Image source: https://www.borealisgroup.com/news/borealis-signs-wind-farm-power-purchase-agreement-with-axpo-continuing-to-significantly-boost-share-of-renewable-energy-used-in-its-belgian-operations

Industry project 1: Machine learning for renewable energies

Bias identification and prediction models

Joël Tauss
Version 2.1
07/01/2023

Bern University of Applied Sciences

Technology and Computer Science Department
Business Engineering

Contents

1. Management Summary

The forecasting of a power output is essential for wind park operators. Therefore, these need to be as accurate as possible to minimize potential financial losses. The goal of this project is to identify systematic errors and their origins in the power forecasts and eliminating these errors to create a more precises prediction. Additionally, the potential uses of graph neural networks within this field is of interest.

In order to achieve the before mentioned goal, the “standard” approach for data science is used. Some of the steps, like the data gathering and model deployment, are excluded. The data is given by the framework conditions and the model deployment is out of scope. The applied machine learning models include the following: Random forest regressor (supervised), convolutional graph node classifier (supervised), graph node outlier detector (unsupervised).

The following systematic biases, influencing the power forecast errors could be identified: time of the day, seasonality, temperature, position of wind turbines, wind speed. With these identified biases, the random forest regressor was able to create a more accurate power forecast for the given wind park. Reducing the root mean squared error of the predicted power output in kilowatts from around 700 to 200. The graph neural networks are not directly applicable to create a more accurate forecast, but other uses could be identified. Such as: Predicting the expected wind wake in a park based on turbine layout; identifying atypical wind turbines; predicting the overall deviation of the power forecast of each turbine, based on its relative location.

2. Introduction

2.1 Context and problem

This project is concerned with the topic of bias detection and predictions in the subject of renewable energies. Renewable energy sources are becoming more and more relevant. The on- and especially off-shore wind parks are predicted to grow in the coming years all across the globe. With such growth it is vital to be able to make accurate predictions of the energy output of these sources in order to plan ahead on the input into the power grid. [1] [2]

The manufacturer of the wind turbines provides a data table (power curve table) with wind speeds, air density and the resulting power output of a turbine. Such tables are based on numerous different calculations and tests. These data tables should be as precises as possible [3]. The operators or owners of wind parks use this information in combination with a fitting weather forecast to predict the power output of the wind park. Therefore, predicting how much electricity the farm is able to insert into the power grid. These forecasts also get used to set prices on the energy market. If this forecast is off or incorrect, then this can lead to major financial losses for the wind park operator. If the agreed upon amount of power is not met and delivered, the electricity must be bought from somewhere else.

Because the power curve tables of the turbines are derived from data measured under very specific conditions, the real-world output will deviate from the theoretical predictions. As previously stated, this can lead to financial losses for the wind park operators. The goal therefore is to make a more accurate prediction, than the power curve allows. To achieve this, the following research questions will have to be answered:

- What systematic biases are in the deviation of the predicted theoretical and the measured power output of the wind turbines?
- If systematic biases can be identified, how can they be corrected?
- How can graph neural networks be applied for such a use case or in the concerning field in general?

2.2 The wind park

The wind park relevant for this project is an onshore operation with a total of 10 turbines. Those are split up into a front and back row. The setup of the wind park with the turbine IDs can be found below (this is an estimation, the exact setup is unknown). The wind park in question, its location as well as the data, is anonymised.

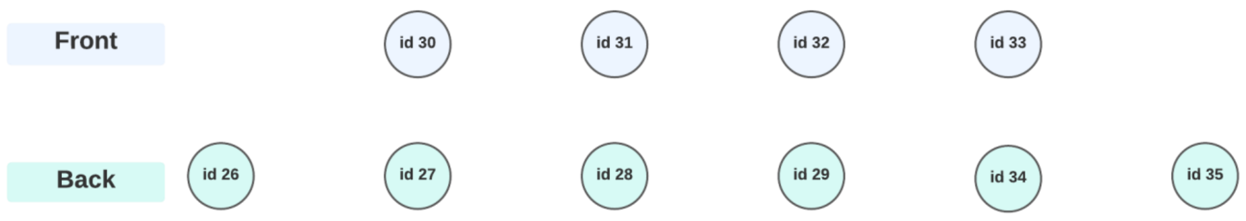


Figure 1 Wind park layout (assumption)

2.3 Relevant data

For this project, five datasets in form of csv-files are used. The data gathering is out of scope for this project. The files were provided by the supervisors.

Scada data:	Scada (supervisory control and data acquisition) is a type of software application for controlling industrial machinery and processes [4]. The gathered data stems from sensors mounted on the wind turbines. It consists of around 1mio. datapoints. The time resolution is 10 minutes, ranging from 2020-01-01 until 2021-12-31. Included features are: wind turbine id, UTC time stamp (central European time), wind speed, generated power output, wind direction, nacelle direction, blade angel, rotor speed, environmental temperature.
Power curve table:	A power curve table is given by the turbines manufacturer. It consists of three columns: Wind speeds, air density and the resulting power output of the turbine.
Weather forecast:	Three separate files are provided. One for temperature, wind speed, and wind direction forecasts each. The time resolution is 6 hours, ranging from 2020-01-01 to 2022-04-01. For each forecasting datapoint, the lead times 1 to 72 hours, with a step size of 15 minutes are given. This results in about 3’200 datapoints for each of the forecast data frames.

3. State of research

3.1 Power prediction of wind turbines

Because of the increase in wind parks being operated, the need to make predictions of the power outputs is not a new concept. There are multiple different approaches to tackle and solve this problem.

One approach is to model the output based on a numerical weather prediction model. Therefore, mainly focusing on the weather forecast itself, but also trying to correct systematic errors and account for wind farm specific effects. Such an approach seems to work best for higher power outputs. In the paper it is noted that the inclusion of measured power data into the prediction, poses a possibility to improve the model's performance. Thus, the measured power curve deviates drastically from the manufacture's data. [5]

To improve the real-life power curve, models like polynomial regression provide an uncomplicated way to do so. But such models are limited by their global nature and are sensitive to outliers and extreme values. To get a good fit to the measured data, a higher degree of polynomial is required. These may not perform as well for values outside the fitted range. A proposed solution would be to implement piecewise polynomial regression models. [6]

A more data driven approach can be used in the form of deep neural networks. An example is the use of deep neural networks with transfer learning for short term prediction of multiple turbines. Goal being to characterize heterogenous features among wind turbine systems. [7]

Convolutional neural networks were also proposed for probabilistic wind power forecasting. Finding that the distribution of wind power data can be statistically formulated to better account for the uncertainties and inaccuracies in the forecast. [8]

3.2 Deep neural networks

Deep neural networks are a type of machine learning model which try to imitate the structure of a human brain. It consists of an input, at least one hidden, and an output layer. On each layer are neurons. The number of neurons on the input layer are equal to the number of input features of the model. The number of neurons on the hidden layers can be chosen arbitrarily. The number of output nodes is decided by the type of model. If it is a regression model, there will always be one output node. If it is a classifier, the number of output nodes corresponds to the number of classes. [9] [10]

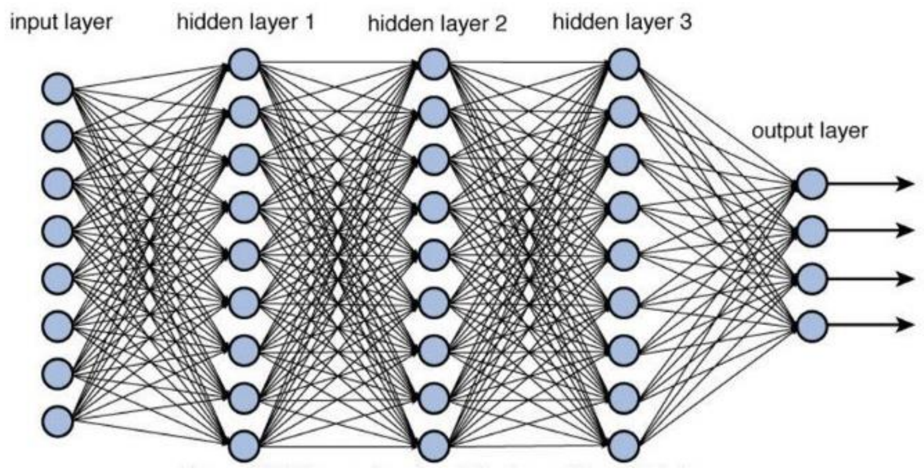


Figure 2 Deep neural networks, source: https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964

3.3 Graph neural networks (GNNs)

Graph neural networks are a type of machine learning models, which can take graph data as input features. A graph typically consists of the below listed features. On each of these elements additional information can be stored as attributes. These further describe the concerning part of the graph. Some examples of graph which can be represented as graphs are images, texts, molecular structures, and social networks. [11]

Graph data elements [11]:
- Vertex attributes / node
- Edge attributes and directions / link or edge
- Global attributes / graph data

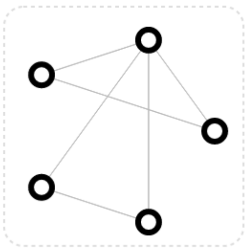


Figure 5 Vertex / Node, source : https://distill.pub/2021/gnn-intro/

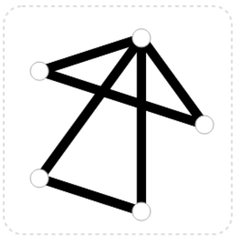


Figure 4 Edge / Link, source : https://distill.pub/2021/gnn-intro/

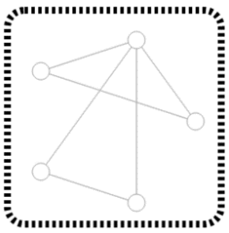


Figure 3 Global / Graph, source : https://distill.pub/2021/gnn-intro/

Graph neural networks are used to solve classification problems or anomaly detection. Those predictions or classifications can be done on three different levels: [11]

Graph level:	Is used to make predictions for the whole graph such as graph level features. Can only be done when data on multiple different graphs is available. Example: Classify if a molecule possesses certain attributes, based on the atom structure and bonds between the atoms. [11]
Node level:	Is used to make predictions for specific nodes and node features or create anomaly detections. Example: Classify an atom’s features in a molecule, based on its bonds and bond types. [11]
Edge level:	Is used to make predictions for specific edges and edge features. Example: Classify the bonds and types of bonds of an atom within a molecule based on the atoms and its features. [11]

Graph neural networks already get applied in various different fields. Some of those applications include: [12]
- Natural language processing or text classification
- Neural machine translations
- Relation extraction
- Image classification
- Object detection
- Physics
- Molecular fingerprints
- Protein interface predictions
- Combinatorial optimization
- Graph generation

When it comes to the application of graph neural networks in the field of energy and renewable energies, the number of papers and research attempts are not very numerous. The most covered use cases are predictions of the stability of power grids, optimal power flow calculations, and wind speed predictions. The main problem which arises in this area is, that graph neural networks are not optimized to work on one dimensional time series data. [13]

4. Methods

4.1 General Approach - Data science

The overarching structure of the accompanying jupyter notebook as well as the results chapter is given by the “standard” approach for data science projects. Although some of the steps and decisions are prerequisites by the framework conditions of this project and are marked as such. [14]

1. Gain / create business understanding:
   Explored in chapters 2 and 3

2. Analytical understanding:
   Split into 4 different approaches: Descriptive approach, diagnostic approach, predictive approach, and prescriptive approach. Relevant is the diagnostic approach for the bias identification and the predictive approach for the bias correction and forecasting.

3. Data collection and data requirements (out of scope):
   Gathering the needed data to handle the use case and requirements. See chapter 2.3.

4. Data understanding:
   Creating an understanding for the given data, how it is structured and what can be achieved with the given data. Often done by the means of plotting or looking at statistical values such as median, means, standard deviations, etc.

5. Data preparation:
   Preparing the data for the needed goal. This includes data cleaning, if necessary, data concatenation or joining, and feature engineering. See chapter 4.2

6. Modelling:
   Selecting suitable machine learning models for predictions. See chapter 4.4 and developing said model. As it is usual in data science, the data will be split into a training, validation, and testing data set [15].

7. Evaluation:
   After creating the predictions, the results have to be assessed and evaluated. Deciding on the validity and precision of the results and if changes to model or model parameters are necessary. Simplifying the model as good as possible.

8. Deployment (out of scope):
   After a successful evaluation, the model will be ready for deployment in a productive environment.

9. Feedback (out of scope):
   To further increase the model’s performance in a production environment, feedback and is implemented.

4.2 Data preparation

Within the available forecast data, there are lead times ranging from 1 to 72 hours in steps of 0.25 hours. Because the modelling and analysing for all lead times would be exceeding the time scope of the project, only three lead times will be regarded: 3, 6, and 12 hours.

As stated in chapter 2.3, the data is split up into multiple different data frames with varying time resolutions. If the weather forecast and scada data would be joined, only a fraction of the scada datapoints would have a corresponding weather forecast. This will result in a massive data loss. To circumvent this, the weather forecast data is interpolated before joining, to a 10 minute-resolution. The interpolation is done using a linear approach to simplify the matter [16]. Other options may be viable, but are not to be further explored in this project. The data can then be concatenated to the scada data without any data loss. Formula for the linear interpolation [16]:

$$y = y_1 + \frac{(x - x_1)(y_2 - y_1)}{x_2 - x_1}$$

x_1, y_1	: first coordinates
x_2, y_2	: second coordinates
x	: point to perform the interpolation
y	: interpolated value

Datetime and time values are on paper just continuously increasing features. But this does not exactly describe the more cyclic nature of the passing of days, months or years. Therefore, these features have to be transformed by the means of cyclic functions. This can aid in finding certain patterns correlating to said time frames. For this project the cosine and sine functions are applied. Both are needed, because both of these on their own, repeat all values in a cycle twice. This will result in the same values of the feature corresponding to different date times. If both are provided, there will be a unique combination of values of the cosine and sine function for each date time, regardless of the scaling. [17]
Additionally, the positional data given (front or back row) are added to the data frame as a categorical value, encoded as 0 or 1.

In order to create a power forecast, the power curve table is applied to the scada data frame. The air density is not given in the data frame and must be derived and approximated from the temperature. This method is not the optimal route to take. The formula was derived from the given source, by linearizing the course of the thermal expansion coefficient. The approximation of the function is precises enough for the relevant temperature range and is rounded to 0.05 to match the power curve table.

$$y = a * x + b$$
$$y = -0.00392 * x + 1.2838$$

y : density
x : temperature

Datapoint for the derivation of the equation, given by the minimum and maximum measured temperature (temperature in Celsius / resulting air density) [18]:

Min: (15 / 1.2250)
Max: (40 / 1.1270)

Normally, wind speed forecasts are done for speeds measured at 10 meters above ground level [19]. Because the average turbine head is mounted at about 100 meters, these values would have to be transformed with the following function [20] [21]. But the given forecasts for the windspeeds were transformed beforehand. Henceforth, this transformation is not executed in the accompanying notebook.

$$v_2 = v_1 \frac{\ln\left(\frac{h_2}{z_0}\right)}{\ln\left(\frac{h_1}{z_0}\right)}$$

v_1 : measured wind speed at h1
v_2 : transformed wind speed at h2
h_1: measurement's v1 height (10 meters)
h_2 : measurement's v2 height (100 meters)
z_0 : roughness class (class 1, 0.03 meters)

4.3 Principal component analysis (PCA)

The value spread of each feature in the data is an important factor. If a feature has minimal variation, it is more likely to have a lower impact on the machine learning model. Meaning, not much information can be gained with such a feature. When applying neural networks, this can be crucial for the feature selection and model simplification. The principal component analysis (PCA) is a tool to analyse such value spread of features and consist of the following steps: [22]

1. Standardization of the values, in order to gain equal contributions from each feature. This is crucial. If no standardization would be applied, the results will be misleading because features with higher values overall would contribute more to the PCA than others. [22]

$$z = \frac{value - mean}{standard\ deviation}$$

2. Calculating the covariance matrix. The goal is to spot any relations between the various features in the data set. If a high correlation between features exists, there is more redundant data within the data set. The covariance matrix is p x p matrix, where p is the number of dimensions or number of features. [22]

3. Computing the eigenvectors and eigenvalues of each covariances will lead to identify the principal components within the given data. The new axis tries to encompass as much data in as few principal components as possible. Meaning, if the first few principal components are much higher than the rest, the lower the differences between the evaluated data. [22]

4.4 Bias detection

In order to detect biases in the forecast, the deviation of the forecasted value is needed. This will be handled by the mean deviation. When compared to a root mean squared error, this method will take into account negative values. The bias detection itself will be done by the means of visual analysis of the plots and graphs.

4.5 Model selection

The base line model to correct biases is a random forest regressor, implemented with the scikit learn (sklearn) library for python 3.

The graph neural networks are based on different libraries, depending on the given level of the classification:

Graph level:	None
Node level:	Tensorflow, PyTorch Geometric, PyGOD
Edge level:	PyTorch Geometric

In order to fit the data into graph from, an additional data frame is created, based on the assumed position of each turbine. Each node represents a turbine. Each edge the potential influence on other turbines. The edges do not include any features, only describing the positional relationship. As stated in the chapter 2.2, the exact layout is unknown and has to be assumed.

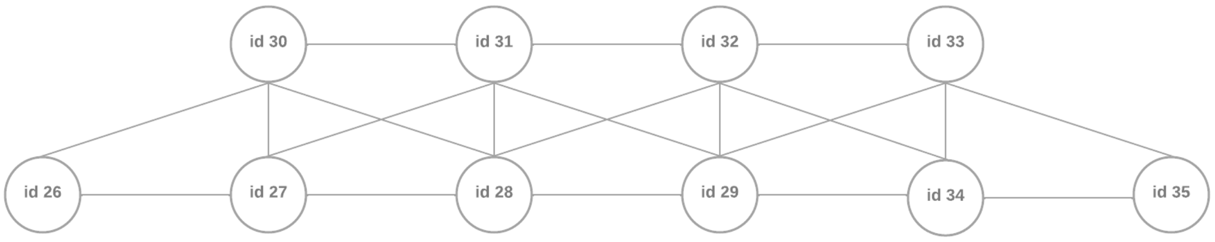
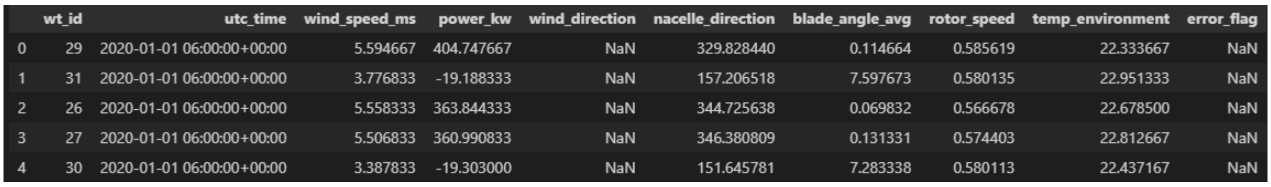


Figure 6 Layout assumption graph data

5. Results

5.1 Data understanding (Notebook chapter 1, 2)

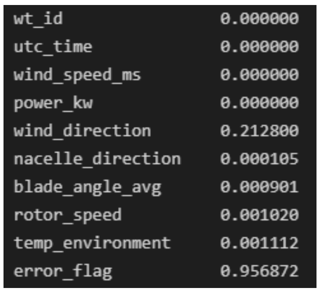
The first thing standing out when taking a look at the scada data is, that the column wind_direction has only NaN values in the first few rows.



	wt_id	utc_time	wind_speed_ms	power_kw	wind_direction	nacelle_direction	blade_angle_avg	rotor_speed	temp_environment	error_flag
0	29	2020-01-01 06:00:00+00:00	5.594667	404.747667	NaN	329.828440	0.114664	0.585619	22.333667	NaN
1	31	2020-01-01 06:00:00+00:00	3.776833	-19.188333	NaN	157.206518	7.597673	0.580135	22.951333	NaN
2	26	2020-01-01 06:00:00+00:00	5.558333	363.844333	NaN	344.725638	0.069832	0.566678	22.678500	NaN
3	27	2020-01-01 06:00:00+00:00	5.506833	360.990833	NaN	346.380809	0.131331	0.574403	22.812667	NaN
4	30	2020-01-01 06:00:00+00:00	3.387833	-19.303000	NaN	151.645781	7.283338	0.580113	22.437167	NaN

Figure 7 scada data

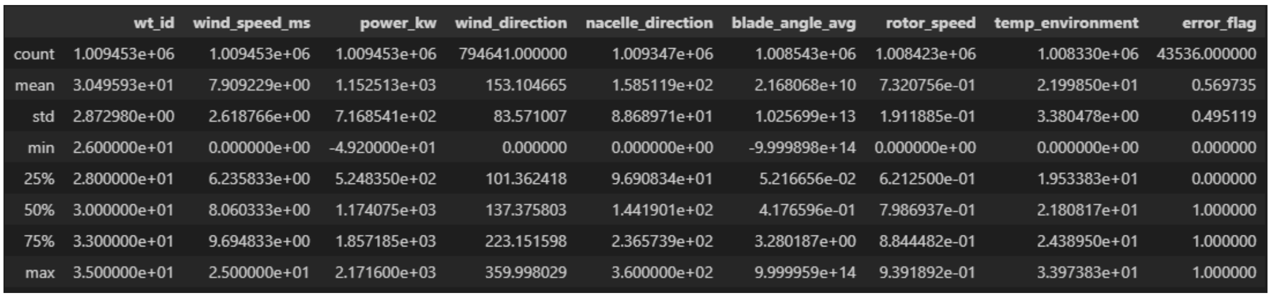
Taking a closer look at all the column's missing data ratio in percent, it is apparent, that the wind direction has the most missing values.



```
wt_id                0.000000
utc_time             0.000000
wind_speed_ms        0.000000
power_kw             0.000000
wind_direction       0.212800
nacelle_direction    0.000105
blade_angle_avg      0.000901
rotor_speed          0.001020
temp_environment     0.001112
error_flag           0.956872
```

Figure 8 Missing scada data

The key figures on the data frame look as follow:



	wt_id	wind_speed_ms	power_kw	wind_direction	nacelle_direction	blade_angle_avg	rotor_speed	temp_environment	error_flag
count	1.009453e+06	1.009453e+06	1.009453e+06	794641.000000	1.009347e+06	1.008543e+06	1.008423e+06	1.008330e+06	43536.000000
mean	3.049593e+01	7.909229e+00	1.152513e+03	153.104665	1.585119e+02	2.168068e+10	7.320756e-01	2.199850e+01	0.569735
std	2.872980e+00	2.618766e+00	7.168541e+02	83.571007	8.868971e+01	1.025699e+13	1.911885e-01	3.380478e+00	0.495119
min	2.600000e+01	0.000000e+00	-4.920000e+01	0.000000	0.000000e+00	-9.999898e+14	0.000000e+00	0.000000e+00	0.000000
25%	2.800000e+01	6.235833e+00	5.248350e+02	101.362418	9.690834e+01	5.216656e-02	6.212500e-01	1.953383e+01	0.000000
50%	3.000000e+01	8.060333e+00	1.174075e+03	137.375803	1.441901e+02	4.176596e-01	7.986937e-01	2.180817e+01	1.000000
75%	3.300000e+01	9.694833e+00	1.857185e+03	223.151598	2.365739e+02	3.280187e+00	8.844482e-01	2.438950e+01	1.000000
max	3.500000e+01	2.500000e+01	2.171600e+03	359.998029	3.600000e+02	9.999959e+14	9.391892e-01	3.397383e+01	1.000000

Figure 9 scada data overview

The shape of the data frame is 1'009'453 rows, and 10 columns. The date range of the measurements start 2020-01-01and end at 2021-12-31. Meaning the data spans two complete calendar years.

A first look at the correlation matrix reveals some insights. There are some obvious connections between the power output and wind speed, nacelle and wind direction, as well as rotor and wind speed. But also, a negative correlation between the temperature and wind speed and therefore power output as well. A definitive reasoning for this could not been found. But the following assumption can be made: The wind park itself is on shore. But if it is located near a body of water, a lower air temperature is likely to result in a lower difference in air and water temperature. Therefore, lowering the wind speed.

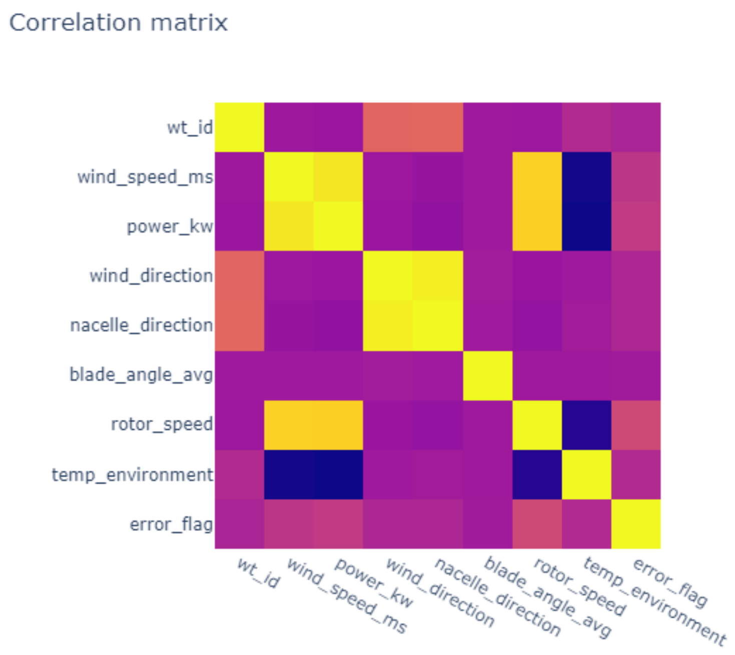


Figure 10 scada correlation matrix

A deviation can already be seen in the different power outputs per turbine and the spread of the data. The medians deviate quite heavily from one another. The outliers all seem to start at 0 and cut off at around 2’000 kilowatts.

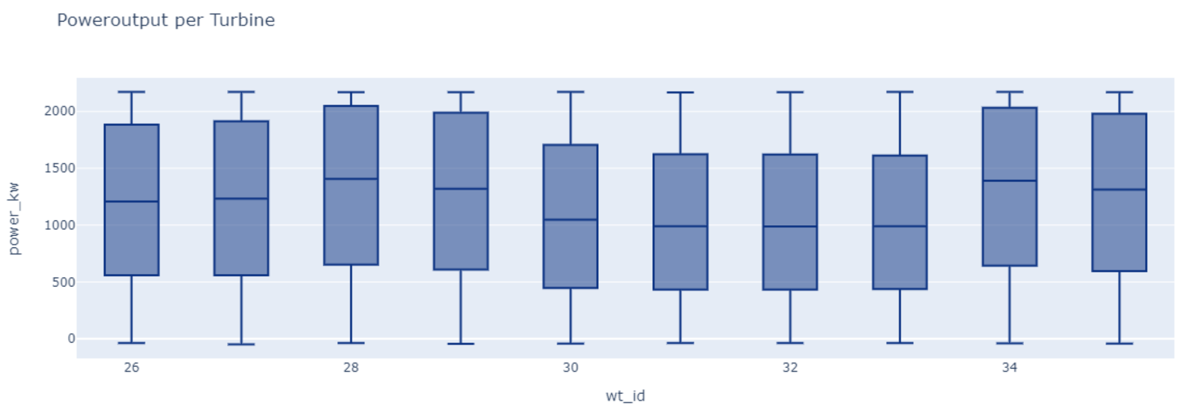


Figure 11 Power output by wind turbine

When looking at the power curve table, the cap at around 2'000 kilowatts in Figure 11 can be explained. The turbines seem to be designed to cap off at said power output. This is probably due to safety reasons and preventing the rotors from spinning out of control. The same reasoning can be applied to explain the drop off at wind speeds of around 21 m/s.

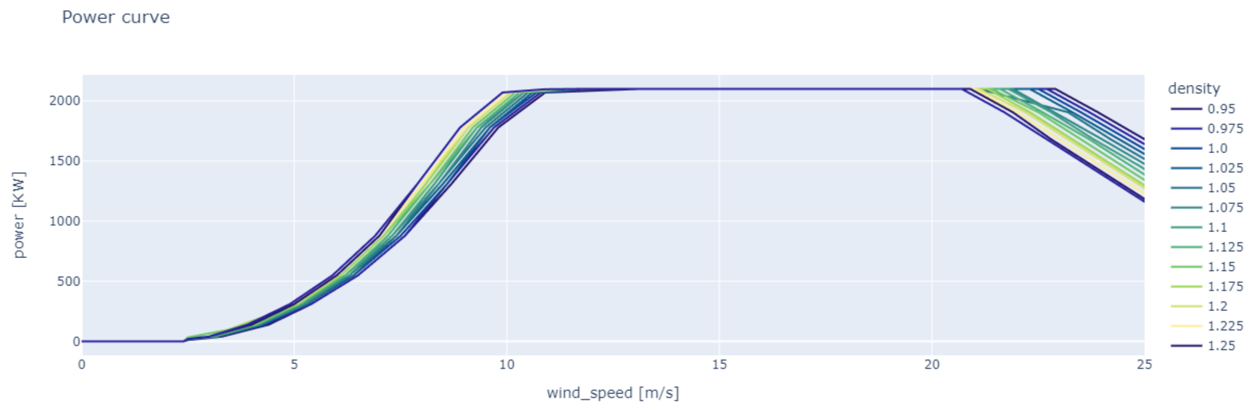
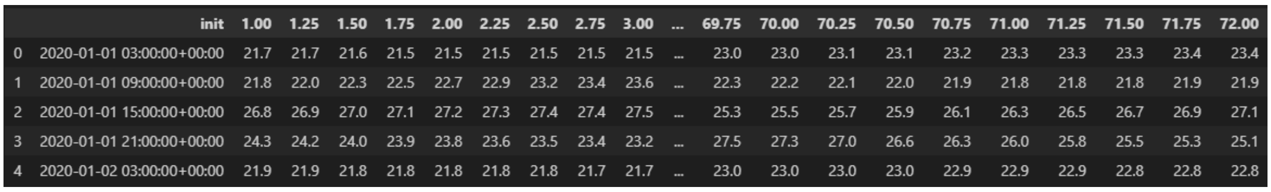


Figure 12 Power curve

The weather forecast data is structured and split up into 3 separate data frames, one for each metric (wind speed, wind direction, temperature). Each of those consist of the same axis. The rows are labelled with the time of the forecast and the columns represent the lead time. The dimensions are 3'281 rows by 286 columns. The measurements range from 2020-01-01 03:00:00 until 2022-04-01 15:00:00. This means, when the scada and forecast data are joined, some of the forecasted values will not be needed.



	init	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00	...	69.75	70.00	70.25	70.50	70.75	71.00	71.25	71.50	71.75	72.00
0	2020-01-01 03:00:00+00:00	21.7	21.7	21.6	21.5	21.5	21.5	21.5	21.5	21.5	...	23.0	23.0	23.1	23.1	23.2	23.3	23.3	23.3	23.4	23.4
1	2020-01-01 09:00:00+00:00	21.8	22.0	22.3	22.5	22.7	22.9	23.2	23.4	23.6	...	22.3	22.2	22.1	22.0	21.9	21.8	21.8	21.8	21.9	21.9
2	2020-01-01 15:00:00+00:00	26.8	26.9	27.0	27.1	27.2	27.3	27.4	27.4	27.5	...	25.3	25.5	25.7	25.9	26.1	26.3	26.5	26.7	26.9	27.1
3	2020-01-01 21:00:00+00:00	24.3	24.2	24.0	23.9	23.8	23.6	23.5	23.4	23.2	...	27.5	27.3	27.0	26.6	26.3	26.0	25.8	25.5	25.3	25.1
4	2020-01-02 03:00:00+00:00	21.9	21.9	21.8	21.8	21.8	21.8	21.8	21.7	21.7	...	23.0	23.0	23.0	23.0	22.9	22.9	22.9	22.8	22.8	22.8

Figure 13 Weather forecast data

When it comes to errors in the forecast data, it seems that there is no missing data. Looking at plots, on the other hand, reveals that there are some errors in the measurements. Because the data point will be interpolated in a later step, the faulty values are simply dropped.

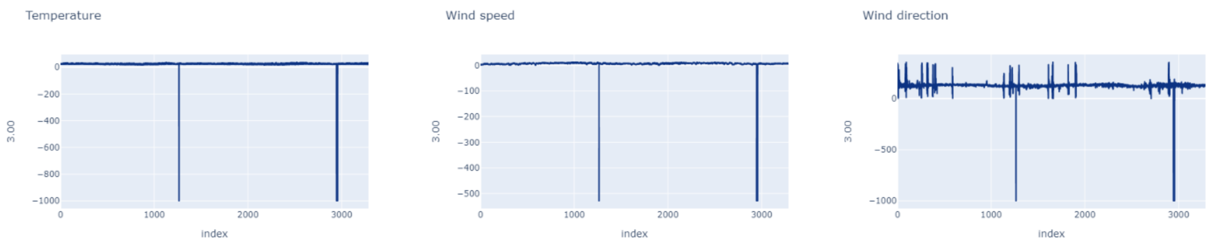


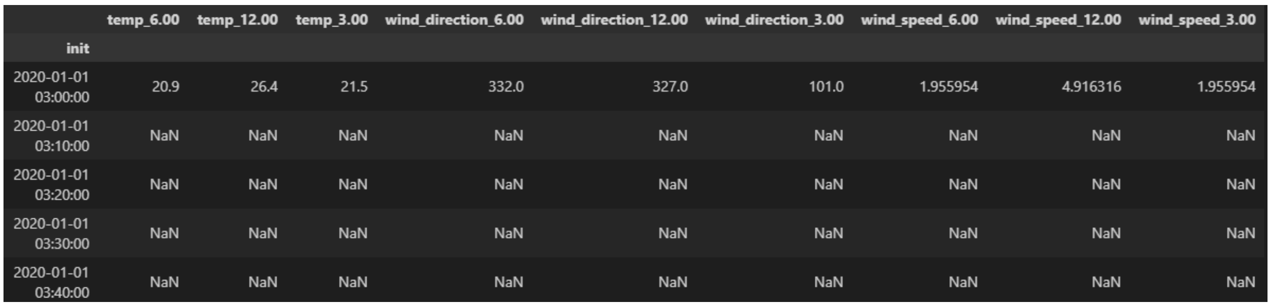
Figure 14 Weather forecast data errors

5.2 Data preparation (Notebook chapter 2, 3)

Because the scada data is almost complete and only a few values are missing, the data frame will be left as it is. As mentioned in chapter 5.1, the faulty values in the weather forecast will be dropped and interpolated.

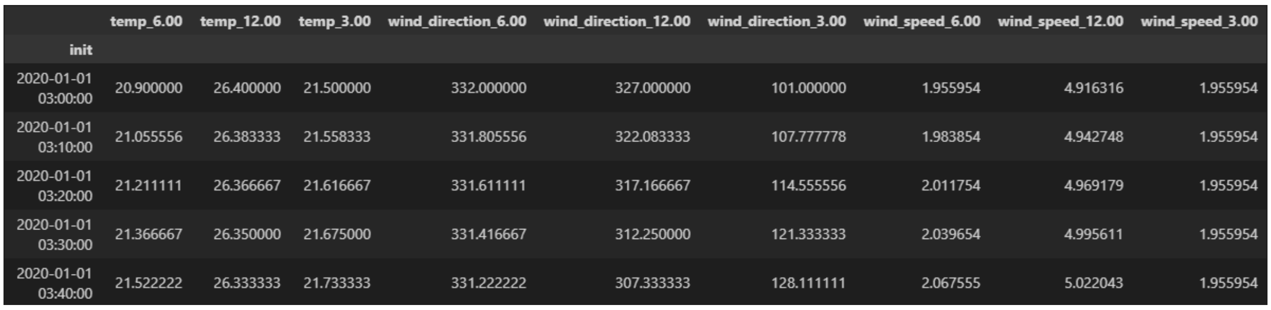
For the interpolation, an empty data frame is created with the according columns. The index consists of date time objects with the same 10-minute resolution as the scada data. Then the given values are put in (Figure 15 Raw weather forecast data). Because the single columns are evenly spaced out, the interpolate function of pandas can be used with its default parameters (Figure 16 Interpolated weather forecast data).

In order to join the weather forecast data correctly on to the scada data, the interpolated data frame is split into 3 data frames. One for each lead time.



	temp_6.00	temp_12.00	temp_3.00	wind_direction_6.00	wind_direction_12.00	wind_direction_3.00	wind_speed_6.00	wind_speed_12.00	wind_speed_3.00
init									
2020-01-01 03:00:00	20.9	26.4	21.5	332.0	327.0	101.0	1.955954	4.916316	1.955954
2020-01-01 03:10:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2020-01-01 03:20:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2020-01-01 03:30:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2020-01-01 03:40:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Figure 15 Raw weather forecast data



	temp_6.00	temp_12.00	temp_3.00	wind_direction_6.00	wind_direction_12.00	wind_direction_3.00	wind_speed_6.00	wind_speed_12.00	wind_speed_3.00
init									
2020-01-01 03:00:00	20.900000	26.400000	21.500000	332.000000	327.000000	101.000000	1.955954	4.916316	1.955954
2020-01-01 03:10:00	21.055556	26.383333	21.558333	331.805556	322.083333	107.777778	1.983854	4.942748	1.955954
2020-01-01 03:20:00	21.211111	26.366667	21.616667	331.611111	317.166667	114.555556	2.011754	4.969179	1.955954
2020-01-01 03:30:00	21.366667	26.350000	21.675000	331.416667	312.250000	121.333333	2.039654	4.995611	1.955954
2020-01-01 03:40:00	21.522222	26.333333	21.733333	331.222222	307.333333	128.111111	2.067555	5.022043	1.955954

Figure 16 Interpolated weather forecast data

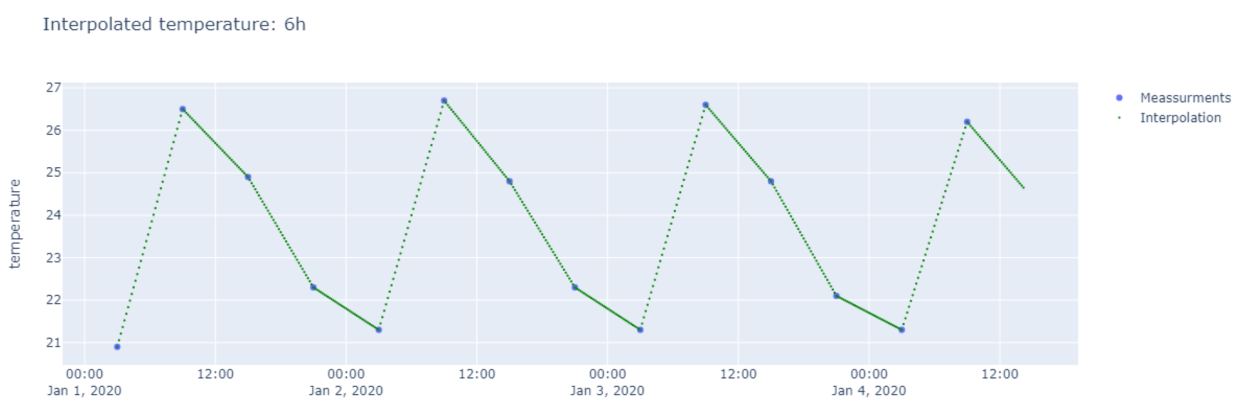


Figure 17 Interpolated temperature 6h lead time

After the interpolation the forecast data is joined onto the scada data frame via an inner join. Meaning, all scada data points without any forecasted values are dropped. To apply the power curve table, the predicted air density must be calculated as described in chapter 4.2 for each of the lead times. Afterwards the power curve table can be applied according to the predicted air density and wind speeds. The forecasted values are joined in a way, that the initial time stamp of the scada data includes the forecasted value from t – 3, 6 and 12 hours. The mean deviation is then calculated by getting the delta of the measured power output and the forecasted power output. These steps result in a new master data frame with the following columns / features:

wt_id, utc_time, wind_speed_ms, power_kw, wind_direction, nacelle_direction, blade_angle_avg, rotor_speed, temp_environment, error_flag, init_6, temp_6.00, wind_speed_6.00, wind_direction_6.00, init_12, temp_12.00, wind_speed_12.00, wind_direction_12.00, init_3, temp_3.00, wind_speed_3.00, wind_direction_3.00, density_6.00, density_12.00, density_3.00, power_6.00, power_12.00, power_3.00, density, wind_speed_theoretical, power_theoretical, density_theoretical, deviation_theoretical, deviation_3.00, deviation_6.00, deviation_12.00

The forecasted power outputs of all three lead times compared to the measured power output, looks as follows:

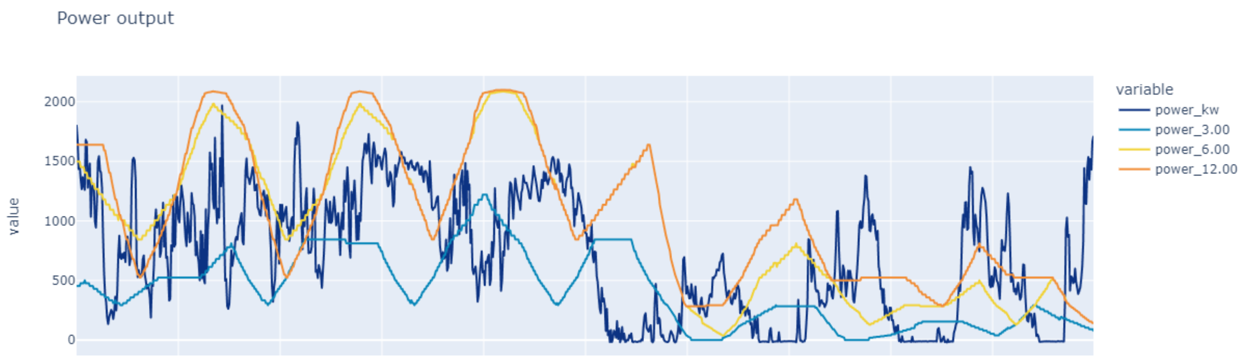


Figure 18 Predicted power output

After generating an all-encompassing data frame, additional features are added. As stated in chapter 4.2, the date time data must be transformed accordingly. The sine and cosine function are applied for the time intervals of 24 hours and one year. Furthermore, the month of the year is added as a numerical value ranging from 1 to 12. Positional data, front or back, is appended for each datapoint, according to the wind turbine id and the given information.

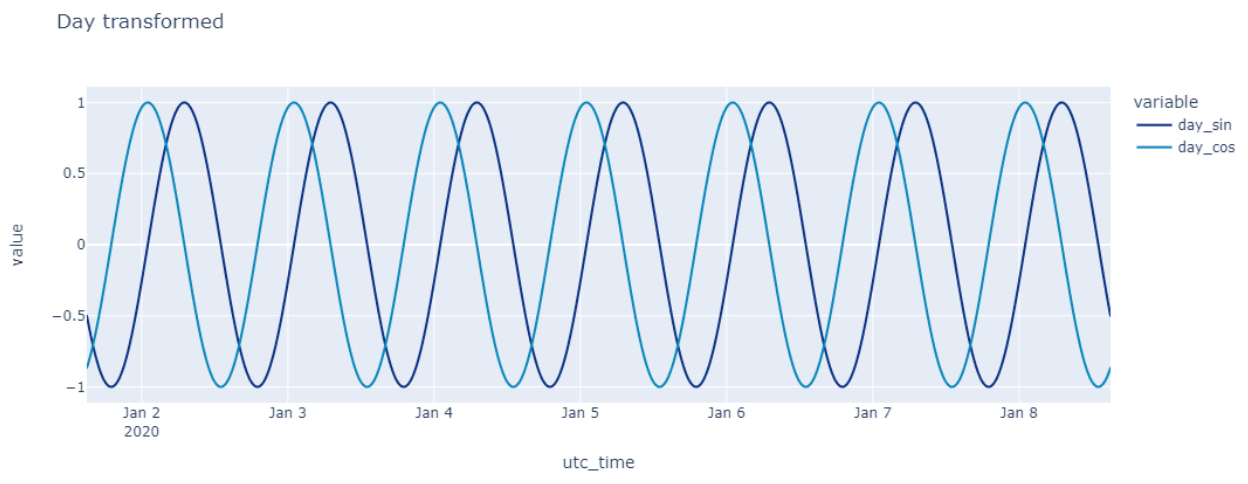


Figure 19 Day sine and cosine

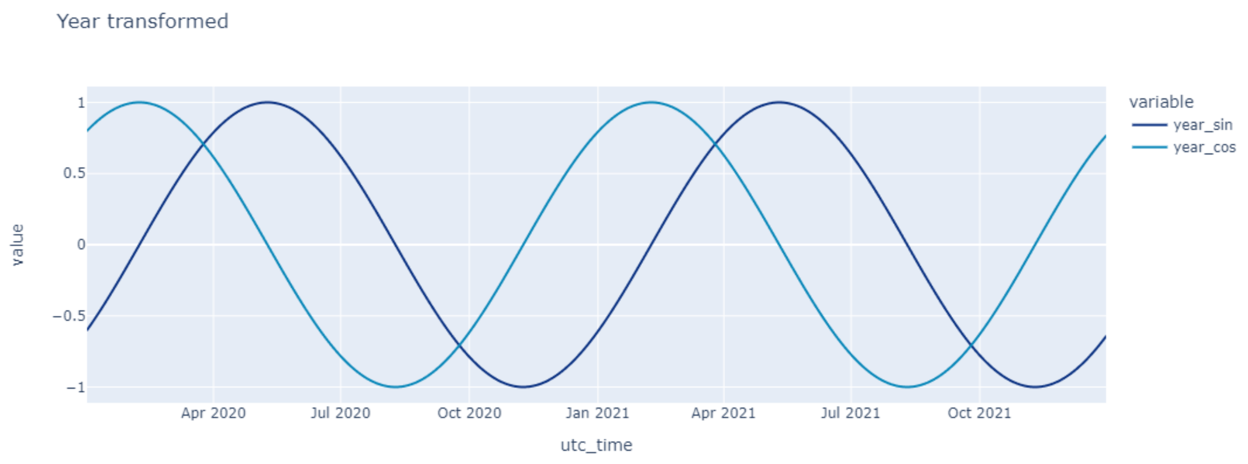


Figure 20 Year sine and cosine

The principal component analysis reveals that there are about 4 features that contain most of the information. The remaining features barely hold any additional information. A reason could be, that some features in the data frame heavily depend on each other, as already shown in the correlation matrix in chapter 5.1.

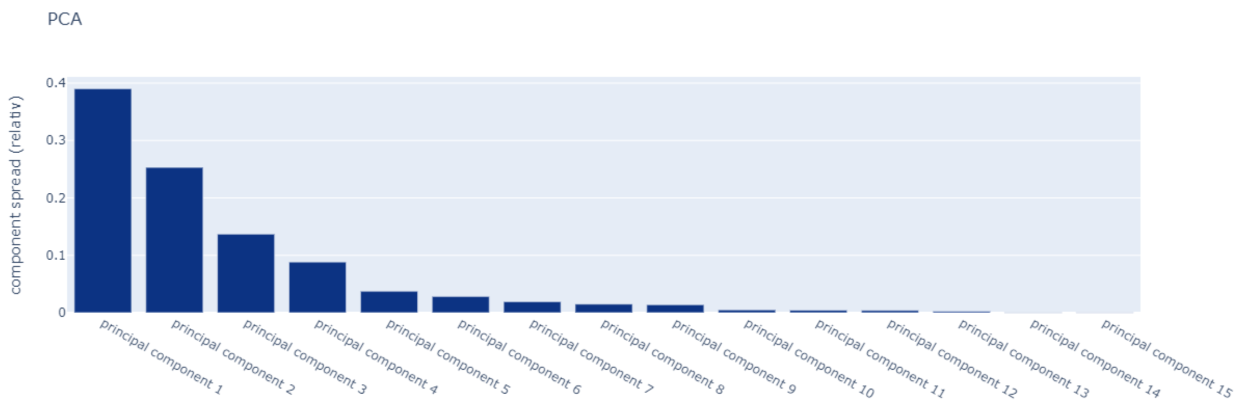


Figure 21 PCA

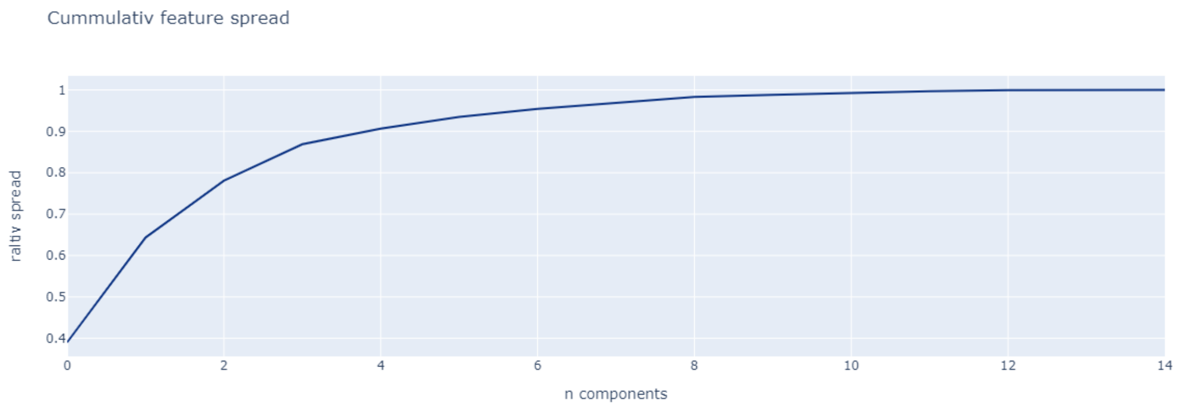


Figure 22 Cumulative feature spread

5.3 Bias identification (Notebook chapter 4)

When keeping the goal in mind, of creating a more accurate power output forecast, the possible factors of relevant biases can be narrowed down to the following features:

- Time of the day
- Seasonality (months)
- Seasonality (sine and cosine)
- Temperature
- Wind direction
- Position (front/back)
- Wind speed

To get a clear and more over viewable picture, the 3-hour lead time will be considered first. After that, all 3 lead times are compared (3, 6, and 12 hours). On each y-axis, the deviation of the forecast is displayed (measured power output minus the predicted power output based on the forecasted wind speeds and temperature). To not falsify the interpretation of the biases, only the non-interpolated values are taken into account for this step.

Looking at the box plot of the deviation against the time of day (transformed with the sine function) a clear pattern can be spotted. The deviation changes depending on the time in a curve like arrangement. The reason for this is unknown, but it most likely depends on the accuracy of the wind speed forecast being more or less accurate during certain daytimes.

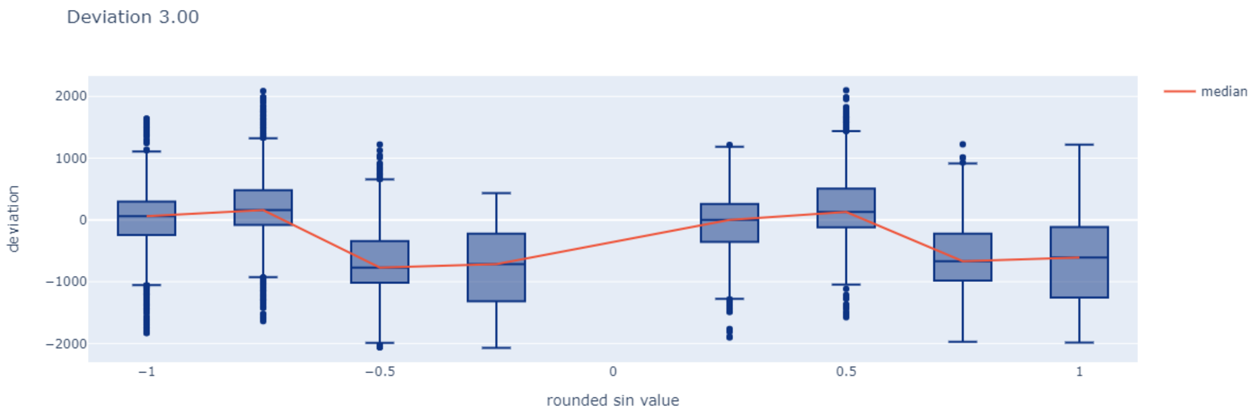


Figure 23 Power output deviation against day sine (3 hours)

A similar pattern, apart from different overall deviation values can be spotted, when comparing all lead time deviations. Thus, a systematic deviation, correlating to the time of day can be confirmed for all three lead times.

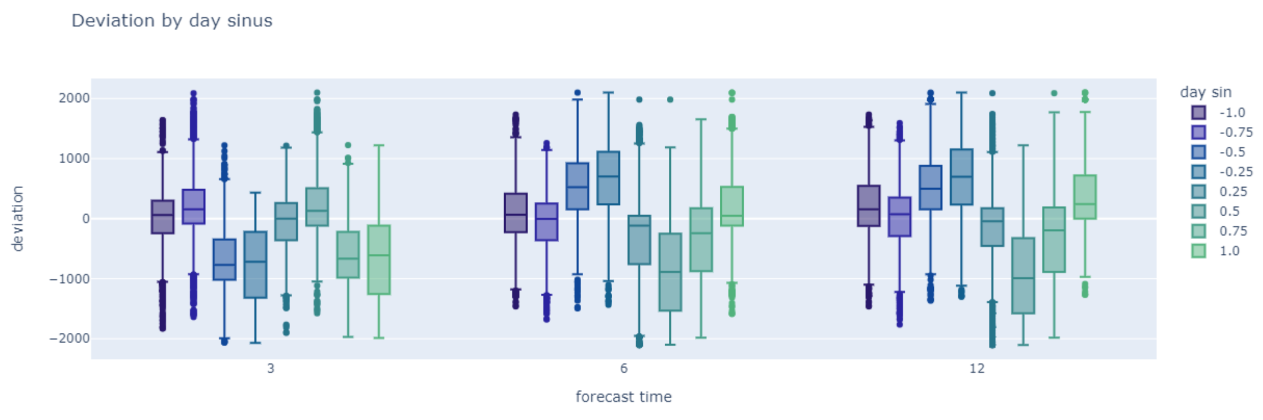


Figure 24 Power output deviation against day sine (3,6, and 12 hours)

Continuing with the time related features, the deviation over the year (by function or month) is not quite conclusive. Each month seems more or less similar, although the median value as well as the value spread could suggest a systematic bias.

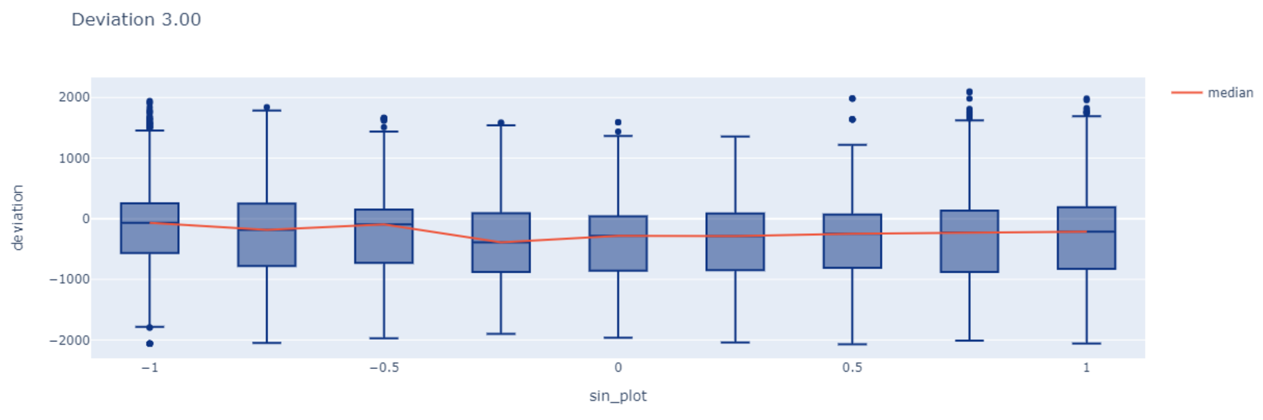


Figure 25 Power output deviation against year sine (3 hours)

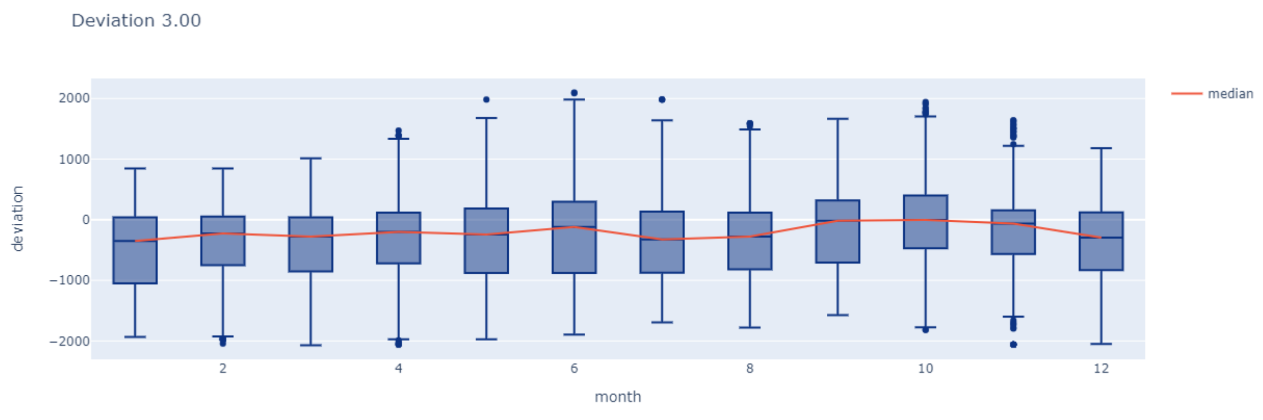


Figure 26 Power output deviation against months (3 hours)

When considering all lead times, a different picture emerges. The 6- and 12-hours deviation show a clear difference in median values as well as the .25 and .75 percentile range, depending on the month. But it is important to note, that there are only two years of data available. This could falsify the conclusion of a systematic bias. Nonetheless, systemic deviation depending on the month will be assumed.

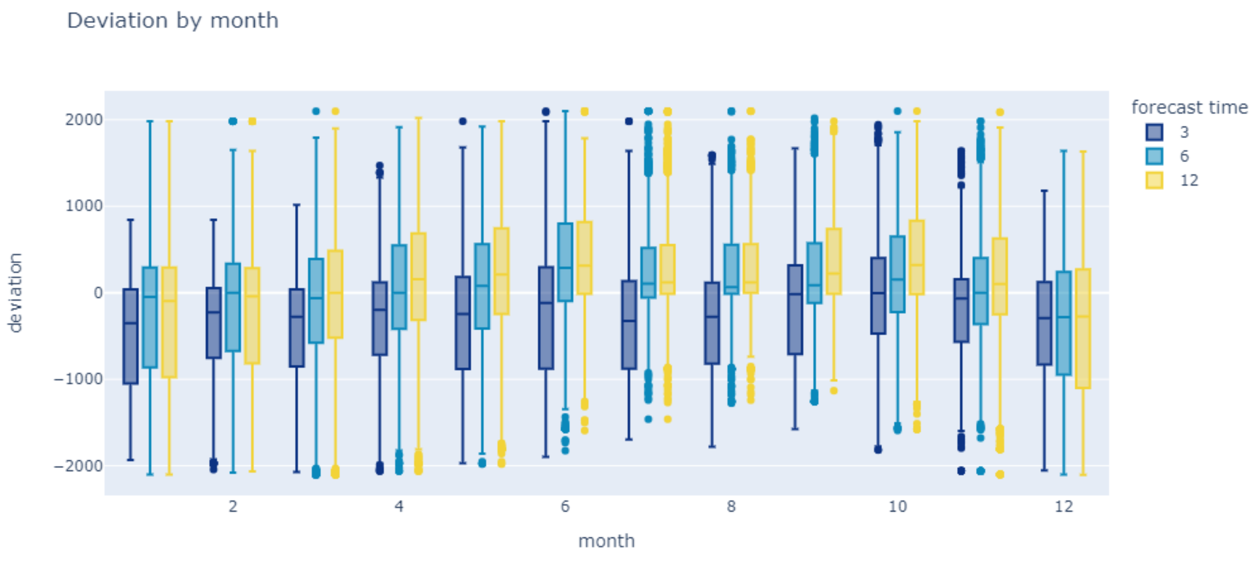


Figure 27 Power output deviation against months (3, 6, and 12 hours)

A clear systematic deviation in respect to the temperature can be seen. The higher the temperature, the more the deviation shifts to the positive. Although, the amount of data points for the lower and higher temperatures is smaller than in the middle, the same trend appears across the board.

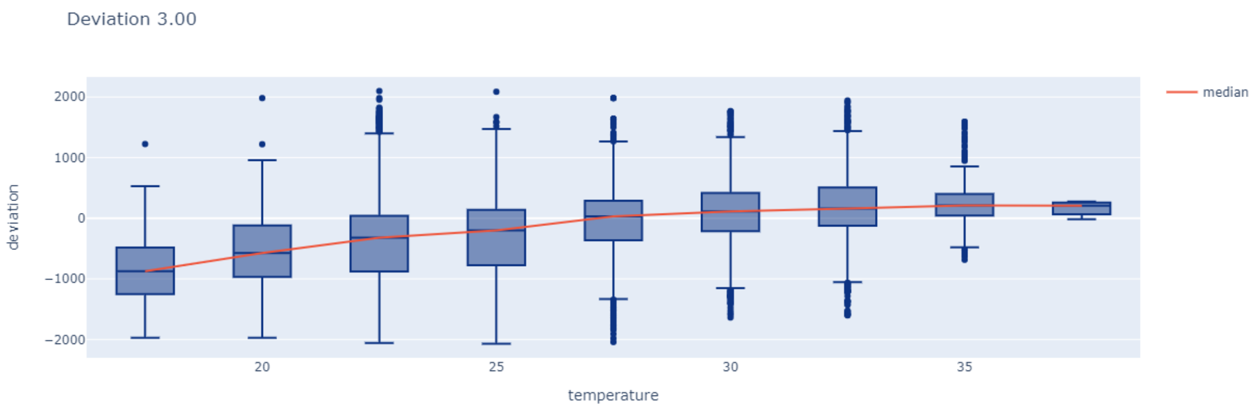


Figure 28 Power output deviation against temperature (3 hours)

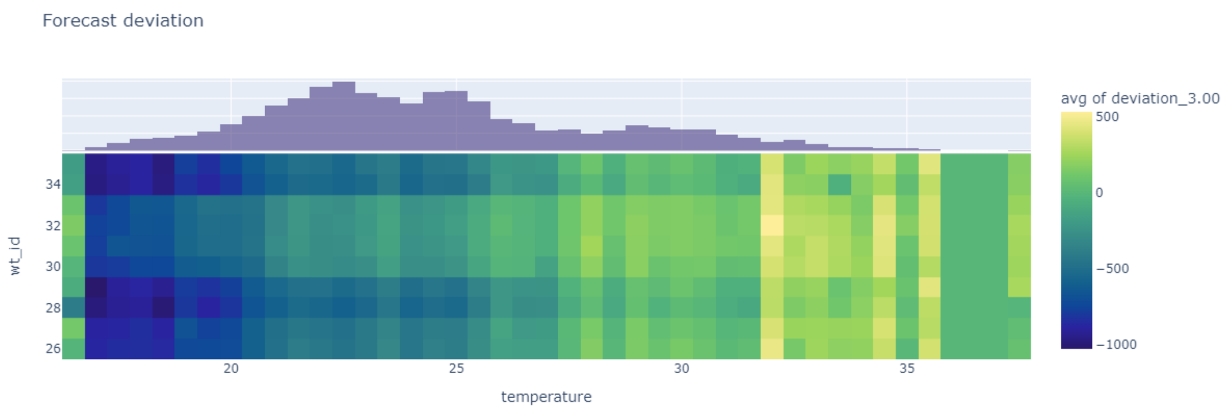


Figure 29 Power output deviation against temperature (3 hours)

The same pattern can be seen in all three lead time deviations, although the data distribution for each temperature varies. The most upper and lower values can be ignored, because of the lack of data. With all that being said, a systematic bias in regards of temperature can be confirmed for all lead times. A possible reason could be, that the environmental temperature has an impact on the physical systems of the wind turbines. Adding or reducing friction due to different thermal expansions of the materials. Another reason could be, that the wind speeds change in regards of the temperature, as stated in chapter 5.1.

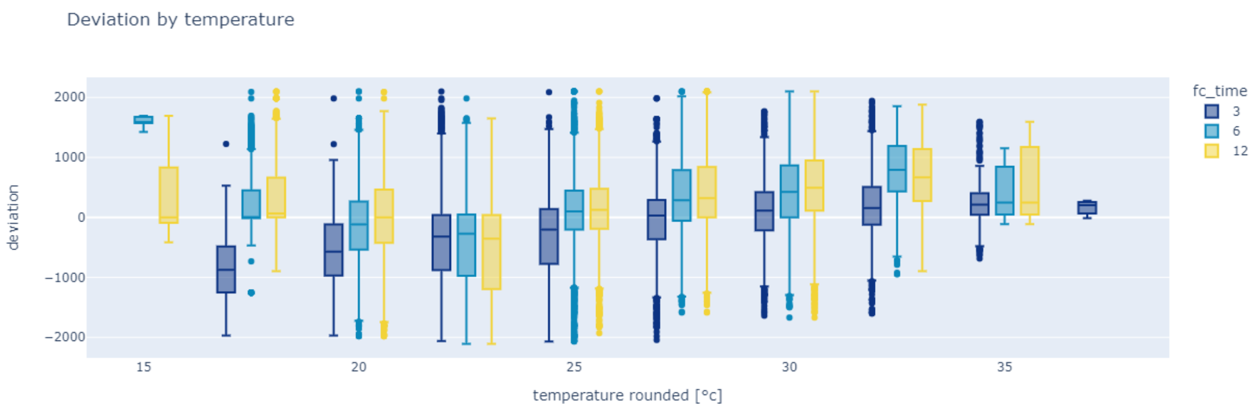


Figure 30 Power output deviation against temperature (3,6, and 12 hours)

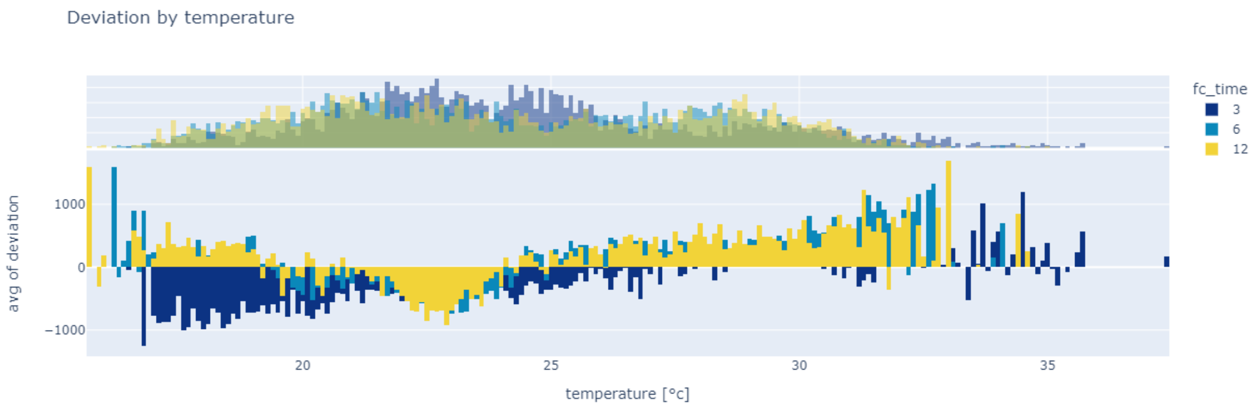


Figure 31 Power output deviation against temperature (3, 6, and 12 hours)

Looking at the deviation in regards of the wind direction, no clear pattern can be made out. It looks like random noise for all lead times. The only noteworthy aspect is, that the wind direction is between 100° and 150° in most cases.

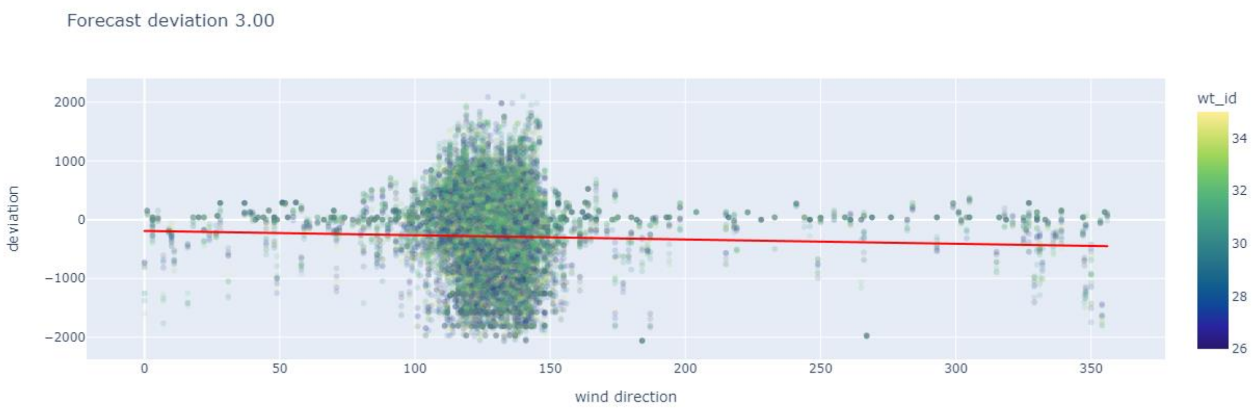


Figure 32 Power output deviation against wind direction (3 hours)

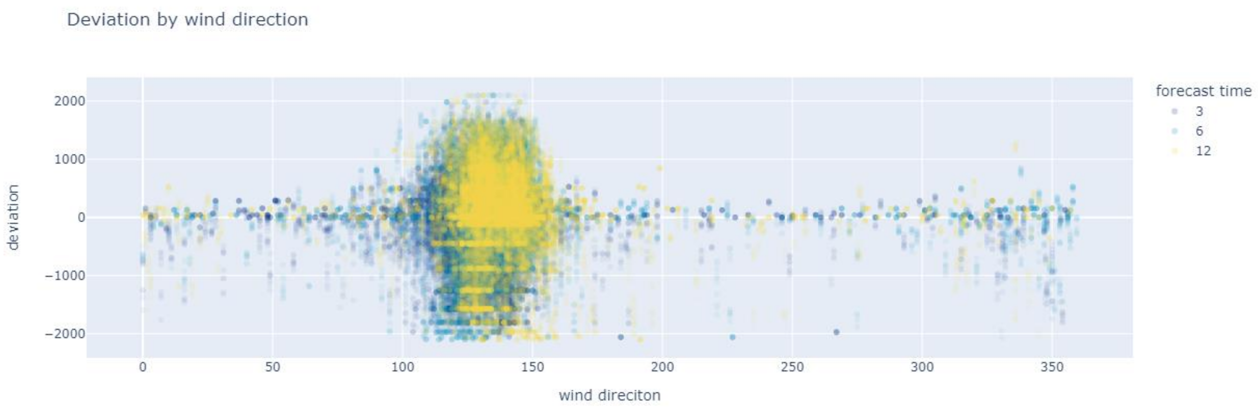


Figure 33 Power output deviation against wind direction (3, 6, and 12 hours)

A systematic bias can be seen in the front and back rows. The front row turbines tend to deviate less than the back row ones. This can be explained by wind wake. Meaning, that the back row turbines catch less of the wind because they are shielded by the turbines in the front.

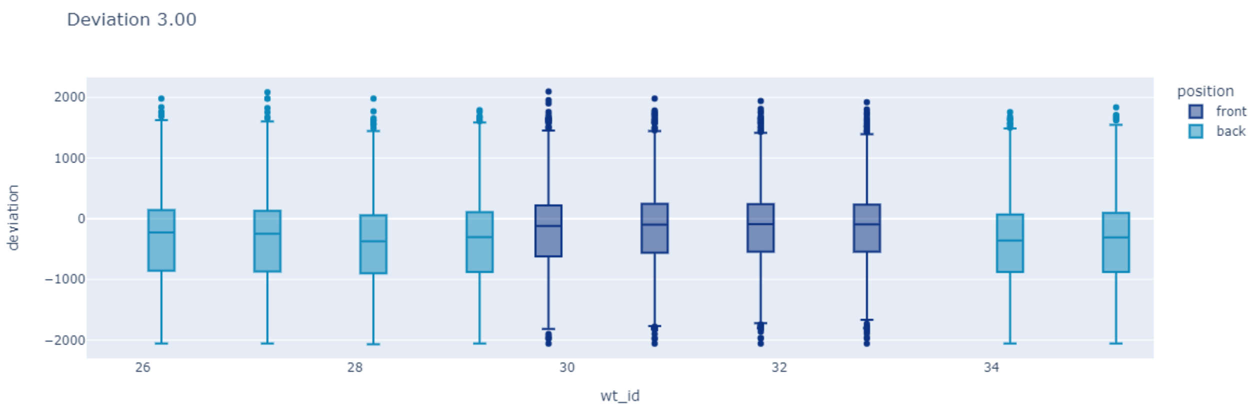


Figure 34 Power output deviation against position (3 hours)

The same tendency can be noticed for all three lead times. Henceforth, a systematic bias concerning the position is confirmed.

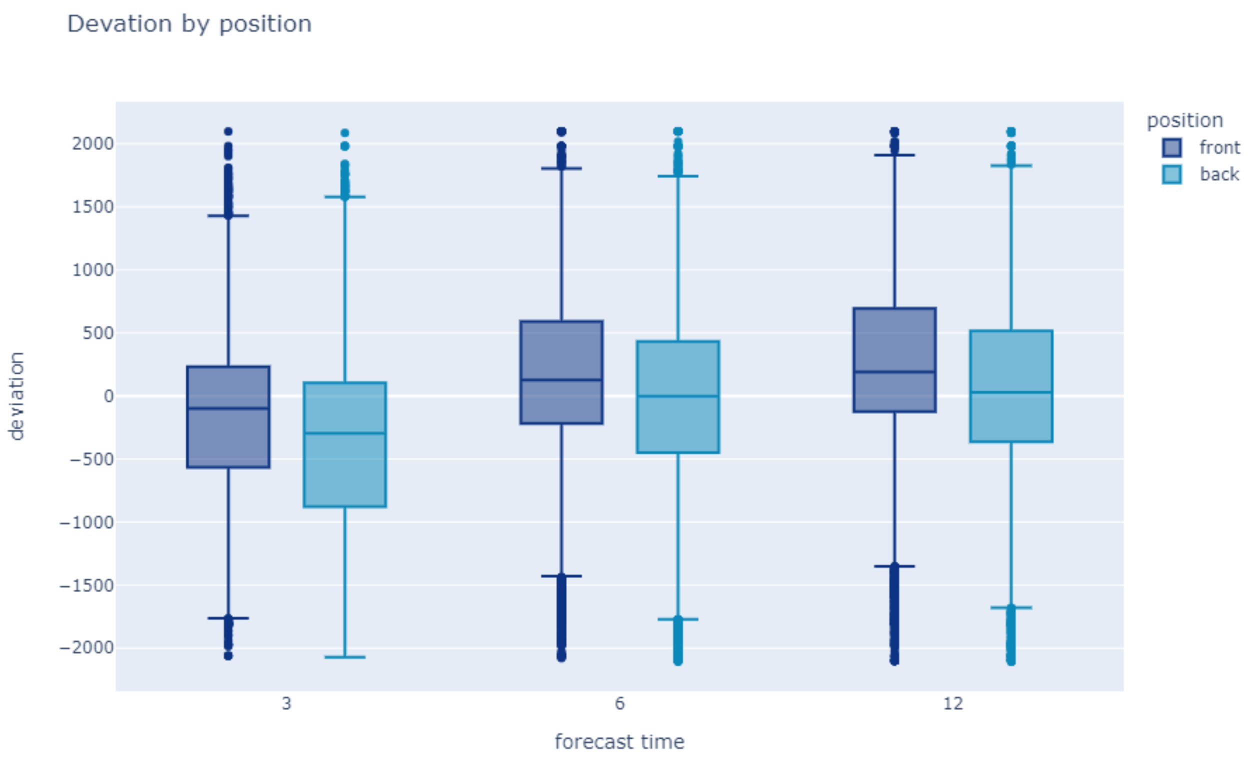


Figure 35 Power output deviation against position (3, 6, and 12 hours)

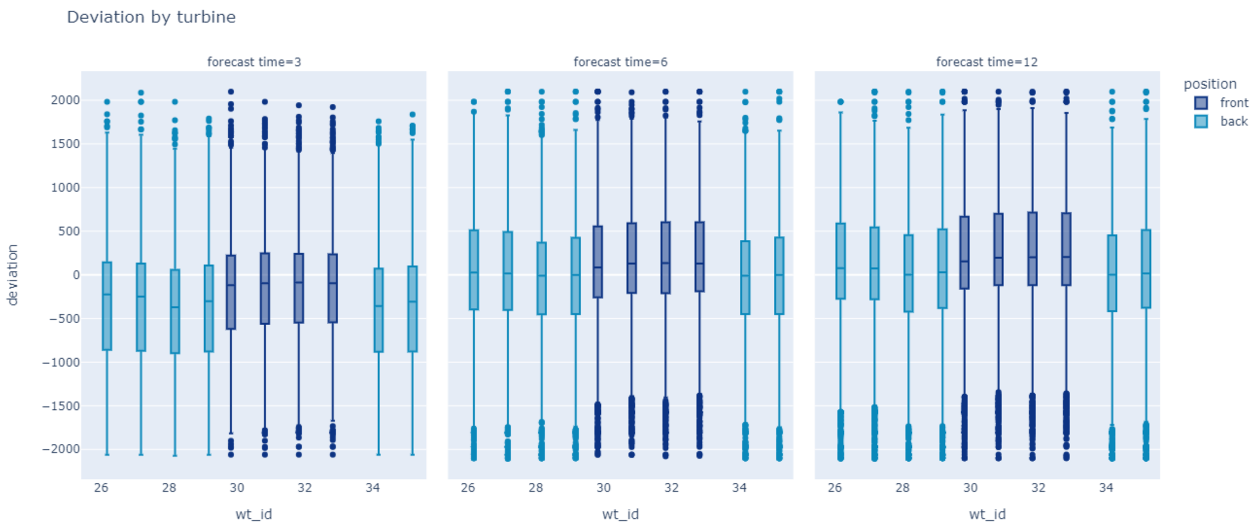


Figure 36 Power output deviation against position (3, 6, and 12 hours)

Looking at the deviation against the forecasted wind speed, a clear pattern emerges among all lead times. Once again, the deviation values themselves differ depending on the lead time. But the overall trend stays the same. It seems unusual, that the deviation gets the highest at the cap of the power curve, 10 meters per second. This could be a hint, that the power curve table does not scale suitably for this wind park.

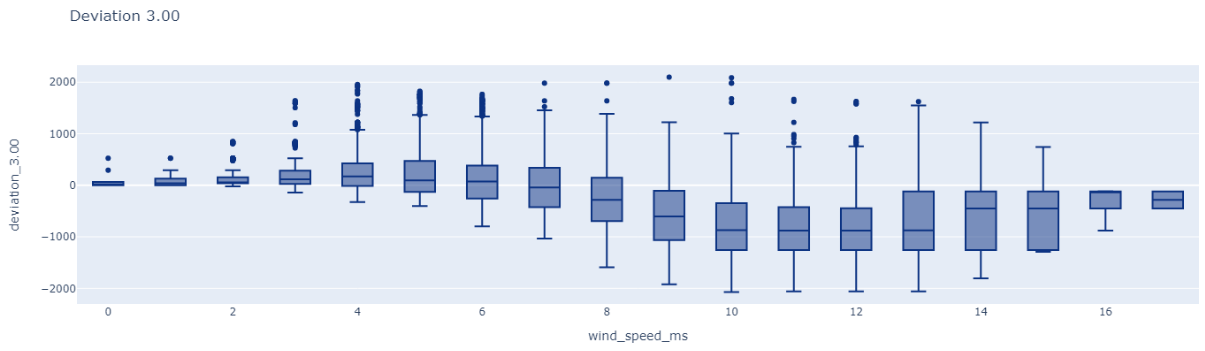


Figure 37 Power output deviation against wind speed (3 hours)

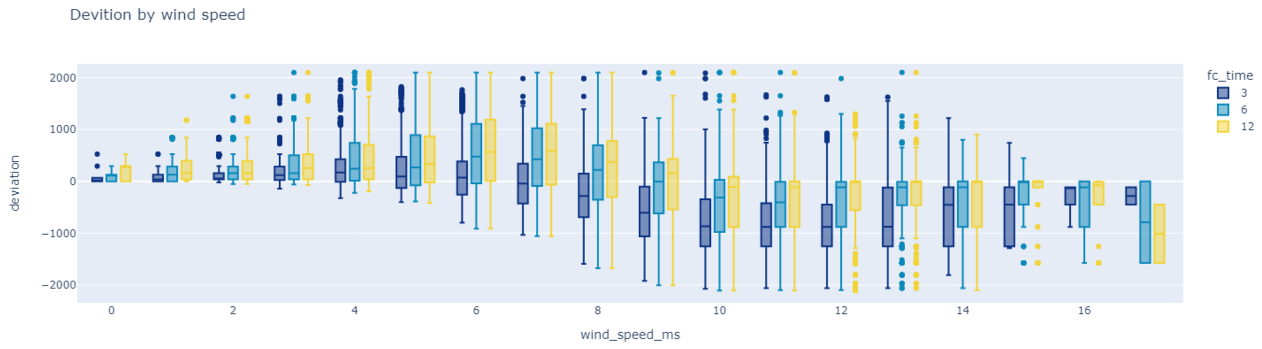


Figure 38 Power output deviation against wind speed (3, 6, and 12 hours)

A last thing to note is, that the deviation per lead time also differs and shows a clear trend. But the deviation is not as expected. Normally, it would make sense, that the forecast is more accurate for lower lead times. But this does not seem to be the case. Not only is the 3-hour prediction deviation median the highest, but also the .25 and .75 percentile range is the biggest. But to spot a clear trend, the other lead times would have to be considered.

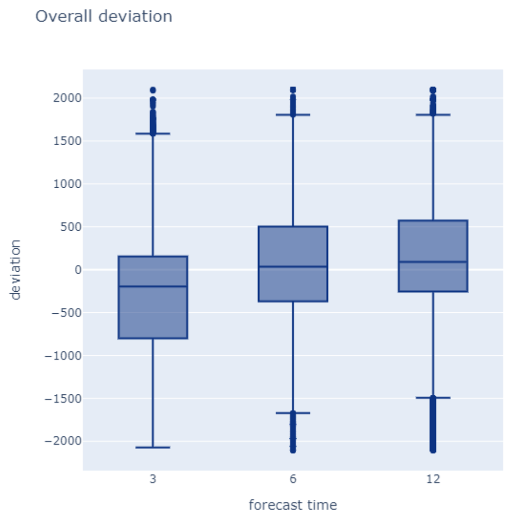


Figure 39 Power output deviation against lead time

5.4 Conventional machine learning (Notebook chapter 5)

In total, six models are created, one for each lead time (3, 6, and 12 hours), one with and one without interpolated data. This approach should reveal, if it is recommendable to interpolate the data to gain a more accurate forecast with machine learning or not. This task is automated by a custom written class, which can be found in the notebook chapter 5.4.

The models (random forest regressors) are only optimized for two parameters. The number of estimators and the maximum number of sample leaves. To speed up the process, the first 12 Fibonacci numbers are taken as possible values, removing the duplicate one in the beginning of the array: [1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233]. With said method, a wide range of values can be tested, without running the model for every single increment of the two parameters.

After splitting the data into the training, validation, and testing set, the number of estimators was optimized first. As can be seen in the fitting graph, the curve of the root mean squared error (rmse) flattens out quickly. Meaning, that the number of estimators does not have that big of an impact on the model's overall performance. A reason could be, that the given data is not too complex and consists only of few features. The graph on the left is with the interpolated data, the one on the right without.

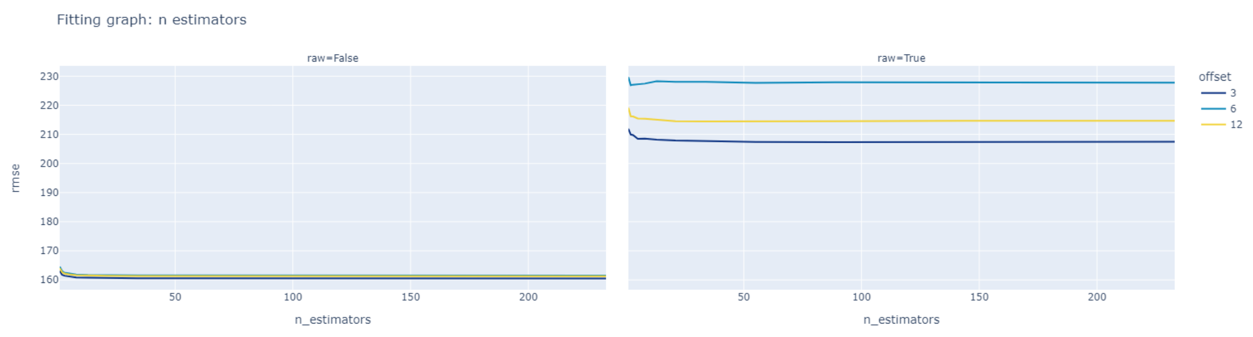


Figure 40 Number of estimators fitting graph

By interpreting these two fitting graphs and deciding on the minimum number of estimators with a reasonable low rmse, the following values are set for the models:

Lead time	Dataset	Estimators
3 hours	raw	55
3 hours	Interpolated	34
6 hours	raw	55
6 hours	Interpolated	34
12 hours	raw	34
12 hours	Interpolated	34

With the optimized number of estimators for each model, the process is run again to get the best number of maximum sample leaves for each lead time and dataset combination. Once again, the fitting graphs with the raw and interpolated data show a similar trend. Both reach their lowest errors with very small values.

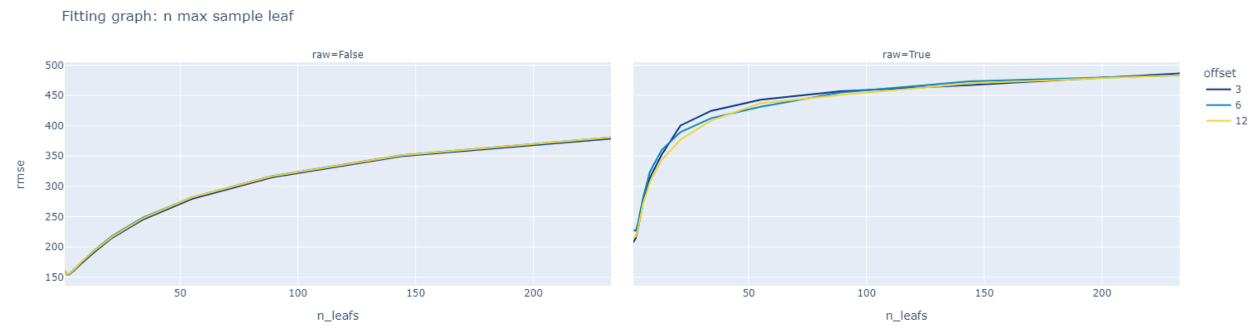


Figure 41 Number of max. sample leaves fitting graph

The following optimal number of maximum sample leaves can be derived from these results:

Lead time	Dataset	Leaves
3 hours	raw	1
3 hours	Interpolated	2
6 hours	raw	1
6 hours	Interpolated	2
12 hours	raw	1
12 hours	Interpolated	2

Looking at the feature importance of the models in Figure 42, a comparable trend emerges among the different models. The most important feature, as could be anticipated, is the forecasted power value. Although not matching perfectly, it can be seen, that the identified systematic biases play an important role in the feature importance of each model. The air density has the lowest impact for all models. This can be explained by the fact, that this feature was calculated using the temperature, and has not a wide spread of values. It is interesting to see, that the forecasted wind direction is the 4th most important feature, considering the fact, that no clear bias based on the deviation could be seen. An explanation for this phenomenon is open to discussion.

Theoretically, the models could be cut down by one or two features each. But considering the already low number of features and simplicity this will not be necessary.

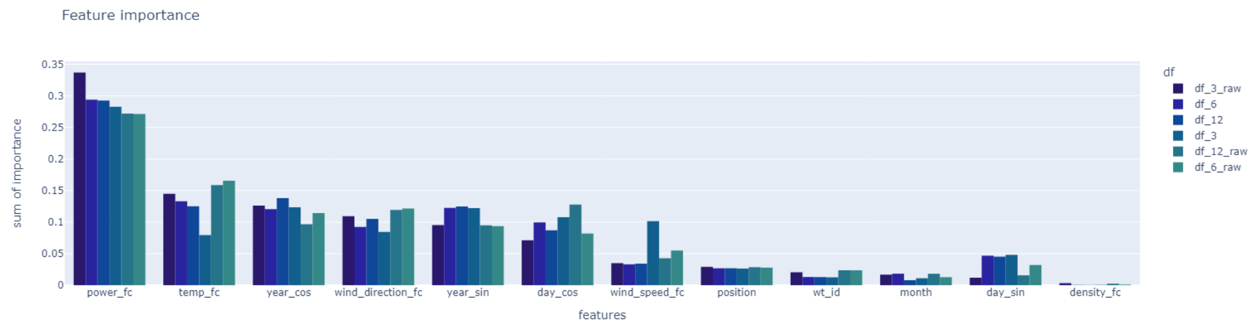


Figure 42 Feature importance by model

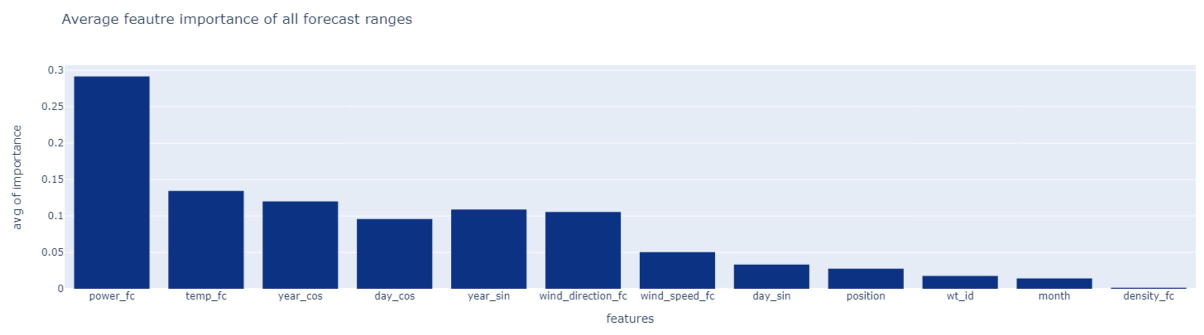


Figure 43 Feature importance overall

When comparing the theoretical forecasted values, based on the applied power curve table alone, the results are clear: The overall accuracy of the forecast can be drastically improved by applying machine learning to the problem.

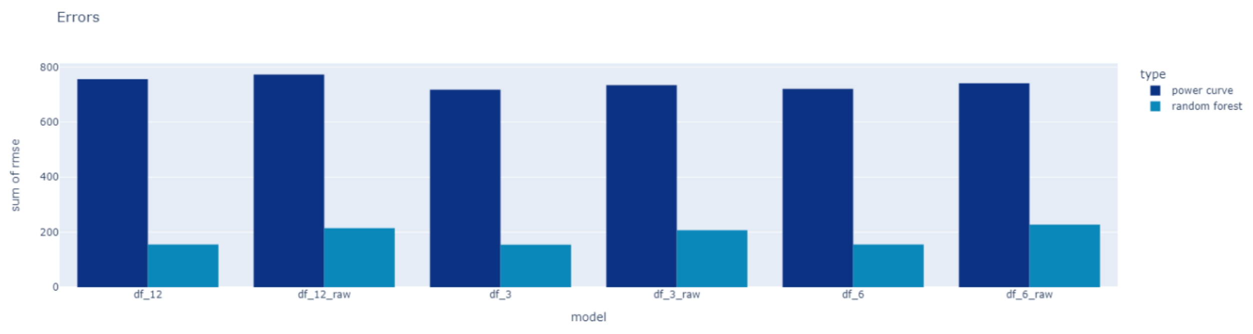


Figure 44 Model errors by model

Interesting to note is, that models which used the interpolated values (df_3, df_6, df_12) are slightly more accurate than the ones with raw data only (df_3_raw, df_6_raw, df_12_raw). The main reason could be, that more data is available for training the models.

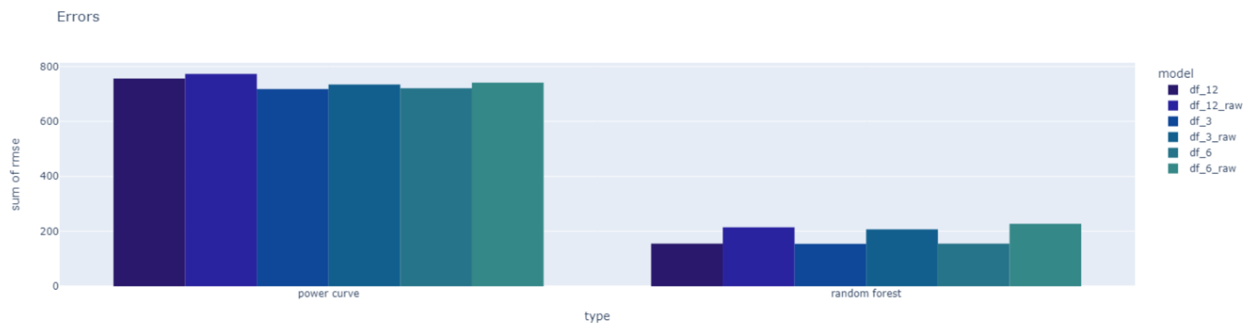
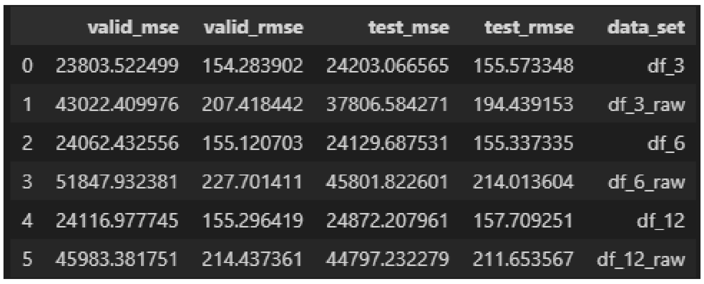


Figure 45 Model errors by type

Running the trained models on the test data set it seems that they generalise well on the data at hand. Meaning the models are not overfitted on the training or validation data.



	valid_mse	valid_rmse	test_mse	test_rmse	data_set
0	23803.522499	154.283902	24203.066565	155.573348	df_3
1	43022.409976	207.418442	37806.584271	194.439153	df_3_raw
2	24062.432556	155.120703	24129.687531	155.337335	df_6
3	51847.932381	227.701411	45801.822601	214.013604	df_6_raw
4	24116.977745	155.296419	24872.207961	157.709251	df_12
5	45983.381751	214.437361	44797.232279	211.653567	df_12_raw

Figure 46 Validation and test errors

5.5 Deep Learning and graph neural networks (Notebook chapter 6)

5.5.1 Model possibilities and limits

When it comes to generating model ideas and modelling, the limiting factors are the type of data available and the limitations of the machine learning frameworks. E.g., are most of the graph neural networks (GNN) models only suited for classification problems, and not regression. This poses some problems for this specific case. Because most of the available scada features are continuous and not clearly separatable into classes.

Furthermore, it would be crucial to have the exact layout of the wind park. Otherwise, the graph, or edges to be precises, will have to be assumed. If the assumptions are not correct, the model's output will be unusable and will result in: Garbage in, garbage out [23].

For this project, only the below mentioned models 3 and 4 will be tested. The reason being the type of the given data and the time constraints.

The following model suggestions should give an idea of the possible application of GNNs in the field of wind parks:

Model 1: Graph level		
Description	Classify the overall structure and layout of a wind park in terms of wind wake. If enough graphs are available, this can be used to optimize layouts of new wind parks or improve the efficiency, if adding new turbines to an already existing wind park.	
Graph structure	Graph Features:	Geo location regarding if it is onshore or offshore
	Nodes:	Wind turbines
	Node Features:	Wind speed efficiency and wind wake potential
	Edges:	Influence in regard to wind direction on neighbouring wind turbines
	Edge Features:	Distance between the turbines

Model 2: Graph level	
Description	Classify the overall efficiency regarding the predicted power deviation from the predicted values (according to the power curve table) and the measured values. Can be applied in the same way as model 1.
Graph structure	Graph Features: Geo location regarding if it is onshore or offshore Nodes: Wind turbines Node Features: Wind speed efficiency and wind wake potential Edges: Influence regarding wind direction on neighbouring wind turbines Edge Features: Distance between the turbines or an influence coefficient

Model 3: Node level	
Description	Classify nodes based on their features and positioning in a wind park. Could be used to predict the class (e.g., overall efficiency, deviation from the power curve table, deviation from the predicted metrics) based on position and features for potential new turbines in an existing wind park.
Graph structure	Nodes: Wind turbines Node Features: Precited value deviations, e.g., predicted wind speed Edges: Positional relation between neighbouring turbines Edge Features: Distance between the turbines or an influence coefficient. This could also be neglected.

Model 4: Node level	
Description	Outlier detection, based on structure (edges) or properties (node features) of all turbines in a wind park. Can hint at anomalies, possible failures of technical components and maintenance requirements of individual turbines.
Graph structure	Nodes: Wind turbines Node Features: All available and aggregated data of each turbine, as well as deviation in output power and predicted power Edges: Positional relation between neighbouring turbines Edge Features: Distance between the turbines or an influence coefficient

Model 5: Edge level	
Description	Link prediction or relative positional prediction based on given features of a turbine within an existing wind park. Can aid when expanding a wind park with new turbines and the optimal position is previously unknown.
Graph structure	Nodes: Wind turbines Node Features: All available and aggregated data of each turbine, as well as deviation in output power and predicted power Edges: Positional relation between neighbouring turbines Edge Features: Distance between the turbines

5.5.2 Data restructuring and aggregation

In order to generate a graph out of the given scada datapoints and additional information, the available features must be aggregated. This is done by calculating the mean, the median, and the standard deviation of the following predicted features per turbine: Wind speed, wind direction, Temperature, and power output.

To check, if these three values of each feature hold any meaning, the distribution of these values is important. Should they be spread out randomly, then the aggregated values are not very conclusive. But if a certain pattern can be spotted in the distribution, the aggregated values can convey certain properties for each turbine. Looking at the distribution of errors of the 4 values, the following conclusion can be made:

- The power prediction errors (Figure 47) are distributed mostly symmetrical around a centre. There are clear outliers, but the overall distribution Is mostly evenly spread out. Conclusion: The aggregated values convey certain properties.

- The temperature prediction errors (Figure 48) strongly suggest a gaussian or normal distribution [24]. Therefore, the curve could be reconstructed, by only using the mean and the standard deviation. Conclusion: The aggregated values convey certain properties.

- The wind speed prediction errors (Figure 49) form an almost perfect gaussian distribution. Therefore, the curve could be reconstructed, by only using the mean and the standard deviation [24]. Conclusion: The aggregated values convey certain properties.

- The wind direction prediction errors (Figure 50) are distributed around two or three peaks for each turbine. The peaks, their heights and distances are not similar between the different turbines. Conclusion: The aggregated values hold little information about this property.

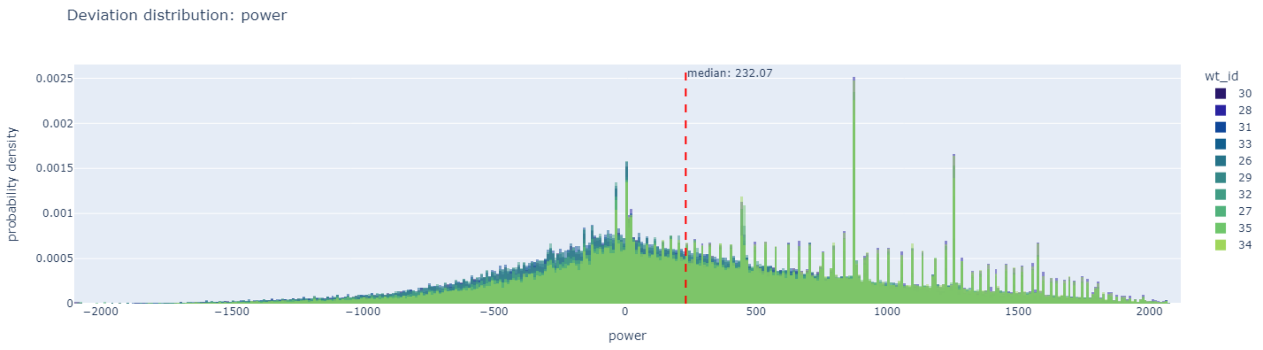


Figure 47 Power forecast deviation

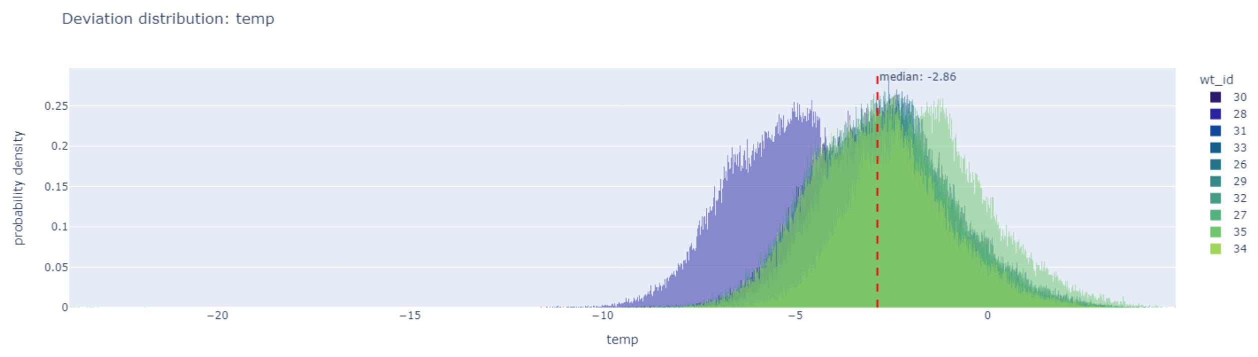


Figure 48 Temperature forecast deviation

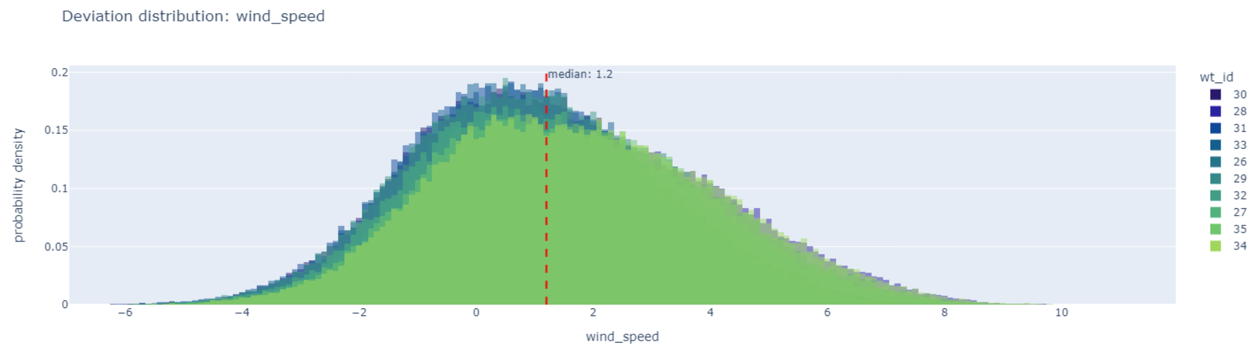


Figure 49 Wind speed forecast deviation

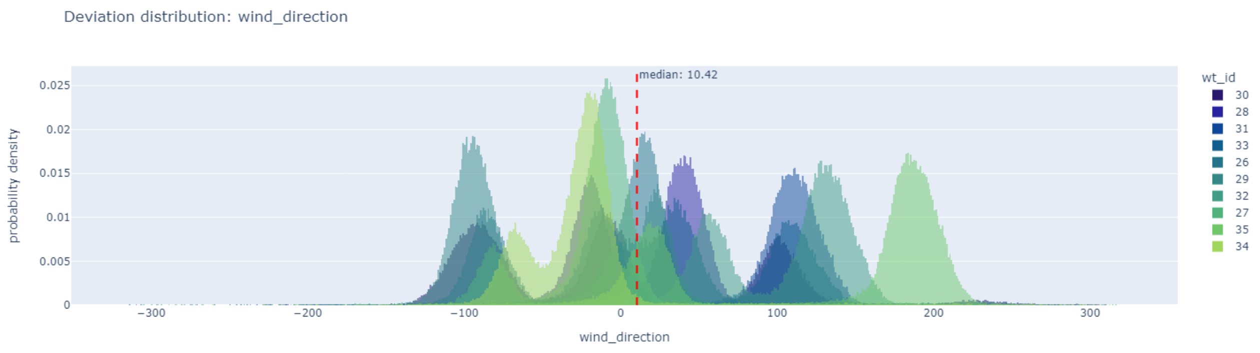
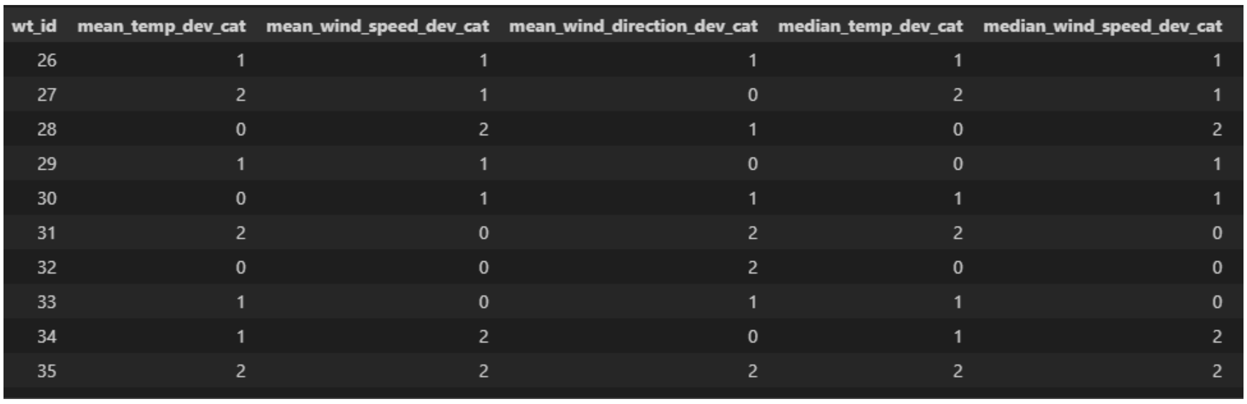


Figure 50 Wind direction forecast deviation

With these conclusions, the node features are set. Because the model should later predict the overall deviation in power output, all three aggregations of the power output values are not put as node features. The mean power output deviation is set as the node class. With this setup, the model predicts the overall mean of the power deviation for turbines, based on previously mentioned properties.

Because the concerning model is a classifier, the node features and classes must be put into 3 classes. If more data would be available, the number of classes could be increased, respectively. If more than 3 classes would be chosen, the population per class would be too low to train the model properly.

The torch geometric library takes numpy arrays or matrices as inputs. Therefore, the data frame is transformed and simplified. Each column represents one of the before mentioned features, put as a category.



wt_id	mean_temp_dev_cat	mean_wind_speed_dev_cat	mean_wind_direction_dev_cat	median_temp_dev_cat	median_wind_speed_dev_cat
26	1	1	1	1	1
27	2	1	0	2	1
28	0	2	1	0	2
29	1	1	0	0	1
30	0	1	1	1	1
31	2	0	2	2	0
32	0	0	2	0	0
33	1	0	1	1	0
34	1	2	0	1	2
35	2	2	2	2	2

Figure 51 Node features

The edge structure is based on the direct positional relationship between the wind turbines. From the assumed layout in Figure 51, a two-dimensional matrix is derived, describing the shown relations.

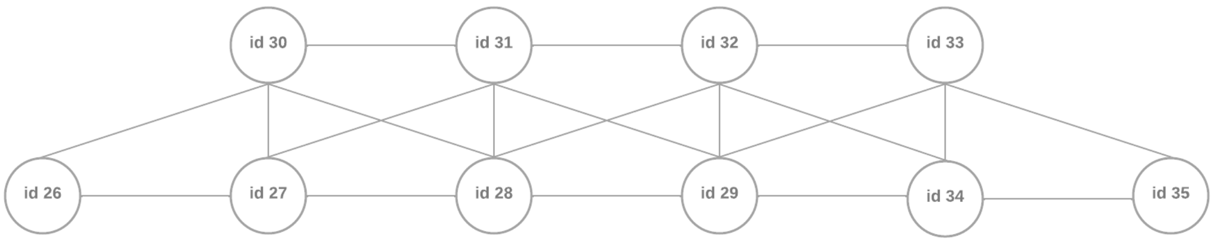


Figure 52 Wind park graph layout

5.5.3 Model 3 - Node classification

The node classifier consists of two parts:

- A standard feed forward neural network or multi-layer perceptron [25] with two hidden layers (baseline model). The activation function on each layer is the rectified linear unit (ReLu) function [10].
- A graph convolutional neural network, built on top of the baseline model. This adds a graph convolution layer bevor the input layer of the baseline model.

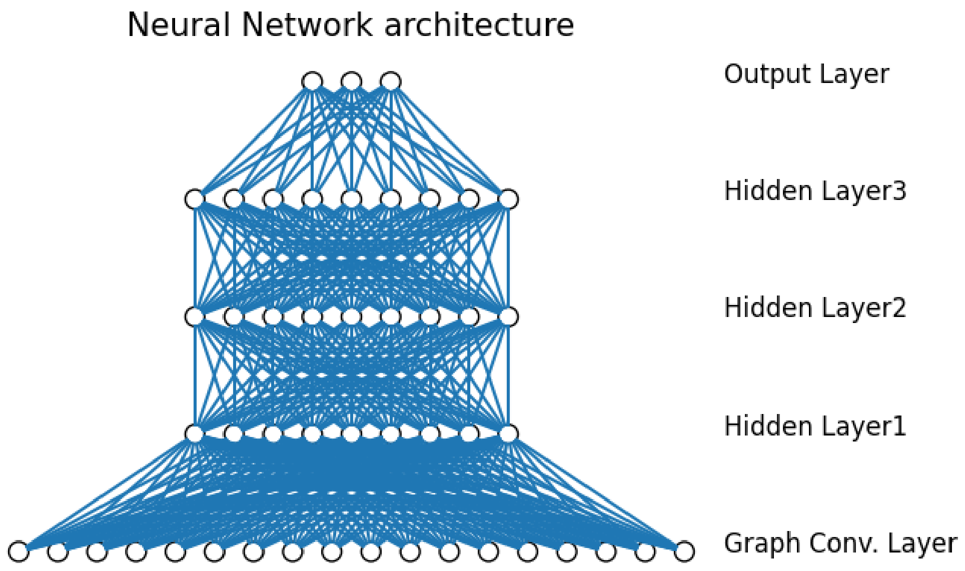
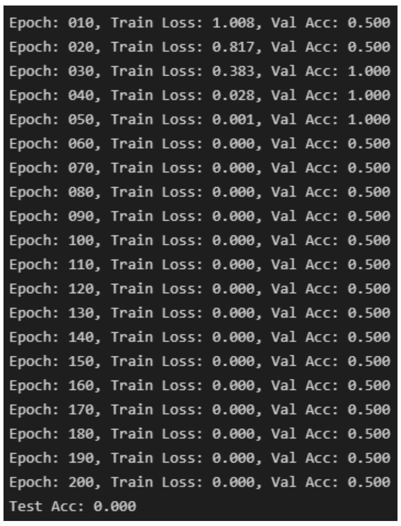


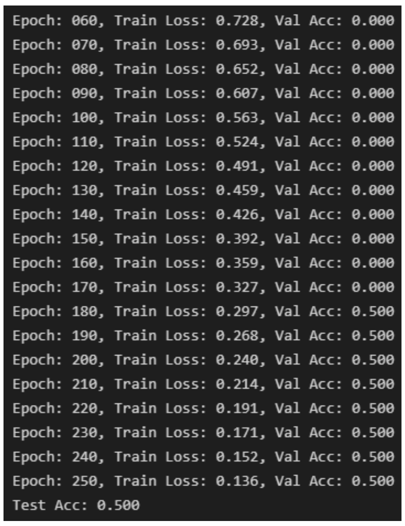
Figure 53 Neural network architecture

Because of the very limited data set with 10 nodes and 20 edges, the performances of the baseline model and the GNN can hardly be compared and evaluated. The accuracies change drastically with each run. But it seems, that the GNN is more consistent across validation and test set than the baseline model:



```
Epoch: 010, Train Loss: 1.008, Val Acc: 0.500
Epoch: 020, Train Loss: 0.817, Val Acc: 0.500
Epoch: 030, Train Loss: 0.383, Val Acc: 1.000
Epoch: 040, Train Loss: 0.028, Val Acc: 1.000
Epoch: 050, Train Loss: 0.001, Val Acc: 1.000
Epoch: 060, Train Loss: 0.000, Val Acc: 0.500
Epoch: 070, Train Loss: 0.000, Val Acc: 0.500
Epoch: 080, Train Loss: 0.000, Val Acc: 0.500
Epoch: 090, Train Loss: 0.000, Val Acc: 0.500
Epoch: 100, Train Loss: 0.000, Val Acc: 0.500
Epoch: 110, Train Loss: 0.000, Val Acc: 0.500
Epoch: 120, Train Loss: 0.000, Val Acc: 0.500
Epoch: 130, Train Loss: 0.000, Val Acc: 0.500
Epoch: 140, Train Loss: 0.000, Val Acc: 0.500
Epoch: 150, Train Loss: 0.000, Val Acc: 0.500
Epoch: 160, Train Loss: 0.000, Val Acc: 0.500
Epoch: 170, Train Loss: 0.000, Val Acc: 0.500
Epoch: 180, Train Loss: 0.000, Val Acc: 0.500
Epoch: 190, Train Loss: 0.000, Val Acc: 0.500
Epoch: 200, Train Loss: 0.000, Val Acc: 0.500
Test Acc: 0.000
```

Figure 55 Training accuracy baseline



```
Epoch: 060, Train Loss: 0.728, Val Acc: 0.000
Epoch: 070, Train Loss: 0.693, Val Acc: 0.000
Epoch: 080, Train Loss: 0.652, Val Acc: 0.000
Epoch: 090, Train Loss: 0.607, Val Acc: 0.000
Epoch: 100, Train Loss: 0.563, Val Acc: 0.000
Epoch: 110, Train Loss: 0.524, Val Acc: 0.000
Epoch: 120, Train Loss: 0.491, Val Acc: 0.000
Epoch: 130, Train Loss: 0.459, Val Acc: 0.000
Epoch: 140, Train Loss: 0.426, Val Acc: 0.000
Epoch: 150, Train Loss: 0.392, Val Acc: 0.000
Epoch: 160, Train Loss: 0.359, Val Acc: 0.000
Epoch: 170, Train Loss: 0.327, Val Acc: 0.000
Epoch: 180, Train Loss: 0.297, Val Acc: 0.500
Epoch: 190, Train Loss: 0.268, Val Acc: 0.500
Epoch: 200, Train Loss: 0.240, Val Acc: 0.500
Epoch: 210, Train Loss: 0.214, Val Acc: 0.500
Epoch: 220, Train Loss: 0.191, Val Acc: 0.500
Epoch: 230, Train Loss: 0.171, Val Acc: 0.500
Epoch: 240, Train Loss: 0.152, Val Acc: 0.500
Epoch: 250, Train Loss: 0.136, Val Acc: 0.500
Test Acc: 0.500
```

Figure 54 Training accuracy GCN

5.5.4 Model 4 - Node outlier detection

The outlier detection model differentiates between two different outliers, which both factor into the final outlier score of each node or turbine [26]:

- Structural outlier: Nodes which are densely connected in contrast to lesser connected nodes
- Contextual outlier: Node with attributes or features which differ from their neighbouring nodes

Unlike all the previous models, this one is unsupervised. Meaning, that there is no way to test the accuracy of the model. One way to interpret the results, is to manually check if the turbines with a higher outlier score seem to be possible outliers. Because the data structure and number of nodes is so small, it should be possible to identify where the score comes from (structural or contextual).

The turbines 27, 28, and 29 seem to have the highest outlier scores. Considering the edges, these three don't seem like structural outliers. Therefore, their features must show some anomalies.

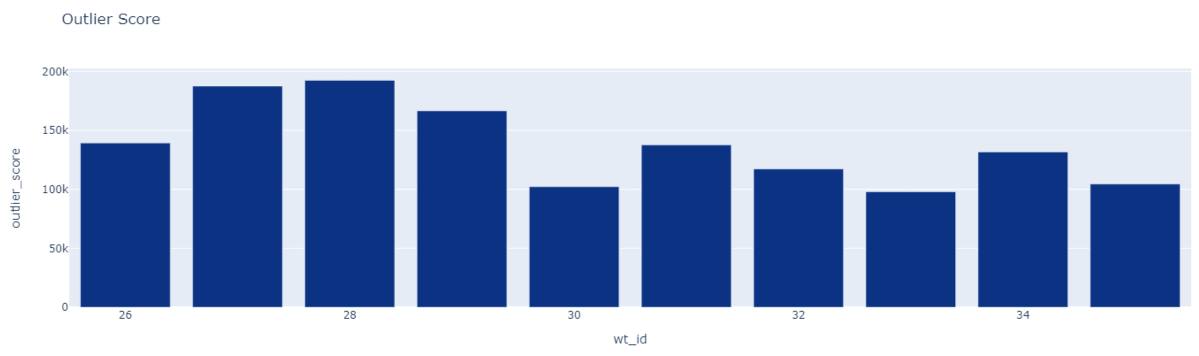


Figure 56 Outlier scores

Looking at the aggregated data values, there are some features which could cause the higher score. When looking at the values no obvious reason can be detected easily. A closer analysis would be needed.

wt_id	26	27	28	29	30	31	32	33	34	35
mean_power_dev	326	340	435	388	217	173	172	174	421	381
mean_temp_dev	-3	-3	-5	-3	-3	-3	-3	-3	-3	-2
mean_wind_speed_dev	1	1	2	2	1	1	1	1	2	2
mean_wind_direction_dev	-16	-17	16	-44	-7	62	99	51	-27	127
median_power_dev	258	273	378	322	154	110	112	116	364	318
median_temp_dev	-3	-3	-5	-3	-3	-3	-3	-3	-3	-2
median_wind_speed_dev	1	1	2	1	1	1	1	1	2	2
median_wind_direction_dev	4	-11	26	-83	-13	97	120	40	-25	176
std_power_dev	677	683	684	677	652	636	631	626	683	681
std_temp_dev	2	2	2	2	2	2	2	2	2	2
std_wind_speed_dev	2	2	2	2	2	2	2	2	2	2
std_wind_direction_dev	60	47	47	75	89	70	57	57	45	87

6. Conclusion

6.1 Findings and discussion

Out of the 7 examined possible systematic biases in regard to the forecasted power deviation, 6 could be confirmed: time of the day, months of year, seasonality, temperature, position, and wind speed. With these factors, it is then possible to correct or account for the biases.

With a random forest regressor, the prediction accuracy can be increased dramatically when compared to the power curve table prediction. The feature importance of the models mostly lines up with the identified biases. This confirms that the model was able to account for the systematic biases in the power forecast.

Multiple use cases for graph neural networks can be found within the branch of wind parks. There are multiple use cases for graph level task, such as classifying power efficiency and wind wake based on layouts (conceptual proof). Models for node level tasks include node classification, where properties of wind turbines can be predicted, based on their node features and edges. Outlier detection for individual turbines can be a great tool to find turbines which do not perform as they should, based on their power output and neighbouring turbines. Link or edge predictions also find their uses. Based on certain properties of a turbine the relative position within a wind park can be determined (conceptual proof).

6.2 Possible further steps

- In depth look at the identified systematic biases and finding reasons why these occur within the given wind park
- Building a deep neural network for predictions of power output. Such as a multi-layer perceptron regressor or a convolutional neural network. Comparing the performance to the random forest regressor.
- Creating a link or edge prediction model, based on the given and transformed scada data.
- Apply the node outlier detection and the node classifier with different node features

7. Statement of authorship

I confirm that this technical report, the project, as well as the jupyter notebook were created independently and without the use of any sources and resources, expect the ones listed in this report and the notebook. All text passages which were not written by me are marked as quotes and labelled with their exact origin.

Joël Tauss
Ostermundigen, 07.01.2023: [illegible]______________________________

8. List of illustrations

9. Bibliography

[1] Bloomberg NEF, "The Next Phase of Wind Power Growth in Five Charts," [Online]. Available: https://about.bnef.com/blog/the-next-phase-of-wind-power-growth-in-five-charts/. [Accessed 2 10 2022].

[2] Renewable Enegry Magazine, "Offshore Wind Market to Grow at a CAGR of 21% To 2030," 23 3 2022. [Online]. Available: https://www.renewableenergymagazine.com/wind/offshore-wind-market-to-grow-at-a-20220523 . [Accessed 2 10 2022].

[3] M. J. J. E. B. Shahab Shokrzadeh, "Wind Turbine Power Curve Modeling Using Advanced Parametric and Nonparametric Methods," *Trabsactuibs on Sustainable Energy,* vol. 5, no. 4, pp. 1262 - 1269, 2014.

[4] P. Loshin, "SCADA (supervisory control and data acquisition)," TechTarget, [Online]. Available: https://www.techtarget.com/whatis/definition/SCADA-supervisory-control-and-data-acquisition. [Accessed 17 12 2022].

[5] D. H. H. M. K. M. H.-P. W. Hans Georg Beyer, "Forecast of Regional Power Output of Wind Turbines," Department of Energy and Semiconductor Research, Faculty of Physics, 26111 Oldenburg, Deutschland.

[6] S. M. M. J. J. a. E. B. Shahab Shokrzadeh, "Wind Turbine Power Curve Modeling Using," *Transactions on sustainable energy,* vol. 5, no. 4, pp. 1262-1269, 2014.

[7] Z. C. b. ,. Z. Z. a. Xin Liu a, "Short-term predictions of multiple wind turbine power outputs based on deep neural networks with transfer learning," *Energy,* vol. 217, 2021.

[8] G.-q. L. G.-b. W. b. J.-c. P. H. J. Y.-t. L. Huai-zhi Wang, "Deep learning based ensemble approach for probabilistic wind power," *Applied Energy,* vol. 188, no. 1, pp. 56-70, 2017.

[9] KDnuggets, "Deep Neural Networks," [Online]. Available: https://www.kdnuggets.com/2020/02/deep-neural-networks.html. [Accessed 06 01 2023].

[10] C. T. B. H. MIAP, "The basics of Deep Neural Networks," Towards Data Science, 13 05 2019. [Online]. Available: https://towardsdatascience.com/the-basics-of-deep-neural-networks-4dc39bff2c96. [Accessed 06 01 2023].

[11] E. R. A. P. A. B. W. Benjamin Sanchez-Lengeling, "A Gentle Introduction to Graph Neural Networks," Distill, 02 09 2021. [Online]. Available: https://distill.pub/2021/gnn-intro/. [Accessed 21 12 2022].

[12] A. Menzli, "Graph Neural Network and Some of GNN Applications: Everything You Need to Know," MLOps Blog, 07 12 2022. [Online]. Available: https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications. [Accessed 21 12 2022].

[13] W. Liao, B. Bak-Jensen, J. R. Pillai, Y. Wang and Y. Wang, "A Review of Graph Neural Networks and Their Applications in Power Systems," *Journal of Modern Power Systems and Clean Energy,* vol. 10, no. 2, pp. 345-360, 2022.

[14] S. Gajare, "Data Science Methodology and Approach," geeksforgeeks, 26 04 2022. [Online]. Available: https://www.geeksforgeeks.org/data-science-methodology-and-approach/. [Accessed 25 12 2022].

[15] S. Agrawal, "How to split data into three sets (train, validation, and test) And why?," Towards Data Science, 17 05 2021. [Online]. Available: https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c. [Accessed 03 01 2022].

[16] toppr, "Linear Interpolation Formula," [Online]. Available: https://www.toppr.com/guides/maths-formulas/linear-interpolation-formula/. [Accessed 25 12 2022].

[17] P.-L. Bescond, "Cyclical features encoding, it's about time!," Towards Data Science, 08 06 2020. [Online]. Available: https://towardsdatascience.com/cyclical-features-encoding-its-about-time-ce23581845ca. [Accessed 25 12 2022].

[18] The engineering toolbox, "Air - Density, Specific Weight and Thermal Expansion Coefficient vs. Temperature and Pressure," 2003. [Online]. Available: https://www.engineeringtoolbox.com/air-density-specific-weight-d_600.html?vA=20&units=C#. [Accessed 25 12 2022].

[19] I. Kuznetsov, “What is the wind speed and how do we measure it,” windy.app, [Online]. Available: https://windy.app/blog/what-is-the-wind-speed.html. [Accessed 26 12 2022].

[20] L. Hartman, “Wind Turbines: the Bigger, the Better,” Office of Energy Efficiency & Renewable Energy, 16 08 2022. [Online]. Available: https://www.energy.gov/eere/articles/wind-turbines-bigger-better. [Accessed 26 12 2022].

[21] Wind data, “Die Website für Windenergie-Daten der Schweiz,” [Online]. Available: https://wind-data.ch/tools/profile.php?h=10&v=5&z0=100&abfrage=Refresh. [Accessed 26 12 2022].

[22] Z. Jaadi, “A Step-by-Step Explanation of Principal Component Analysis (PCA),” built in, 26 09 2022. [Online]. Available: https://builtin.com/data-science/step-step-explanation-principal-component-analysis. [Accessed 30 12 2022].

[23] R. Ozminkowski, “Garbage In, Garbage Out,” Towards Data Science, 13 11 2021. [Online]. Available: https://towardsdatascience.com/garbage-in-garbage-out-721b5b299bc1. [Accessed 06 01 2022].

[24] E. W. Weisstein, “Normal Distribution,” Wolfram Mathworld, [Online]. Available: https://mathworld.wolfram.com/NormalDistribution.html. [Accessed 06 01 2023].

[25] C. Bento, “Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis,” Towards Data Science, 21 09 2021. [Online]. Available: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141. [Accessed 06 01 2023].

[26] T. Masui, “Graph Neural Networks with PyG on Node Classification, Link Prediction, and Anomaly Detection,” Towards Data Science, 06 10 2022. [Online]. Available: https://towardsdatascience.com/graph-neural-networks-with-pyg-on-node-classification-link-prediction-and-anomaly-detection-14aa38fe1275. [Accessed 06 01 2021].

[27] W. Liao, B. Bak-Jensen, J. R. Pillai, Y. Wang and Y. Wang, “A Review of Graph Neural Networks and Their Applications in Power Systems,” *Journal of Modern Power Systems and Clean Energy,* vol. 10, no. 2, pp. 345-360, 2022.

[28] Wolfram Mathworld, “Fibonacci Number,” Wolfram, 13 12 2022. [Online]. Available: https://mathworld.wolfram.com/FibonacciNumber.html. [Accessed 03 01 2023].

[29] J. Moody, “What does RMSE really mean?,” Towards Data Science, 05 09 2019. [Online]. Available: https://towardsdatascience.com/what-does-rmse-really-mean-806b65f2e48e. [Accessed 03 01 2023].