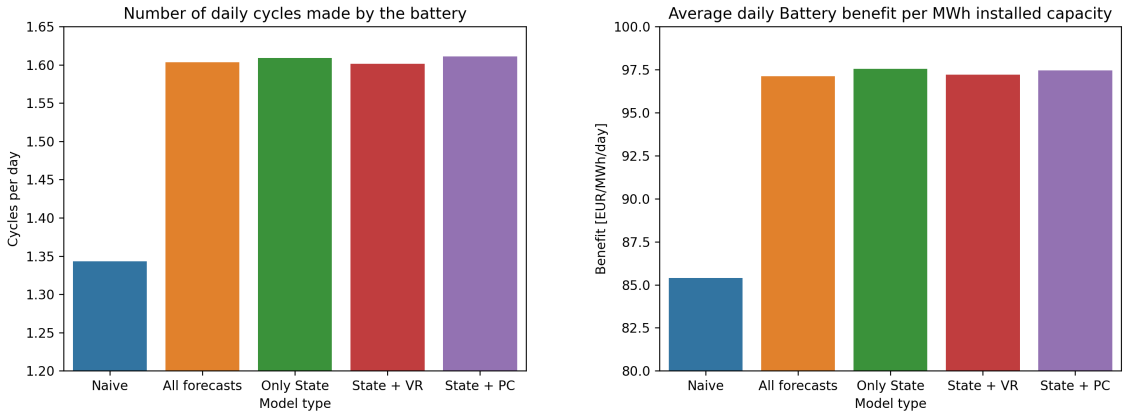
(b) Taking the aggregated consumers into account

Figure 7.1: The optimal C-rate for the naive approach not considering the consumers in 2022.

The results show that the profit stagnates when the consumers are included at a C-rate of 0.7, surprisingly this C-rate is lower than the optimal C-rate when the consumers are not included. This is surprising for the fact that the power needs to be shared with the consumer as well. Nonetheless, the c-rate of 0.7 will be used for the imbalance forecast-aided approach. In this naive approach the safety limits, as shown in Figure 5.1, on the imbalance part of the battery are removed. That battery part does not require extra safety limits, as the battery would only run outside those limits due to a miss-match on the Consumer and Day-Ahead (CDA) segment. That risk is already covered by the limits on the CDA-segment.

7.1.2. Forecast aided model comparison

The next step is to evaluate the forecast-aided models. For these models, the whole year of 2019 is used and the aggregated group of consumers. A single simulation of a year takes roughly 15 minutes, therefore only a single year is considered for the comparison analysis. Figure 7.2A shows the number of cycles the battery makes per day, where a single cycle is defined as fully (100% DoD) charging and discharging the battery. In the warranty of Tesla, they state that the battery should not be lower than 80% capacity after 10 years or 37.8 MWh of throughput, which converts to roughly 2700 cycles considering a 100% DoD (Tesla, 2017). Considering the 1.6 cycles per day, it would take a little less than 5 years to reach the end of the warranty period and the battery would likely be degraded to 80% of the initial capacity.



(a) The average daily cycles

(b) The average daily benefit in 2019

Figure 7.2: The average daily benefit. The naive model is the model that is not taking any risk, while the others use the state forecast to take on some extra risk. This extra risk also yields higher profits. The models are summarized in Table 7.1.

Figure 7.2B shows the average daily profit per MWh of installed capacity. The figure shows that the imbalance forecasts provide an extra 15% of profit on the imbalance trading activities. This can mainly be attributed to the fact that more volume is traded, which can be observed based on the increase in daily cycles, shown in Figure 7.2A. The added risk of more volume being traded is completely absorbed by both the implemented safety limit and considering a 60% DoD. The battery never goes outside the physical battery limits, the 100% DoD, but reaches close to the outer boundaries.

Model	**State forecast**		**Not bid based on price forecast**
	Reduce open bid	Not bid	
Naive			
All forecast	X	X	X
Only State	X		
State + VR	X	X	
State + PC	X		X

Table 7.1: The forecasts used, and the ways how, by the different aFRR bidding models.

The different models, and the forecasts used, are summarized in Table 7.1. The models that reduce/increase the price of the bids based on the chance that a price 20% above/below the median occurs (State + PC) and the model that reduces the volume based on the chance of an upward state occurring (State + VR) barely adds anything to the model performance, nor does it change anything about the opportunity costs. Therefore, it can be concluded that the naive aFRR bidding model aided by a state forecast (Only state) works best and will be used as standard implementation. The fact that the State + VR does not work is not surprising either as the model is using the same forecasts as the state forecast. Not placing a bid if the model is 80% sure that the state will be reached has relatively little impact, because otherwise the bidding algorithm would only consider 20% of the bid-in volume for calculating available volume for the next bid. Besides, there is still a 20% chance that the actual state will be of the opposite an opportunity that would have been missed by not bidding in at all.

7.2. Naive vs forecast model

The annual cash flows per market segment are shown in Figure 7.3, where in stripes the imbalance forecast aided model (only state) is shown. The imbalance cash flows only differ as the other inputs are the same for both models. The results show that the imbalance forecast-aided model increases the imbalance generated

profit for every year by at least 20%.

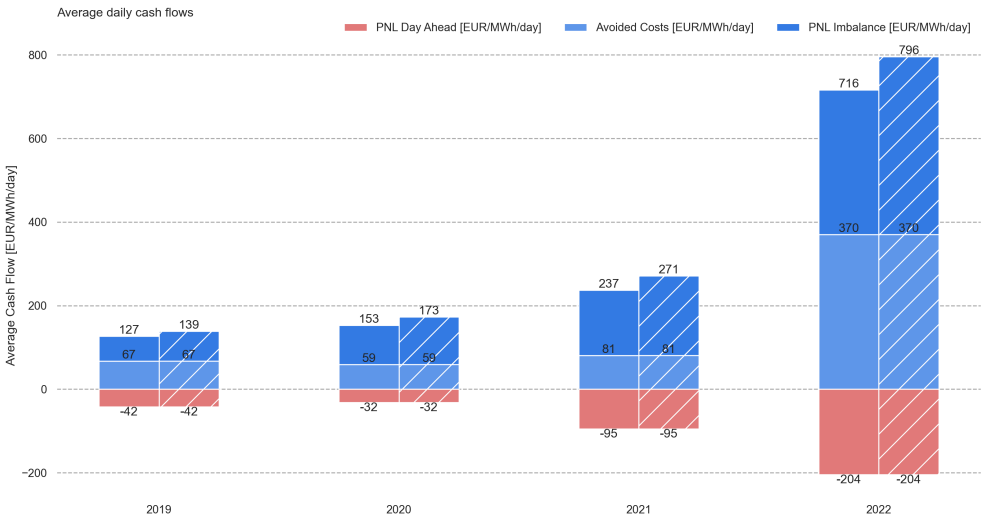
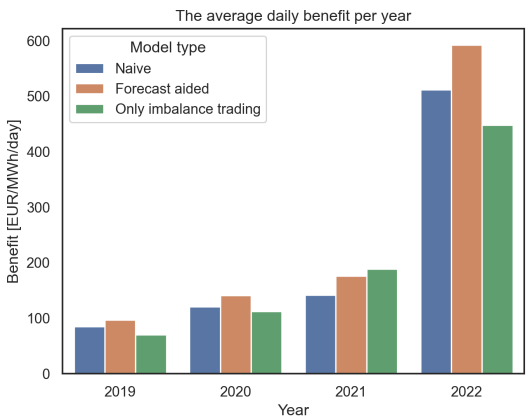
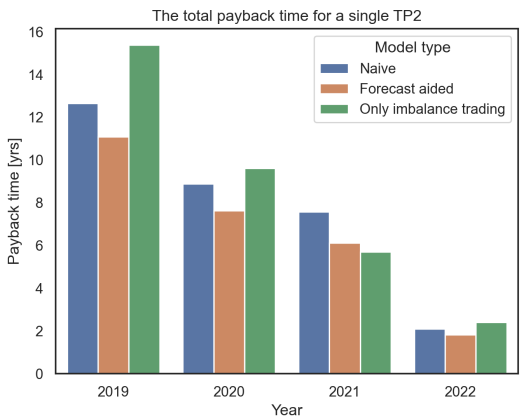


Figure 7.3: The daily average cash flows for the naive model and the imbalance forecast aided model (stripes) considering the aggregated group of consumers.

The battery benefit is the total sum of the bars shown in Figure 7.3, which are plotted in Figure 7.4 as a separate number. The results show that the benefit is increasing with time, even before the large price increase seen at the end of 2021. The time it would take to pay back a single TP2 is reduced from 11 years in 2019 up to less than two years in 2022. In the payback period, the capital cost for a TP2 is set to 5500 EUR, which equals roughly 390 EUR/kWh. A large storage unit might cost less per kWh as they would require only a single converter. The 5500 EUR does not account for the recent price spike in lithium, which rose by 70% in 2022 alone. Therefore the payback period can only be used as an indication.



(a) The yearly benefit



(b) Amount of years it would take to pay the battery back considering the benefit for that year

Figure 7.4: The benefit of owning a shared battery is increasing steadily over time, which thus reduces the payback time for such a storage unit.

Lastly, Figure 7.4 shows the daily profit that would have been generated if the battery was only used for imbalance trading. The profit is on average 20% higher by including the consumer as well. 2021 is an

exception, which can be attributed to the fact that the utility companies sold electricity for a cheaper price than the market price, as shown in Figure 4.11.

7.3. Individual consumer versus grouped consumers

Running the initial simulation of a single consumer showed that the battery had difficulties staying within its physical boundaries of the battery. This is caused by the weak performance of the individual demand forecast. To mitigate these issues, the limits on the imbalance part of the battery have been set at 30%-70% for the lower and upper limits. Furthermore, the bid price is increased to the maximum of 500 EUR/MWh and four times the rolling median if the volume in the balance capacity drops below 20%. If the balance energy reaches above 80% of the total imbalance energy capacity, the energy will be put onto the market for a price of 0 EUR/MWh. The measures, and the comparison between the individual and the aggregated group of consumers is summarized in Table 7.2. After the introduction of the extra measures the battery did not run outside the physical limits for the majority (60%) of the individual consumers, but it did not mitigate all cases. Figure 7.5 shows the the percentage of the overall time that the battery was outside the physical limits for each individual consumer. There is even a consumer whereby the battery spend 10% of the time outside the physical limits.

	Individual consumer	**Aggregated consumers**
Imbalance capacity (I_c) safety limits	30-70 %	20-80 %
I_c below 20% increase bid price to	500 EUR/MWh	N.a.
I_c above 80% increase ask price to	0 EUR/MWh	N.a.

Table 7.2: The different measures to keep the total battery energy within the physical boundaries of the battery.

The implemented measures will impact the maximum obtainable profit but are necessary to prevent the battery as good as possible from either running out of energy or not being able to store more. Running out of energy, or having too much would create an undesirable grid imbalance which is aimed to be prevented. It is unlikely that the battery would not be able to satisfy the voluntary aFRR bids due to a lack of energy or having too much, as the battery would already be stopped bidding if the limit drops below 30% of the imbalance capacity.

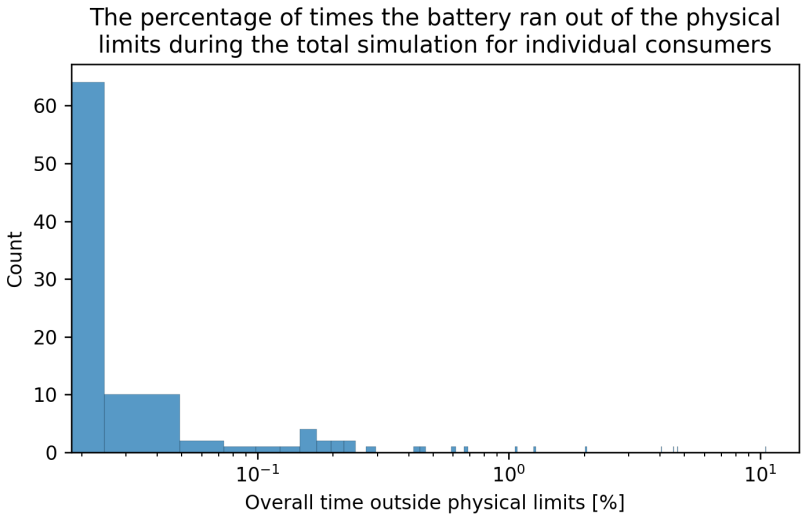
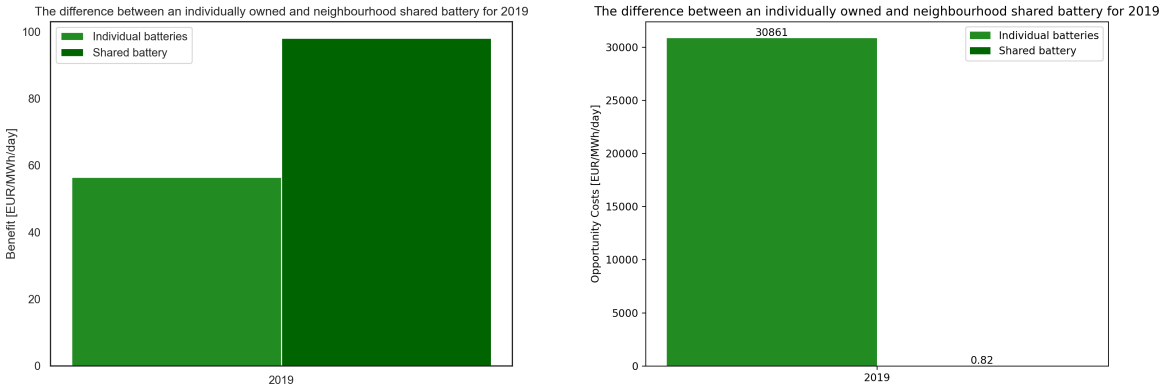


Figure 7.5: The percentage of time that the battery ran outside the physical limits during the simulation for each individual consumer.

Figure 7.6A shows the difference between the obtainable benefit for an individually owned battery versus a

neighborhood shared. In 2019 the daily average benefit per day MWh of installed capacity was almost twice as high for a neighborhood shared battery. The lower benefit can be attributed to less free energy for trading due to the lower demand forecast accuracy. This becomes especially clear when comparing the opportunity costs in Figure 7.6B. The opportunity costs for individual-owned batteries renders owning a BESS useless, as the consumer makes higher virtual costs by not being able to use a certain appliance than the benefit that having a battery generates.



(a) The benefit comparison

(b) Comparison of the opportunity costs

Figure 7.6: Not only the benefit of a shared battery is larger, the opportunity costs (not included in the benefit) is extreme for the individually owned battery.

Therefore, we can conclude that a neighborhood shared battery outperforms individually owned batteries. The shared battery can even be repaid promptly for all years considered. The extreme prices we currently see in the market repay the battery in as little as 2 years, but also the formerly 'normal' prices in 2019 and 2020 realize a payback period of 8-11 years, well within the life expectancy of the battery of 15 years

8

Conclusion and Discussion

This chapter concludes the research in Section 8.1, followed by a discussion of the assumptions made in Section 8.2. Lastly, recommendations for further research are summarized in Section 8.3.

8.1. Conclusion

This study set out to determine whether the use of home batteries is feasible without government subsidies. To do so, this study has built multiple fully automated machine learning algorithms and a battery operating algorithm, that can trade on different Dutch electricity markets while still satisfying the consumer's demand. The machine learning algorithms forecast the day-ahead price, the imbalance state, the consumer demand and power, and the probability of an above-median price. Those forecasts are combined in an operating algorithm, which makes decisions for the battery. The forecasted consumer energy demand is bought on the day-ahead market by solving a linear programming problem that maximizes the profit. The battery is virtually split into two, a part of the energy capacity is reserved for the consumer and a part for utilization on the aFRR market, while the available power is shared between the two. Therefore, the unused power is bid in on the secondary reserve market to optimally utilize the battery and generate an extra profit.

By applying the linear optimization algorithm solely to the day-ahead market and the consumer, I showed that the daily benefit, considering perfect foresight, from owning a Tesla Powerwall 2 (TP2), which has been used as a reference battery, was roughly 256EUR per year for a combined group of consumers before the price spike in electricity prices that started at the end of 2021. After the price spike, the benefit increased to 2600EUR per year, which results in a payback period of fewer than 3 years. The 10-fold in profit is a combination of larger price deltas on the day-ahead market and the large difference between the fee that a utility company charges and the price on the day-ahead market.

To alter the perfect foresight situation to a forecast based situation, two different machine learning algorithms have been compared for forecasting day-ahead prices, namely an XGBoost and a DNN. Their performances were assessed in a rolling window approach, where the expanding window proved significantly more accurate than a sliding window. For both approaches, trading based on the forecasts of the DNN yielded larger returns than trading based on the XGBoost forecasts. Trading based on the forecasts of the DNN resulted in a 20% higher return compared to the XGBoost on a year-to-year basis. The forecasts of the DNN resulted in an average return of 70% in the perfect foresight scenario.

As the data changed over time, concept drift is inevitable to occur, which degrades model performance over time. Therefore, the concept drift was monitored by the ADWIN algorithm, which checks the distribution of the model errors. If the ADWIN algorithm detected a change, then the model was retrained, which fully automates the machine learning process. However, in the time before ADWIN detects a change, some drift already occurs, for which this research has also implemented a bias correction algorithm. The neural network

was especially prone to this bias, but the correction algorithm was able to reduce the MAE by up to 40%.

For the feature selection, two different versions of the relief Algorithm have been tested, both aided and not aided by the TuRF elimination algorithm. It has been shown that the TuRF addition reduced the MAE by 8.5%. The difference between the SURF and MultiSURF has mainly been observed in the time it takes for a single fit, whereby the SURF + TuRF algorithm took half the time of the MultiSURF + TuRF. Furthermore, this research finds that only 3-70 features, out of the roughly 1500 available, are necessary to forecast the Dutch day-ahead prices. The large range in the required features comes from the change in difficulty of price forecasting. During certain period, only a small fraction of the features are necessary to accurately forecast the prices, as the day-to-day price cause remained the same. Out of the available features, the Capacity Utilization Factor (CUF) is one of the main drivers for price setting, as it obtained the highest feature ranking by the Relief Based Algorithm (RBA). It is shown that the Belgian, Dutch, and German CUF are some of the most important features.

Furthermore, this research shows that using a ResNet-inspired architecture for forecasting the day ahead prices aids the DNN model in both stability and performance. The proposed model architecture managed to further reduce the MAE by almost 10%. The stability, expressed in the standard deviation of the MAE, was equal to the simpler 2-layer dense network. In both cases the standard deviation of the MAE was 0.7% of the MAE. This is a large improvement from the instability experienced in prior work by Van Der Heijden et al. (2021).

On the subject of imbalance price forecasting, the research shows that forecasting imbalance prices does not result in useful predictions. Therefore, it was decided to forecast the imbalance state and whether the price will be 20% above/below the rolling daily median, whereby the rolling median is assumed to be greater than 0. The feature selection showed that the deviation of the actual generation of solar and wind from the previously forecasted generation is an important predictor. The production of pumped-hydro is another driver for extreme price forming and the spark and dark spread. The features are indicative of the availability of flexible steering sources, whereby the unavailability of such sources will lead to higher prices. An interesting observation is the lack of Dutch features, as there is an over-representation of features from neighboring countries. This might be due to poor data quality in the Netherlands. The introduction of higher quality data might also be beneficial for the model accuracy, which currently is performing comparable to the naive forecast method.

For forecasting the consumer demand a comparison was made between forecasts of an individual's and aggregated consumption patterns. Both the peak power and the energy consumption were forecasted on two different time frames: the day ahead consumption on an hourly basis and the quarterly consumption patterns half an hour ahead. For the forecasts, a simple Lasso and vanilla XGBoost were tested, where the Lasso showed too little variance in the forecasts. Therefore, the XGBoost was chosen as the main model. The comparison between the individual consumer forecast and the aggregated consumers showed a large difference. Where the individual consumer forecast was barely able to capture the demand patterns, the aggregated consumer forecast resulted in useful predictions.

The ability to forecast consumer demand turned out to be key for the operating algorithm. Predicting way too much or too little demand resulted in a large opportunity cost. This is reflected in the results of the operating algorithm, where the benefit generated by the aggregated battery is almost twice as high per MWh of installed capacity. Furthermore, aggregating all the consumers required fewer batteries and thus a smaller CAPEX. But most importantly, the opportunity cost of having an individual battery is many times higher than a neighborhood shared battery. Therefore, we can conclude that sharing a battery results in a larger benefit and a lower opportunity cost showing that it thus makes more sense to share a battery.

Lastly, a naive and forecast-aided bidding strategy for the aFRR market has been compared. The naive strategy places a bid based on the available capacity while taking prior placed bids into account. The forecast-aided strategy uses the probability that a PTU will be of a certain state. The obtained profits have been increasing year over year. In the period from 2019 until 2021, the annual profit for the naive strategy increased from 85EUR to 140 EUR, while the forecast aided strategy increased from 97 to 176. The profit that can be attributed to the trading on the imbalance alone, the forecast aided strategy generates a 20% extra profit. In 2022 the absolute profit increased, but the relative extra benefit of the forecast-aided strategy

stayed at 20%, this shows that the strategy keeps performing even though the market changes.

In conclusion, the research shows that owning a battery that can trade on the day ahead market, supply the consumer and can place bids on the voluntary aFRR market is a viable business model. Depending on which market prices will be used for future scenarios, the battery can be fully paid back in between 1.5 and 11 years, where the lowest payback times are directly related to the recent high electricity prices in the Netherlands.

8.2. Discussion

This research has made some simplification steps that could affect the results. The most important assumption is the assumption of being a price taker on both the day-ahead and the aFRR market. The day-ahead market has enough volume for this assumption to hold, but the aFRR market has less volume. However, the adjusted TP2 stack considered only has a total power of 1.5 MW, which is small enough for this assumption to likely hold as well, especially considering that every quarter has at least a volume of 300 MW. However, a larger BESS combination will likely have a significant effect on the merit order, therefore the values obtained cannot simply be scaled to a larger BESS.

The second assumption is the assumption that produced solar energy by the consumers is dumped on the grid without any intervention of the battery. This assumption comes from a lack of data on the production of solar PV. The effect of not taking the energy produced by the consumer into account is that also the extra benefit is ignored. All the production that is above the own consumption simply goes to waste. However, the penalty for creating an imbalance is also not taken into account, but this would be minimal as the battery would only create an imbalance due to solar production if the battery is full already. However, it might be possible that the solar production is larger than the consumption, and thus a larger battery is required, which would increase the CAPEX. Or the solar PV power generation could be curtailed, which could be done if the added CAPEX does not outweigh the extra benefit generated by the generated power. Lastly, the solar PV power will reduce the available power in the downward direction, and thus smaller bids would be placed. Therefore, it depends on whether the benefit of producing free energy outweighs the extra CAPEX and the reduction in available power during solar hours.

Furthermore, the aFRR forecast based bidding strategy used might not be the most optimal approach. The approach of reducing the open volume based on the chance that the state will occur likely works well with longer chains as the errors would average out. However, the chain considered here is relatively short with only 2 consecutive periods occurring. Therefore, this approach might not be the best solution.

Furthermore, this study ignores the power requirements of the aFRR. The minimum order size is 1 MW and likely can only be bid in multiples of 1. Furthermore, the minimum contract size for the day-ahead market is 0.1 MW, which is also not considered in this research. These requirements restrict the bids an actual user could place. This study allows more flexibility in the bids, and, therefore, somewhat overestimates the profits. However, this flexibility could be achieved in reality by combining with the bids of a larger portfolio of a utility company.

Moreover, for the aFRR, it is assumed that the TSO will fully use the ability of the battery to instantaneously deliver 100% of the sold power. However, this might not be instantaneous but follow a certain ramp rate. This assumption has a relatively small impact, because if the TSO only activates considering a certain ramp rate, then the TSO also deactivates with that same ramp rate. If the activation signal follows the pattern of an isosceles trapezoid, then the activated energy would be equal to a 100% ramp rate. The full activation of power starts later, but also ends later, resulting in an equal area under the curve. Therefore, this assumption does not affect the imbalance except when the power that has been bid in is not fully activated, or the activation signal does not follow the pattern of an isosceles trapezoid. In such a case, this assumption would result in a slight overestimation of the imbalance profit.

The next assumption made is that the battery does not degrade after a cycle and always stays at its original capacity. However, a battery's capacity decreases over time, limiting the available energy storage and thus the possible revenues.

Next, it is assumed that the TP2 can be configured to a different power to energy ratio, defined as the c-rate, which is fixed. The optimal c-rate turned out to be 0.7c, which differs from the 0.35c that the TP2 originally comes with. But this need not be an unrealistic assumption, as an actual operator could instead contract a different battery builder that sells a 0.7c grid-scale BESS, instead of buying 155 TP2s.

Furthermore, the study uses the assumption that activated peak power and activated aFRR always fall at the same moment within a PTU. However, the consumer peak demand may occur during the first minute of the PTU, while the aFRR will only be activated during the last minutes. This would mean that the power limits are not breached, and the power of the consumer is not reduced. The effect of this assumption is at most an overestimation of the opportunity costs.

Lastly, for the machine learning-based forecasts it is assumed that all the information used is available at the moment of forecasting. Almost all data is obtained through public sources and are thus prone to delivery issues. The data providers do not have any incentive to deliver the data in a timely or correct manner. Any information that is used in the forecast, which was not available at that time, likely overestimates the model performance. Therefore, in a real-world setting, it is recommended to use a more reliable alternative for the ENTSO-E data.

8.3. Further research

This research has left room for further research that could extend this work. At first, the simplifications listed in Section 8.2 could be addressed. Furthermore, in this research, it is chosen to only bid in on the aFRR and not to directly trade on the imbalance itself. Trading directly on the imbalance can be done by feeding or taking power from the grid which is not bought prior from a market party. The energy taken from or feed in is settled against the price that is set by the aFRR market. Doing so creates an uncertainty as the imbalance state could flip form a negative imbalance to a positive imbalance (imbalance state 2). During a state 2, the parties having an imbalance are charged/paid based on their net position which thus creates a risk. Therefore, using aFRR is a more save option for the battery owner, as they will never have to pay more than the price they bid. However, by trading directly on the imbalance, prices could be forecasted more accurately, as you need to forecast at most 15 minutes in advance, instead of the 45 minutes needed for the voluntary bidding on the aFRR.

Another point for further research is the optimal operating limits. In this research, a depth of discharge of 60% is considered to extend the battery life. However, it might be economically viable to further loosen those limits. Implementing a method of discounting future cash flows could allow for making decisions to (dis)charge the battery beyond those limits if financially beneficial.

Another improvement could be made in forecasting the time aFRR will be activated. In this research, it is assumed that, when calculating the available capacity, the aFRR will be activated for 15 minutes. It often occurs that the imbalance is solved within 15 minutes, and thus too much capacity has been reserved. By forecasting the length of activation the battery can bid more capacity and thus increases the revenue.

On the subject of consumer demand forecasting the state-of-the-art Temporal Fusion Transformer created by Lim et al. (2019) could be used. The model has shown some remarkable predictive power by allowing the combination of known and unknown data for both categorical and temporal data. Such a model will likely improve the forecast for the consumer demand, and might even be able to improve the other time series-based forecasts used in this research.

Lastly, the data available for forecasting the imbalance price and state was of poor quality. Replacing this data with more reliable data could improve the forecast quality and usability of those forecasts.

Bibliography

Abudu, K., Igie, U., Minervino, O., & Hamilton, R. (2021). Gas turbine efficiency and ramp rate improvement through compressed air injection. *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, *235*(4), 866–884.

AGSI. (2022). https://agsi.gie.eu/#/graphs/eu

Amjady, N., & Keynia, F. (2009). Day-ahead price forecasting of electricity markets by a new feature selection algorithm and cascaded neural network technique. *Energy Conversion and Management*, *50*(12), 2976–2982.

Aurelien, G. (2019). *Hands-on machine learning with scikit-learn and tensorflow: Concepts, tools, and techniques to build intelligent systems*. OReilly.

Aurélien, G. (2017). Training deep neural networks. *Hands-on machine learning with scikit-learn and tensorflow: Concepts, tools, and techniques to build intelligent systems* (pp. 333–335). O'Reilly Media.

Banerjee, P. (2020). A guide on xgboost hyperparameters tuning. https://www.kaggle.com/code/prashant111/a-guide-on-xgboost-hyperparameters-tuning/notebook

Baumgarte, F., Glenk, G., & Rieger, A. (2020). Business models and profitability of energy storage. *Iscience*, *23*(10), 101554.

Benedetti, M., Cesarotti, V., Introna, V., & Serranti, J. (2016). Energy consumption control automation using artificial neural networks and adaptive algorithms: Proposal of a new methodology and case study. *Applied Energy*, *165*, 60–71.

Bichler, M., Buhl, H. U., Knörr, J., Maldonado, F., Schott, P., Waldherr, S., & Weibelzahl, M. (2022). Electricity markets in a time of change: A call to arms for business research. *Schmalenbach Journal of Business Research*, 1–26.

Bifet, A., & Gavalda, R. (2007). Learning from time-changing data with adaptive windowing. *Proceedings of the 2007 SIAM international conference on data mining*, 443–448.

Bottieau, J., Hubert, L., De Grève, Z., Vallée, F., & Toubeau, J.-F. (2019). Very-short-term probabilistic forecasting for a risk-aware participation in the single price imbalance settlement. *IEEE Transactions on Power Systems*, *35*(2), 1218–1230.

Brownlee, J. (2017). Lift performance with learning rate schedules. *Deep learning with python* (pp. 98–104). Machine Learning Mastery,

Brownlee, J. (2019). *A gentle introduction to dropout for regularizing deep neural networks*. Retrieved June 11, 2022, from https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/

Brownlee, J. (2021). Gentle introduction to the adam optimization algorithm for deep learning. https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

Cambridge Economic Policy Associates. (2018).

Cappellen, L. v., Jongsma, C., & Rooijers, F. (2022). Het net slimmer benut!

CBS. (2022). *Elektriciteitsbalans; aanbod en verbruik*. Retrieved March 5, 2022, from https://opendata.cbs.nl/statline/#/CBS/nl/dataset/84575NED/table?dl=66160

Celik, B., & Vanschoren, J. (2021). Adaptation strategies for automated machine learning on evolving data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(9), 3067–3078.

Chen, B. (2020). *Learning rate schedule in practice: An example with keras and tensorflow 2.0*. https://towardsdatascience.com/learning-rate-schedule-in-practice-an-example-with-keras-and-tensorflow-2-0-2f48b2888a0c

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Choi, D., Shallue, C. J., Nado, Z., Lee, J., Maddison, C. J., & Dahl, G. E. (2019). On empirical comparisons of optimizers for deep learning. *arXiv preprint arXiv:1910.05446*.

Compendium voor de Leefomgeving. (2020). https://www.clo.nl/indicatoren/nl0589-temperatuur-extremen

CS231n. (2022). *Convolutional neural networks for visual recognition*. Retrieved March 26, 2022, from https://cs231n.github.io/neural-networks-3/

Dankers, K. E. (2021). *Forecasting the country wide total wind power using a top-down method*.

Deloitte. (2019). *Assessing the flexibility of coal-fired power plants for the integration of renewable energy in germany*.

Docurbs. (2018). *Relief wiki*. https://commons.wikimedia.org/w/index.php?curid=68142587

Dumas, J., Boukas, I., de Villena, M. M., Mathieu, S., & Cornélusse, B. (2019). Probabilistic forecasting of imbalance prices in the belgian context. *2019 16th International Conference on the European Energy Market (EEM)*, 1–7. https://doi.org/10.1109/EEM.2019.8916375

Electrical Technology. (2021). Classification of electric power distribution network systems. https://www.electricaltechnology.org/2021/10/electric-power-distribution-network.html

Engels, J., Claessens, B., & Deconinck, G. (2017). Combined stochastic optimization of frequency control and self-consumption with a battery. *IEEE Transactions on Smart Grid*, *10*(2), 1971–1981.

Engels, J., Claessens, B., & Deconinck, G. (2019). Techno-economic analysis and optimal control of battery storage for frequency control services, applied to the german market. *Applied Energy*, *242*, 1036–1049.

Englberger, S., Jossen, A., & Hesse, H. (2020). Unlocking the potential of battery storage with the dynamic stacking of multiple applications. *Cell reports physical science*, *1*(11), 100238.

ENTSO-E. (n.d.). Policy 1: Load-frequency control and performance.

ENTSO-e. (2022). Central collection and publication of electricity generation, transportation and consumption data and information for the pan-european market. https://transparency.entsoe.eu/generation/r2/installedGenerationCapacityAggregation/show

EPEX. (2022). *Guarantees of origin: First pan-european spot market in 2022*. Retrieved April 6, 2022, from https://www.epexspot.com/en

Essent. (2022). *Energieprijzen voor stroom en gas van essent: Essent*. Retrieved March 3, 2022, from https://www.essent.nl/content/particulier/producten/alle_energietarieven/tarieven-stroom-en-gas/modelcontract-elektriciteit-en-gas-variabel.html

F, B. (2021). Lithium-ion battery care guide - part four. https://cleantechnica.com/2021/05/27/lithium-ion-battery-care-guide-part-four/

Gibescu, M., van Zwet, E. W., Kling, W. L., & Christie, R. D. (2008). Optimal bidding strategy for mixed-portfolio producers in a dual imbalance pricing system. *Proc. 16th Power Systems Computation Conference*.

Giga Storage B.V. (2021). Informatiememorandum giga storage b.v.

Gonçalves Jr, P. M., de Carvalho Santos, S. G., Barros, R. S., & Vieira, D. C. (2014). A comparative study on concept drift detectors. *Expert Systems with Applications*, *41*(18), 8144–8156.

Goodarzi, S., Perera, H. N., & Bunn, D. (2019). The impact of renewable energy forecast errors on imbalance volumes and electricity spot prices. *Energy Policy*, *134*, 110827.

Greene, C. S., Penrod, N. M., Kiralis, J., & Moore, J. H. (2009). Spatially uniform relieff (surf) for computationally-efficient filtering of gene-gene interactions. *BioData mining*, *2*(1), 1–9.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hugenholtz, D. E. (2020). *Batteries and energy arbitrage, a techno-economic analysis of electricity arbitrage opportunities for utility-scale battery energy storage in the netherlands*.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International conference on machine learning*, 448–456.

Jakaša, T., Andročec, I., & Sprčić, P. (2011). Electricity price forecasting—arima model approach. *2011 8th International Conference on the European Energy Market (EEM)*, 222–225.

Jauhari, P. (2019). Gradient boosting, a birds eye view into one of the most widely used ml algorithms. %7Bmedium.com/analytics-vidhya/gradient-boosting-a-birds-eye-view-into-widely-used-ml-algorithm-part-2-6684c68fc262%7D

Karim, R. (2018). Retrieved June 11, 2022, from https://towardsdatascience.com/intuitions-on-l1-and-l2-regularisation-235f2db4c261

Kavikondala, A., Muppalla, V., Krishna Prakasha, K., & Acharya, V. (2019). Automated retraining of machine learning models. *Int J Innovat Technol Explor Eng*, *8*(12), 445–452.

Keles, D., Scelle, J., Paraschiv, F., & Fichtner, W. (2016). Extended forecast methods for day-ahead electricity spot prices applying artificial neural networks. *Applied energy*, *162*, 218–230.

Keras. (2022a). *Batchnormalization layer*. Retrieved January 10, 2022, from https://keras.io/api/layers/normalization_layers/batch_normalization/

Keras. (2022b). *Keras documentation: Adam*. Retrieved February 23, 2022, from https://keras.io/api/optimizers/adam/

Kim, T.-Y., & Cho, S.-B. (2019). Predicting residential energy consumption using cnn-lstm neural networks. *Energy*, *182*, 72–81.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kira, K., Rendell, L. A. et al. (1992). The feature selection problem: Traditional methods and a new algorithm. *Aaai*, *2*(1992a), 129–134.

Klæboe, G., Eriksrud, A. L., & Fleten, S.-E. (2015). Benchmarking time series based forecasting models for electricity balancing market prices. *Energy Systems*, *6*(1), 43–61.

Kolcun, R., Popescu, D. A., Safronov, V., Yadav, P., Mandalari, A. M., Xie, Y., Mortier, R., & Haddadi, H. (2020). The case for retraining of ml models for iot device identification at the edge. *arXiv preprint arXiv:2011.08605*.

Kolter, J. Z., & Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *The Journal of Machine Learning Research*, *8*, 2755–2790.

Kononenko, I. (1994). Estimating attributes: Analysis and extensions of relief. *European conference on machine learning*, 171–182.

Kononenko, I., & Sĭkonja, M. R. (2007). *Non-myopic feature quality evaluation with (r) relieff*. Chapman; Hall/CRC.

Kononenko, I., Šimec, E., & Robnik-Šikonja, M. (1997). Overcoming the myopia of inductive learning algorithms with relieff. *Applied Intelligence*, *7*(1), 39–55.

Krawczyk, B., Minku, L. L., Gama, J., Stefanowski, J., & Woźniak, M. (2017). Ensemble learning for data stream analysis: A survey. *Information Fusion*, *37*, 132–156.

Lago, J., De Ridder, F., & De Schutter, B. (2018). Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms. *Applied Energy*, *221*, 386–405.

Lim, B., Arik, S. O., Loeff, N., & Pfister, T. (2019). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *arXiv preprint arXiv:1912.09363*.

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., & Zhang, G. (2018). Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 1–1. https://doi.org/10.1109/tkde.2018.2876857

Mahajan, T., Singh, G., Bruns, G., Bruns, G., Mahajan, T., & Singh, G. (2021). An experimental assessment of treatments for cyclical data.

Maheshwari, A., Paterakis, N. G., Santarelli, M., & Gibescu, M. (2020). Optimizing the operation of energy storage using a non-linear lithium-ion battery degradation model. *Applied Energy*, *261*, 114360.

McKay, H., Griffiths, N., Taylor, P., Damoulas, T., & Xu, Z. (2020). Bi-directional online transfer learning: A framework. *Annals of Telecommunications*, *75*(9), 523–547.

McMahan, H. B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D., et al. (2013). Ad click prediction: A view from the trenches. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1222–1230.

Metz, D., & Saraiva, J. T. (2018). Use of battery storage systems for price arbitrage operations in the 15-and 60-min german intraday markets. *Electric Power Systems Research*, *160*, 27–36.

Milieu Centraal. (2022). Gemiddeld energieverbruik: Is jouw verbruik hoog of laag? https://www.milieucentraal.nl/energie-besparen/inzicht-in-je-energierekening/gemiddeld-energieverbruik/

Monteiro, C., Ramirez-Rosado, I. J., Fernandez-Jimenez, L. A., & Conde, P. (2016). Short-term price forecasting models based on artificial neural networks for intraday sessions in the iberian electricity market. *Energies*, *9*(9), 721.

Moore, J. H., & White, B. C. (2007). Tuning relieff for genome-wide genetic analysis. *European conference on evolutionary computation, machine learning and data mining in bioinformatics*, 166–175.

Narajewski, M., & Ziel, F. (2020). Econometric modelling and forecasting of intraday electricity prices. *Journal of Commodity Markets*, *19*, 100107.

Netbeheer Nederland. (2022). Capaciteitskaart invoeding elektriciteitsnet. https://capaciteitskaart.netbeheernederland.nl/

Next Kraftwerke Benelux B.V. (2022). *Primaire reserve (fcr) - wat is het? - next-kraftwerke.nl*. Retrieved February 26, 2022, from https://www.next-kraftwerke.nl/kennis/primaire-reserve-fcr

Office of Energy Efficiency and Renewable Energy. (2017). *Confronting the duck curve: How to address over-generation of solar energy*. Retrieved June 12, 2022, from https://www.energy.gov/eere/articles/confronting-duck-curve-how-address-over-generation-solar-energy

Park, S. (2021). A 2021 guide to improving cnns-optimizers: Adam vs sgd. https://medium.com/geekculture/a-2021-guide-to-improving-cnns-optimizers-adam-vs-sgd-495848ac6008

Parmezan, A. R. S., Souza, V. M., & Batista, G. E. (2019). Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. *Information sciences*, *484*, 302–337.

Peltarion. (2022a). *Adam*.

Peltarion. (2022b). *What is the optimizer adadelta?* Retrieved April 12, 2022, from https://peltarion.com/knowledge-center/documentation/modeling-view/run-a-model/optimizers/adadelta

Perez-Arriaga, I. (2016). Perfect competition. *Regulation of the power sector* (pp. 67–95). Springer London Ltd.

Perez-Arriaga, I. J. (2016). The law of supply and demand. *Regulation of the power sector* (pp. 65–66). Springer London Ltd.

Poirot, T., & Baschet, C. (2021). *Inside europe's newest frequency response opportunity for energy storage, afrr*. Retrieved December 8, 2021, from https://www.energy-storage.news/inside-europes-newest-frequency-response-opportunity-for-energy-storage-afrr/

PowerTech. (2021). *Lithium-ion battery advantages*. Retrieved December 16, 2021, from https://www.powertechsystems.eu/home/tech-corner/lithium-ion-battery-advantages/

Prabhakaran, S. (2019). *Augmented dickey fuller test (adf test)*. Retrieved June 23, 2022, from https://www.machinelearningplus.com/time-series/augmented-dickey-fuller-test/

Robnik-Šikonja, M., & Kononenko, I. (1997). An adaptation of relief for attribute estimation in regression. *Machine Learning: Proceedings of the Fourteenth International Conference (ICML'97)*, *5*, 296–304.

Robnik-Šikonja, M., & Kononenko, I. (2003). Theoretical and empirical analysis of relieff and rrelieff. *Machine learning*, *53*(1), 23–69.

Rodrigo, J. A. (n.d.). Skforecast. https://joaquinamatrodrigo.github.io/skforecast/0.4.3/index.html

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). *Learning internal representations by error propagation* (tech. rep.). California Univ San Diego La Jolla Inst for Cognitive Science.

Sauvage, J., & Bahar, H. (2013). Cross-border trade in electricity and the development of renewables-based electric power: Lessons from europe. *OECD Trade and Environment Working Papers*. https://doi.org/10.1787/5k4869cdwnzr-en

scikit-learn. (2022). *Time-related feature engineering*. Retrieved February 16, 2022, from https://scikit-learn.org/stable/auto_examples/applications/plot_cyclical_feature_engineering.html

Statistics Netherlands. (2022). *Gemiddelde energietarieven voor consumenten* [Dataset]. Retrieved May 25, 2022, from https://opendata.cbs.nl/#/CBS/nl/dataset/84672NED/table

TenneT. (2018). Productinformation afrr.

TenneT. (2020). Imbalance pricing system. https://www.tennet.eu/fileadmin/user_upload/SO_NL/Imbalance_pricing_system.pdf

TenneT. (2021a). Retrieved November 24, 2021, from https://www.tennet.org/english/operational_management/system_data_preparation/offering_regulating_reserve_capacity/bid_price_ladder.aspx

TenneT. (2021b). Product information mfrrda (incident reserve).

TenneT. (2022a). Handboek afrr voor bsps.

TenneT. (2022b). Handboek fcr voor bsps.

TenneT. (2022c). *Our grid*. Retrieved January 3, 2022, from https://www.tennet.eu/our-grid/

Tesla. (2017). Tesla powerwall warant (european warranty region).

Tesla Inc. (2018). https://www.tesla.com/sites/default/files/pdfs/powerwall/Powerwall%5C%202_AC_Datasheet_nl.pdf

Uniejewski, B., Marcjasz, G., & Weron, R. (2019). Understanding intraday electricity markets: Variable selection and very short-term price forecasting using lasso. *International Journal of Forecasting*, *35*(4), 1533–1547.

Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S., & Moore, J. H. (2018). Relief-based feature selection: Introduction and review. *Journal of biomedical informatics*, *85*, 189–203.

Urbanowicz, R. J., Olson, R. S., Schmitt, P., Meeker, M., & Moore, J. H. (2018). Benchmarking relief-based feature selection methods for bioinformatics data mining. *Journal of biomedical informatics*, *85*, 168–188.

Van Der Heijden, T., Lago, J., Palensky, P., & Abraham, E. (2021). Electricity price forecasting in european day ahead markets: A greedy consideration of market integration. *IEEE Access*, *9*, 119954–119966.

Van der Munckhof, G. (2020). *Forecasting river discharge using machine learning methods*.

Visser, L., AlSkaif, T., & Van Sark, W. (2020). The importance of predictor variables and feature selection in day-ahead electricity price forecasting. *2020 International Conference on Smart Energy Systems and Technologies (SEST)*, 1–6.

Vugt, B. v. (2020). *Verification methods for delivery of ancillary services provided by wind assets* (Doctoral dissertation).

Wang, K., Xu, C., Zhang, Y., Guo, S., & Zomaya, A. Y. (2017). Robust big data analytics for electricity price forecasting in the smart grid. *IEEE Transactions on Big Data*, *5*(1), 34–45.

Wu, L., & Shahidehpour, M. (2010). A hybrid model for day-ahead price forecasting. *IEEE Transactions on Power Systems*, *25*(3), 1519–1530.

Wyk, A. v. (2018). Encoding cyclical features for deep learning. https://www.avanwyk.com/encoding-cyclical-features-for-deep-learning

Ye, A. (2020). Xgboost, lightgbm, and other kaggle competition favorites. https://towardsdatascience.com/xgboost-lightgbm-and-other-kaggle-competition-favorites-6212e8b0e835

Ziel, F. (2017). Modeling the impact of wind and solar power forecasting errors on intraday electricity prices. *2017 14th International Conference on the European Energy Market (EEM)*, 1–5.

Žliobaitė, I., Pechenizkiy, M., & Gama, J. (2016). An overview of concept drift applications. *Big data analysis: new algorithms for a new society*, 91–114.