**One-page abstract: summarizes the entire report, insights, conclusions and proposal**

**(a) Problem description (business goal and data mining goal)**

Proposal + better relevance: high costs if promised electricity cant be delivered at intraday market

**(b) Data description**

The dataset to be examined includes volume (in MWh) and price (in €/MWh) of the energy sold at each auction sale. For the day-ahead auction the data is provided in hourly-, for the intraday auction in 15-minute intervals. As a whole the dataset encompasses 731 days, two years, which results in 17544 data instances for the day-ahead data and 70176 instances for the intraday auction data during the years 2016 and 2017. The very first step was to scrape the needed market data from the website <https://www.epexspot.com/de/>. After the successful collection of day-ahead and intraday auction price and volume information, the hourly data was converted into 15-minute increments and attached to the data file of the intraday auction data. Since there is no missing data, no respective preparations were required.[[1]](#footnote-1) As a unique identifier of each row in the merged data file, a timestamp is used.



**(c) Data preparation**

In the following it is described which steps the data preparation included. In addition to the scraped electricity data, some feature values needed to be added. As one of these relevant features the German holidays were scraped from the page <https://feiertage-api.de>. This data later allows conclusions to be drawn as to whether electricity prices and consumption are dependent on public holidays. In our dataset the holidays are reflected by a boolean value, which indicates whether a day was a holiday or not. Furthermore, two other features specify the days distance of the observed day from the last and the next holiday. Another added feature is the number of participants at the market, the electricity providers. Of course, they also have some influence on the price and the price premium for a certain day 🡪 SOURCE. The data for the participants were scraped from the page <https://www.acer-remit.eu/portal/register-download?fileType=XML&euregId=> on a monthly level. Last but not least, various weather data was scraped, as the weather also has an enormous influence on the electricity price of the market, especially concerning renewable electricity generation. The monthly data was scraped from the website <https://www.weatheronline.de/Deutschland/> for all available German locations and then averaged per each month. From these weather data the following features were added to the dataset: Daily average temperature in Celsius, daily maximum temperature in Celsius, daily minimum temperature in Celsius, number of freezing days, number of icy days, monthly rain volume in mm, number of rainy days, the average of daily sunshine hours, the average monthly wind speed in km/h, the monthly snow days and the average of the daily snowfall in cm.

Nextly, the data hat to be interpolated. This addresses the circumstance that all data had to be adjusted to the same 15-minute granularity. For this a series with a 15-minute granularity was created, which was later connected as a dataframe with the actual data by a left merge via the respective datetime features to a dataframe. The result of this merge was a dataframe with a 15-minute granularity, but the features were only filled for every full hour. This was solved by using a forward fill. After this, all the still unrelated files of scraped data could be merged into a large file via the datetime column.

Also, the price premium, which is calculated by the difference of the day-ahead price and the intraday price at time t, was added. The following formula represents this calculation:

To fix notation, let fit denote the electricity day-ahead price observed on day t for delivery during hour i of day t + 1, and let Si,t+1 denote the intraday price for hour i of day t + 1.

An analysis of the scraped weather data revealed that the data containing information about the monthly number of days with snow and the monthly snow volume had to be discarded, due to extreme data scarcity causing the aggregated data to be too biased to be useful. Additionally, the data containing information about the average minimum and maximum daily temperature per month had to be removed from the dataset. The reason for this step was that the named data did not add valuable information in addition to the average of the daily mean temperature per month and leaving it would cause unnecessary multicollinearity.

For scaling the StandardScaler was used because the price data has some strong outliers which would cause problems with some other scalers and scaled data is denser than when using RobustScaler.

The previous described data set is used for the descriptive part of the task. In order to prepare the dataset for the predictive approach, the data had to be applied furthermore. Initially, the specification of the datetime was dropped and instead represented by one hot encoding of the respective month, weekday and hour. Furthermore, the original price premium was removed and replaced by a categorical value. This binary value indicates whether the premium is positive or negative. To add a history of past electricity consumption and price to the records, the most relevant historic electricity data for both auction types were calculated for each time slot (see figure “Aggregated Columns Example.png”). The most relevant data for a time slot includes the auction data at the last date, which is available at the prediction time and which also contains the slot. Furthermore, it comprises the slope value of price and consumption for the week before the target time slot. This additional historic approach causes the exclusion of the first eight days for the predictive dataset since the former electricity data for these entries is not included in the dataset.

**(d) Data mining solution: Methods applied (with sufficient detail and screenshots; use Appendix if needed) and appropriate performance evaluation (proper choice of measures, benchmarking).**

**Descriptive**

After the basic understanding of the data and the additional preparation the first methods can be applied in order to accomplish the first goal of this analysis: Finding hidden pattern in the data. The methods used for this purpose should be applicable to time series data and fit the data well. Furthermore, the most important feature is interpretability of the model since the second goal is to derive potential trading suggestions and these can only be deduced if the model is transparent in which and how the respective attributes are causing a potential hidden pattern.

The first two features are provided by statistical ARMA based models. The presumption for ARMA models is that the time series is stationary. That means that mean and standard deviation ought to be stable over time. ARMA models are divided into an “Auto Regressive” (AR) and a “Moving Average” (MA) part. In order to describe a particular timeslot t the auto regressive part represents the influence of the last p timeslots and the moving average part contributes of the last q timeslots. Therefore the standard parameter for ARMA are p,q which are extended with several others depending on different modifications of ARMA such as ARIMA (Auto Regressive Integrated Moving Average), where I is the parameter that describes the number of differencing in order to make the time series stationary. Other modifications of ARMA models are SARIMA, which considers seasonality or ARIMAX, which considers exogenous variables. The biggest drawback of ARMA type models is the difficulty of interpretation, which is one of our criteria. Thus, ARMA models are not used in this part, but in the predictive part of our further analysis. Other opportunities especially for pattern recognition in time series are seasonality and trend. Both meets all the required criteria. A visual analysis shows that trend can be obviously excluded, but seasonality more difficult to assess. [PLOT PLAIN TIMESERIES] Hence a ACF-Plot shows which previous timeslots are most correlated with a specific timeslot. [PLOT ACF] The plot states out that there is a high autocorrelation between a timeslot t and the timeslot t-96, which is the timeslot 24 hours before t. Another peak can be observed at 672, which describes a high autocorrelation between t and the timeslot one week earlier. (TODO: ADDING AN EVALUATION FOR A SIMPLE MODEL THAT ONLY CONSIDERS SEASONALITY BASED ON ITS AUTOCORRELATION- t = 0.6\*t-96 + 0.4\*t-672  evaluation evt auch über decomposition/residual)

(TODO: weitere modelle die speziell auf time series daten spezialisiert? – ordinale ts analyse?)

Apart from models which are designed especially for time series data there are other models that can be taken into account such as a decision tree model. The major advantage of a tree model is its interpretability. In order to find the best model parameter with the least number of leaves that fits the data well, a grid search is implemented for the model. In addition, a time series capable cross validation is implemented to verify the accuracy, which is used to measure how well the model fits the data. The resulting tree shows that the model uses only three variables to identify whether the “price premium” is positive or negative: “dailySunnyHoursAvg”, “numberRainyDays” and “participants”. [tree model plot]

The accuracy is 78.89% (and the confusion matrix is shown in the following.)

[[38012 6582]

[ 8068 16746]]

[TODO: trading suggestions einfügen wenn Tree Ergebnis geklärt -> normalisiert anderes Ergebnis ohne]

**Predictive**

The predictive analysis is in contrast to the descriptive part not focused on interpretability and goodness of fit. The only dimension of interest is the predictive power, which is represented through accuracy throughout this project. In order to simplify the interpretation of the performance through the accuracy a baseline is calculated first. Therefore, a simple “ZeroR” classifier is introduced, which predicts only the class which has the most observations in the dataset. This technique simulates random predictions. With regard to our dataset the baseline is: 64.25%=44594/69408. Above mentioned the tree classifier can also be used for out-of-sample predictions and with reference to its accuracy 78.89% it can be stated out that it is around 14% better than the defined baseline.

For the predictive classification the following models were trained and tested: Stochastic Gradient Descent (SGD) Classification, Binary Support Vector Classification (SVC), Gaussian Naive Bayes, Soft Voting Ensemble, Hard Voting Ensemble, Logistic Regression, Recurrent Neural Network (RNN), Decision Tree, Autoregressive-Moving Average (ARIMA) and Random Forest.

For each model a time series compatible grid and random search was implemented. After the search finished, the hyperparameter combination of the best performing model, measured by accuracy rate, was picked and evaluated on the full dataset using daily split increments. Exemplary screenshots for the Logistic Regression model can be found in the appendix. The implementation of a time series compatible search can be seen in figure xy. An analysis of the hyperparameter search is shown in figure xy and figure xy visualizes the implementation of the final model run.

For a performance comparison of the applied models see figure “modelComparision.png”. Plot in text (high relevance)?

The best performing model was the Recurrent Neural Network (RNN) with an accuracy of 81.4% and ROC-score of 0.75. The confusion matrix is displayed through figure xy. Analysis of matrix. The plot of the ROC curve is depicted by figure xy. Here the neural net receives an input similar to a normal neural net. In addition, as with a neural network, various complex calculations are carried out in the network and we finally get a prediction for a certain input. First the weights at the knots must of course be trained. The peculiarity about a recurrent neural network is that the previously calculated values still affect the future values. This is especially useful in this case, because values for a certain point in time are searched, which are affected by the values of past time slots. For our RNN we first and foremost have the data we need for a prediction into a supervised learning problem. It's about copying the target feature and if we want to shift from time t to t+1 predicting by one to the top, just like we do. By example, if we want to predict from time t to t+2, we have to copy the time variable and shift it upwards by two values.

**(e) Conclusions (advantages and limitations) and operational recommendations**

1. A small exception of this is the data that is affected by the switch to daylight saving time and the switch back to standard time. If two values ​​exist for a time slot, only one value was included in the dataset. If there was a missing hour, additional values ​​were generated for the affected slots. [↑](#footnote-ref-1)