UNIVERSITY OF COPENHAGEN
FACULTY OF SCIENCE

MSc Thesis

Author: Georgios Garidis

Offshore wind power modeling and forecasting with very high-resolution weather forecasts

Supervisors: Pierre Pinson, Troels C. Petersen

Handed in: October 15, 2021

1 Abstract

Wind power forecasting gives insight into the future and allows for optimal decision making. So far, coarse Numerical Weather Prediction forecasts (e.g. ECMWF) have been used to predict what the energy production of a wind farm could be in the future e.g. the coming day or week. Recent advances in computational power and efficiency allow for even higher temporal and spatial resolution, more localized and physics rich weather forecasts (e.g. Whiffle's GRASP model).

The purpose of this work is to compare the ability of the GRASP and ECMWF forecast datasets, in predicting the total day-ahead power production at Anholt wind farm and find out where or how one is superior to the other. In the process, two different machine learning setups of predicting the total power will be constructed that are also to be compared. A general model taking in all information at once and a set of more specialized models whose results are to be combined each time in the end. In total 4 comparisons will be made, two setups each one testing two different datasets. Lastly an attempt to improve the best performing setup-forecast dataset combination will be undertaken.

The specialized setup using the dataset provided by the GRASP forecasting model performed the best among all setup-forecast dataset combinations with 16.86 % RMSE. It had $\approx$0.8 % difference with the general setup using the same forecast dataset and $\approx$ 1.7 % difference with the other specialized model using the ECMWF forecast dataset. Results suggest that the GRASP forecast dataset is superior at predicting the wind speed at Anholt wind farm which is the most important feature.

For improving the best performing model i.e. the specialized model using GRASP forecast data, a selection of the best features was performed and after that a linear neural network was added that would combine the results in a weighted manner. The overall RMSE predicting performance of the model is now 16.75 % which is an improvement of 0.11%.

2 *Acknowledgements*

Throughout the writing of this thesis I have received a great deal of support and assistance.

I would like to thank my external supervisor, Professor Pierre Pinson. His research inspired me to work with him on this project in the first place and his insight was invaluable throughout our collaboration. I am grateful to him for pushing me to constantly discover new methods but also allowing me explore on my own and trust my judgement. His feedback and guidance have helped me refine my way of thinking and I am grateful that I had the chance to work with him.

I would also like to thank my primary supervisor, Lector Troels Christian Petersen. I enjoyed our meetings to the fullest as they helped me understand the details of the problem better and sparked inspiring scientific conversations in the process.

I also extend my gratitude to Remco Verzijlbergh and Gerrit Deen at Whiffle. They provided us with their simulations without which, this work would not be possible.

Finally, I would like to extend my gratitude to my family and friends, for always encouraging me to go after my dreams and making sure to equip me with the strength to do so. From the bottom of my heart, I thank them for their love and support.

Contents

1 *Abstract* 1

2 *Acknowledgements* 3

3 *Introduction* 7

4 *Theory* 13

5 *Data Preprocessing* 25

6 *Methods* 35

7 *Results and Discussion* 47

8 *Bibliography* 67

3 Introduction

3.1 From wind gusts to wind forecasts

3.1.1 A brief history of harnessing the wind

Humanity throughout history has sought to built tools that would result in more fruitful work and better chances of survival. It was not long after, that it sought to harness the elements of nature. People realised the potential of using energy provided by nature to carry out labor-intensive work. Wind, being an omnipresent force in nature, was used in many ways since the ancient times.

People discovered that the wind exerted a big force on a big piece of cloth which could be used to propel ships or rafts, that would otherwise require human effort, thus giving birth to seafaring. Another example are windmills, which were used to grind grain, to pump water, and to cut wood as sawmills. Wind mills are the predecessors of modern day wind turbines, to which they are very similar conceptually and structurally.

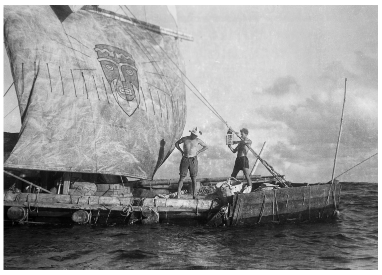


Figure 3.1: The Kon-Tiki expedition. A 1947 attempt to prove how early South-American people could have settled the Polynesian islands with simple naval means such as this raft. *Used with permission from the Kon-Tiki museum.*

3.1.2 The rise of wind power

During the last few decades, global energy demand has been growing exponentially with no apparent slowdown.[1] Figure 3.3 shows the global energy production by source from the year 1800 to 2019. So far, our needs have been mostly covered by conventional energy sources such as coal, oil and gas with devastating consequences to the climate and the environment. Wind stands out as one of the most promising forms of renewable energy, with goverments investing heavily in order to meet climate goals for carbon neutrality, have access to limitless, free energy and create a more sustainable future for their citizens. The increasing demand has created need for constant innovation in order to make the transition more economically appealing rather than a mere necessity. This thesis is a small step in that direction. Our objective is to improve the power prediction at Anholt wind farm operated by Ørsted. More on that in the chapter Research questions and Thesis Objective.

Figure 3.2: Iconic windmills in Mykonos. *Image credited to [Diakiw, 2008] and used with his permission.*

[1] Hannah Ritchie and Max Roser. Energy. *Our World in Data*, 2020. https://ourworldindata.org/energy

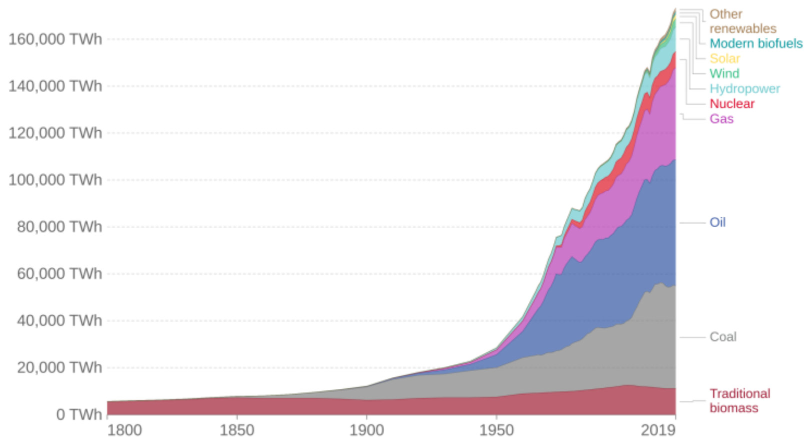


Figure 3.3: Energy production by source. After 1950s there was a spike in energy production. We can see that renewables make up only a small percentage of the overall energy production. *Image is free to use and credited to [Ritchie and Roser, 2020]*

3.1.3 *Ways wind energy can be extracted*

There are various ways that one can harvest energy from the wind. Several startups are coming up with interesting ways to make this possible. Each one of them has their advantages and disadvantages. Some more imaginative concepts include the vortex blade-less wind generator shown in Figure 3.4. This device uses a principle called Vortex Induced Vibration [2]. The shape and length of the mast is such that its resonant frequency coincides with that of the average wind intensity. The mast vibrates and the vibration is converted to electricity.[Yáñez, 2018]. One other interesting design is that of the ion wind generator. The concept is that a high voltage is applied to wires exposed to the air. These wires ionize the air or another substance e.g. water and create charged particles. Some of the charged particles are absorbed by the wires and some get blown away by the wind. The amount of particles that are pushed away result in the amount of electricity that is produced.[3] These two methods offer an alternative way to extract energy from the wind. Such methods, along with alternatives are researched due to their low cost of production and maintenance. However they are nowhere near the current power output capability that conventional wind turbines have.

Conventional wind turbines use some sort of blade or fin that will be pushed by the wind resulting in circular, continuous motion. That motion is then transferred via a gear system to the generator. There is usually a very large permanent magnet to which the motion will be transferred. That magnet is encased in coils and the alternating magnetic field through the coils is producing electricity through electromagnetic induction.[4] The components of the wind turbine's nacelle are illustrated in Figure 3.5. The power output ranges from just some hundreds of kW all the way to 10 MW for a single turbine. Nominal power output is proportional to the size of the wind turbine and its swept area. During operation the power output is a function of the incoming wind speed. The relation between the wind speed and the power output is called the power curve. Figure 3.6 shows a typical power curve. We can see that the operational range starts from 3 m/s up to 25 m/s. Below 3 m/s wind speed is not sufficient for

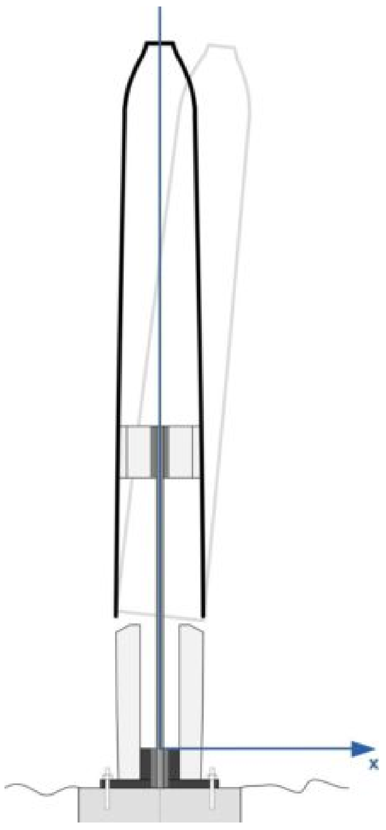


Figure 3.4: Vortex Bladeless Wind Generator. *Image courtesy of Vortex Bladeless SL.*

[2] David J. Yáñez. Viv resonant wind generators. 06 2018

[3] D. Djairam, A.N. Hubacz, P.H.F. Morshuis, J.C.M. Marijnisen, and J.J. Smit. The development of an electrostatic wind energy converter (ewicon). In *2005 International Conference on Future Power Systems*, pages 4 pp.–4, 2005. DOI: 10.1109/FPS.2005.204208

[4] Henk Polinder and Maxime Dubois. Generator systems for wind turbines. *Proceeding PCIM'03*, (May), 2003. URL `https://www.researchgate.net/publication/262643332`

the turbine to start producing energy and above 25 m/s the turbine stops for safety reasons. Each turbine has a specific power curve that is designed by the manufacturer.

Wind turbines are by far the most advanced, studied and scalable method of converting wind energy to electric power. The most common way to scale production is to place multiple wind turbines in a wind farm configuration. That comes with additional challenges of course. Wind farms require vast areas to install, come with a high manufacturing cost and can pose a threat to the surrounding ecosystem. They are however one of the most promising ways that humanity can tap into the planet's renewable energy sources.

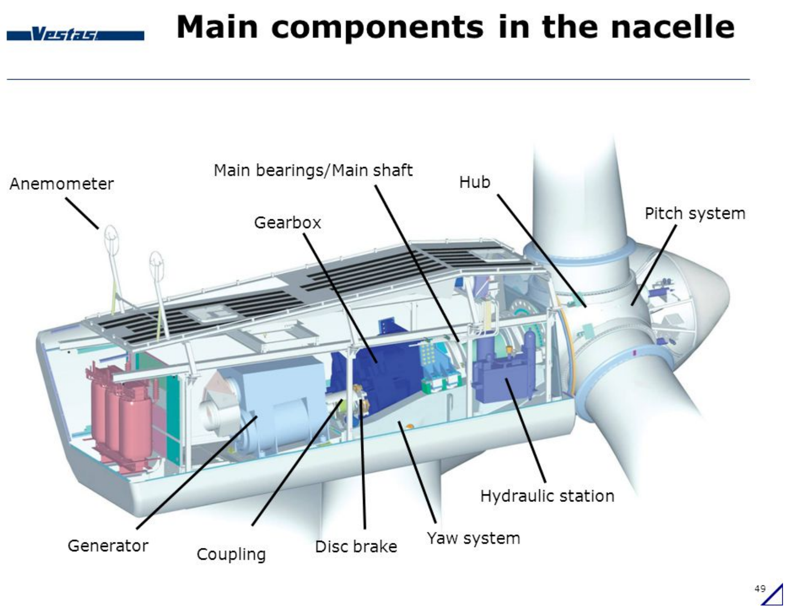


Figure 3.5: Diagram illustrating components in the nacelle's interior. Image *Courtesy of Vestas Wind Systems A/S*.

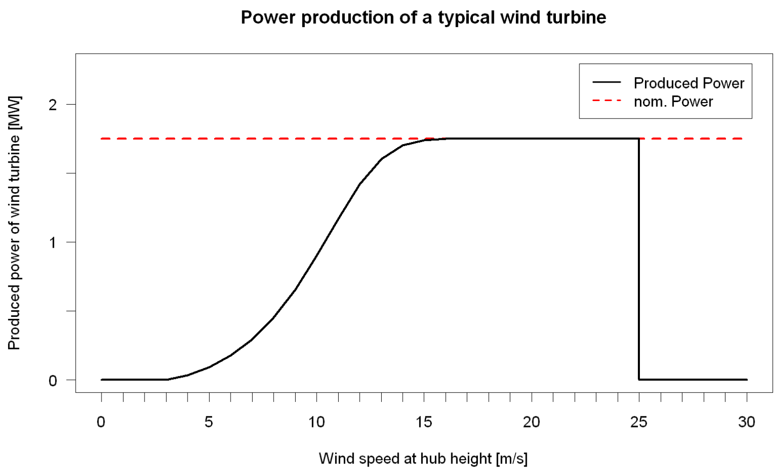


Figure 3.6: A typical power curve. It shows the conversion relation between wind speed and wind power. The curve usually looks like a sigmoid function in order to reach and continue working at nominal power capacity from low wind intensities. [Isjc99, 2012]

3.1.4 *Wind power and weather forecasting*

The fact that wind is a random phenomenon we cannot control, introduces some problems. The energy produced by sources such as coal, gas and nuclear can be increased or lowered on demand. Wind on the other hand is a stochastic process. When there is no wind, the deficit in energy is covered by conventional energy sources. Apart from this, energy providers have to provide day-ahead estimate of the power they will produce, which are then placed in the energy market and in return influence how electricity will be distributed between producers and consumers. Producers have a loss in the energy market both when they overproduce or underproduce energy in respect to their bid. It is also important for them to know when they can service their wind turbines with minimum loss and maximum safety i.e. a day when the wind is not blowing. It is immediately clear that it is important to know beforehand what the weather conditions will be at least in the short term and then estimate the power production as accurately as possible for the wind farm. A slight improvement in the power production prediction could have notable economic benefits which makes this subject worth the research.

So far the state of the art approach was to use forecasts from simulations based on Numerical Weather Prediction that are offered at low temporal and spatial resolutions due to computational costs and run-time. These predictions are called deterministic.[5] The results from these coarse weather forecasts are used in some statistical or Machine Learning algorithms to predict the power production of the whole wind farm.[6] One organisation producing such forecasts is the European Center for Middle-Range Weather Forecasts, ECMWF for short. There are more similar centers but this is the one providing us with our data.

[5] Probabilistic forecasts are also very promising, often performing better than deterministic since their results also come with an uncertainty in the prediction. However they are more difficult to interpret and integrate into existing institutional and industrial applications.

[6] Conor Sweeney, Ricardo J. Bessa, Jethro Browell, and Pierre Pinson. The future of forecasting for renewable energy, 2020. ISSN 2041840X

With the increase in efficiency and computing power more physics-rich, high resolution simulations than those of ECMWF have emerged. One Dutch company named Whiffle is producing such simulations. Their forecasts use the standard ECMWF forecasts as boundary conditions for an additional simulation, that takes into account small scale physical phenomena and is able to run at higher temporal and spatial resolutions thus improving the forecast data quality. The result is more specific and localized forecasts.

3.2 *Thesis Objective and Research Questions*

This section is devoted to making sure that the objectives of this thesis are clear to the reader from the very beginning. It begins with a summary of the concepts that the reader needs to understand. Then, a clear problem statement is formulated and the research questions that want to be answered along the way are phrased.

3.2.1 *Summary*

The simple idea, on which this work is based, is that more information should result in better forecasts. A higher resolution forecasting model, with additional physical processes taken into account should do better than a lower resolution, simplified model.[7]

[7] ECMWF is referred to as a simplified model only because of its resolution and scale of operation compared to Anholt wind farm. It runs simulations on much larger scales therefore omitting small scale phenomena that might be important at the wind farm level.

We have at our disposal 3 datasets. The first is the Anholt wind farm dataset containing all the measurements from the wind farm. The low resolution forecast data is provided by ECMWF and the high resolution forecast data by Whiffle which will be referred to as GRASP since this is how they call their model.

3.2.2 *Objectives*

The goal set in this thesis is simple. We would like to examine if the use of GRASP as the forecast model, provides an advantage over the use of the ECMWF forecast model in the day-ahead forecasting ability of Anholt wind farm. This has been done before at Horns Rev I wind farm off the coast of Denmark and successfully made the point, that a high resolution LES model such as GRASP can be advantageous over coarser NWP models like ECMWF for this purpose.[8] The goal is to examine if the same holds for Anholt wind farm so that the same hypothesis can be tested. This work is experimental and since nothing will be proved analytically, the result could only be considered a partial proof of concept.

[8] Ciaran Gilbert, Jakob W. Messner, Pierre Pinson, Pierre Julien Trombe, Remco Verzijlbergh, Pim van Dorp, and Harmen Jonker. Statistical post-processing of turbulence-resolving weather forecasts for offshore wind power forecasting. *Wind Energy*, 23(4): 884–897, 2020. ISSN 10991824. DOI: 10.1002/we.2456

The approach to this is to implement a suitable Machine Learning algorithm that will use the forecast data and predict in an optimal way the day-ahead power production for Anholt wind farm. The same model will be applied to both ECWMF and GRASP forecast data so that their predictive capability is shown to be comparable. The actual data provided by Ørsted would be used to evaluate how well each model did eventually.

3.2.3 *Research questions*

During this work a lot of approaches were tested such as different Machine Learning methods and models, feature engineering, hyper-parameter optimization, feature analysis, sliding window techniques etc. It was essential to experiment and optimize the learning models to see how well they could perform. When that would have been achieved there were two basic research questions that we would like to answer.
The first one is about whether it is better to use an aggregate model or individual models whose results are afterwards summed to obtain the aggregate production.[9] In an aggregate setup, you feed the model with all the features for all the turbines and train the model to produce the total power output of the wind farm at each point in time.[10] The individual approach is to have a separate model for each turbine at each point in time. This means that for predicting the power at one moment we need to run 111 different models, each

[9] What we are interested in is the aggregate power production of the whole wind farm which is what would be used in the day ahead power market.

[10] Each individual turbine could have e.g. wind speed and wind direction as features. Having 111 turbines means that we end up with 222 features if we keep it down to 2 features per turbine. The input amount to the model in this approach scales rapidly.

one having just the features that are relevant for each turbine.[11] The question therefore comes down to: *Is it better to use a model with all the information or divide it to smaller models? Does more information result in better performance?*
The second research question is more abstract and leaves more space for exploration and experimentation: *Is GRASP forecast data(LES model) better at predicting the energy than the ECMWF forecast data(NWP model)? Where does it perform better? Can we find the reasons why?* We want to see how the models compare and if there is a clear winner in the comparison. If not where is each model superior to the other? These questions will be answered in the Results chapter.

[11] In this case one would run 111 models, each one given e.g. wind speed and wind direction as features. These 111 individual results will be summed to obtain the aggregate power output that can again be used in the energy market.

3.3 *Structure of the thesis*

This thesis was written using the Tufte-book design, which is named after its creator Edward Tufte. It consists of a main text column and a marginal column. The main column contains the core of this work, the results and the details. The marginal content consist of sidenotes, citations and marginal figures. Side-notes help by giving some explanatory thoughts or details on something mentioned in the main text, citations are shown fully in order to give the reader a better indication of the work that is cited and marginal figures are figures that would not qualify for a place in the main text but can still prove informational to the reader. This next section will explain to the reader the contents of this work and how it has been structured.

The **Introduction** is providing the reader with a basic history of how wind power came to be, how it is extracted and how forecasting can help to harness it more effectively. Section **Thesis Objective and Research questions** states clearly what the goal of this thesis is and the questions we would like to answer along the way.

The **Theory** chapter introduces the theoretical models and computational tools that were used to carry out the analysis.

The **Data Preprocessing** chapter analyses the 3 datasets that were used. The acquisition and production of the data is explained and a small analysis of the Anholt wind farm dataset is made.

The **Methods** chapter goes into more detail about how I implemented the algorithms that were used, how some problems were overcome and how the process was optimized.

The **Results and Discussion** chapter contains the results of the goals that were initially set, answers to the research questions and what could be improved in future experiments.

4 Theory

This chapter will provide the theory on various methods and techniques that were used for obtaining the results or performing some type of analysis throughout the thesis.

4.1 Machine Learning

4.1.1 The learning problem

Machine Learning has become a very broad field with new models and techniques emerging every day. There are different types of Machine Learning such as Supervised, Unsupervised and Reinforcement Learning. This thesis will focus on Supervised Learning where the aim of the algorithm is to be given some input and output and to be able to learn the relation between them. The input data and outcome are connected, usually through a high dimensional target function which is unknown or difficult to define analytically. A Machine Learning algorithm basically tries to find a hypothesis that best approximates that target function.

4.1.2 Training, validation and test set

Machine learning algorithms can become very good at learning the specific relationships between data and output, provided that the algorithm is complex enough to be able to do that. When training on a set of data, it is wrong to try to evaluate the algorithm on the same set because the result will be much better compared to when the algorithm will be given data it has never seen before. That happens because the algorithm can learn each specific case better than it should, failing in this way to generalize its learning capability to new data. The solution to this is to split the initial dataset into 3 parts. The train, validation and test set. During this process it is important to make sure that the all sets have equal amounts of the different examples[1] the algorithm will want to learn. Thus, the 3 sets should be sampled randomly if that is possible.[2]

[1] e.g. in a classification problem if the ratio of True to False is 70 - 30 % then each set should have the same ration of examples.

[2] Some problems do not allow for that kind of sampling because they need to preserve their order e.g. time dependent problems, such as ours.

The training set will be used to train the algorithm. It usually consists of around 50 to 80 percent of the dataset, depending the amount of the data originally. The validation set is 10 to 25 percent of the initial set and is used to optimize the algorithm's parameters, avoid overfitting and give a measure of how well the algorithm is doing during training. The test set is usually as big as the validation set so that it has roughly the same level of different cases as the validation set. It is used after all the optimization has been done in order to check how well the algorithm actually performs on data that it has never seen before.

Although the algorithm does not explicitly train on the validation set, it is familiar with it. That is because the validation set has been used to optimize the learning process. That means, it is used as an evaluation set during training, during which it informs us on which parameters will improve the learning process and at which point the algorithm generalizes best. Basically we have optimized our algorithm by trying to increase its predictive ability on the validation set. Therefore, it is logical that the algorithm will do better on the validation set than on the test set.

4.1.3 *Loss function*

The algorithm can come up with any kind of hypothesis to match the target function if it does not have a measure of how well it actually performs. In order to give the learning algorithm a measure of the quality of estimation, a loss function is introduced. An example of a loss function, Mean Square Error in this case, would look something like this:

$$J(h_\theta, x_i) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x_i) - y_i\right)^2 \tag{4.1}$$

where x_i is the input data, h_θ is the hypothesis with parameters θ being tested, y_i is the true value,[3] m is the number of data points and J the loss function. It has to be noted that the loss function is specific to each problem. In a regression problem the loss function could be the Root Mean Square Error(RMSE) or the Mean Square Error(MSE) function as in this example. $RMSE = \sqrt{MSE}$ and the reason for that is because the units in the RMSE are the same with the quantity evaluated. The MSE is given in quantity units squared and a lot of the times a physical quantity like e.g. $power^2$ is not intuitive. Another very classic loss function is the Mean Absolute Error(MAE) described by the following equation:

[3] The target function is unknown so the notation $f(x_i)$ is usually replaced with y_i. y denotes the true value in the data while $\hat{y}$ represents an estimation through a model. The expression $h_\theta(x_i)$ could therefore be changed to $\hat{y}$.

$$MAE = \frac{1}{m}\sum_{i=1}^{m}\left(|\hat{y}_i| - |y_i|\right) \tag{4.2}$$

The difference between the MSE loss function of equation 4.1 and the MAE loss function of equation 4.2 and the reason why to chose one over the other resides in the way they penalize the difference between prediction and real value. The MAE is linear and penalizes large and small differences with linear austerity.[4] MSE or RMSE on the other hand square the differences, creating very large losses for larger differences therefore prioritizing the algorithm to learn those examples first. In a problem where large differences are more important than small ones, the choice of the loss function would be the MSE. In a problem where the result is a physical quantity it

[4] Jakob W. Messner, Pierre Pinson, Jethro Browell, Mathias B. Bjerregård, and Irene Schicker. Evaluation of wind power forecasts—An up-to-date view. *Wind Energy*, 23(6):1461–1481, 2020. ISSN 10991824. DOI: 10.1002/we.2497

makes more sense to use the RMSE instead of the MSE to preserve the original units of the quantity in the result.[5]

[5] That is what we have done in our analysis as well.

In a classification problem the loss function could be the Negative Log Likelihood or the Binary Cross Entropy function. Those come with their differences as well and therefore the choice of loss function really depends to each problem and also on the user's preference.

4.1.4 Learning models

The simplest case of a learning model is a linear model. A linear model is the following:

$$y = \sum_{i=1}^{n} w_i x_i + b = \mathbf{w}^T \mathbf{x} \tag{4.3}$$

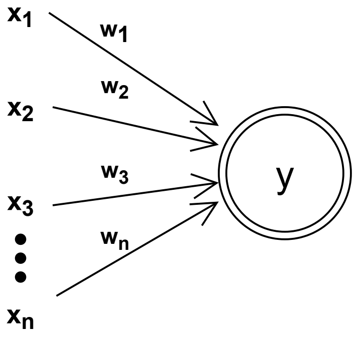


Figure 4.1: This is a simple linear model. The **x** vector represents the input features to the model and the **w** vector the weights. *This figure was created with diagrams.net*

in which **x** is a feature vector and y is the predicted value. The bias b can be included in the vector if $x_0 = 1$ and $w_0 = b$. The algorithm has to optimize these weights so that they are able to reproduce the underlying target function as best as possible. A visual representation of a simple linear model can be seen in Figure 4.1. Instead of just stopping the model after 1 layer of neurons[6] we could also add more layers between the input and output layers, called hidden layers, to create a deep neural network. A representation of a deep neural network can be seen in Figure 4.2. In principle each node in the input layer is a linear model. A node in the next layer works in the same principle. Each input is already a linear model, so each node in the hidden layer is described by equation 4.4. This is the equation that would describe just the first node in the hidden layer.

[6] We refer as neurons to the arrows since they behave and were modelled after biological neurons.

$$H_1 = \sum_{i=1}^{S} z_i y_i + b_{H_1} = \sum_{i=1}^{S} z_i (\sum_{j=1}^{n} w_j x_j + b_{y_i j}) + b_{H_1} \tag{4.4}$$

In equation 4.4, H_1 denotes the output of the first node in the hidden layer, S denotes the number of nodes in the input layer, z_i denotes the weights of each node in the input layer towards the first node in the hidden layer and b_{H1} denotes the bias of the first node in the hidden layer. By introducing additional layers, the model becomes non linear. After the weights are multiplied we end up with a much more complex model that is able to learn more complex relations between the input and output. The deeper the network, the more parameters there are to optimize, the better its learning capacity.

4.1.5 Gradient descent

Gradient descent is an essential part of Machine Learning. It is the part that makes the algorithm improve and learn the underlying target function better. Gradient descent refers to the method by which a function can be minimized. In Machine Learning we use gradient descend to find the set of parameters that minimizes the loss function, thus improving the algorithm. To find the local minimum of a function using gradient descent, we must take steps proportional

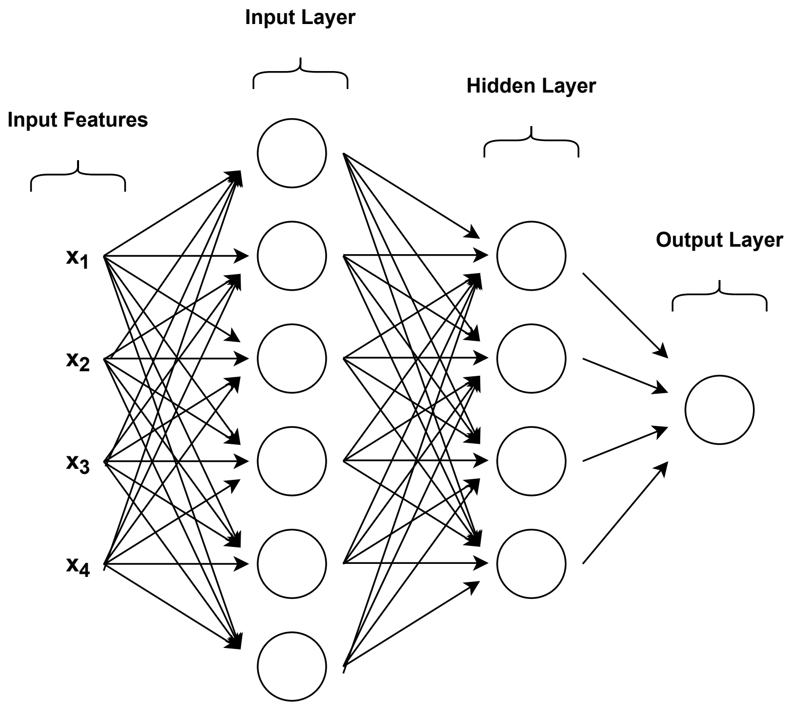


Figure 4.2: This is a representation of a deep neural network. It is visible that the complexity of the algorithm increases very fast with the number of nodes and layers. *This figure was created with diagrams.net*

to the negative of the gradient (move away from the gradient) of the function at the current point. If we take steps proportional to the positive of the gradient (moving towards the gradient), we move towards a local maximum. In Figure 4.3 one can see the gradient descent in a function with only 1 parameter.

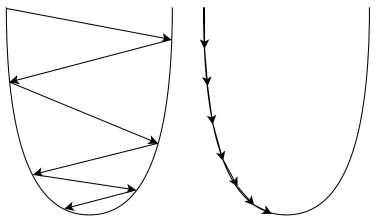


Figure 4.3: Gradient descent in a 1 dimensional, parabolic loss function. A large descent(learning) rate has been used in the left optimization and a smaller in the right one. There are advantages and disadvantages to both cases and usually a dynamic descent rate is used in Machine Learning. *This figure was created with diagrams.net*

In the case of a simple linear network such as that in Figure 4.1, the parameters to be optimized are the components of the **w** vector as in equation 4.3. If train and validation set are the same, the loss function can only give a different result when the parameters of the model change. In this case the loss function in equation 4.1 becomes:

$$J(h_{\mathbf{w}}) = \frac{1}{m}\sum_{i=1}^{m}\left(h_{\mathbf{w}}(x_i) - \hat{y}_i\right)^2 \tag{4.5}$$

In the example of our linear model the **w** vector has n components that need to be optimized. This results in an n-Dimensional loss function. A prerequisite in gradient descent is that the function that is to be minimized is differentiable. Since we do not have an analytical form of the hypothesis function to differentiate, this is done numerically. The following equation shows how the **w** vector should be updated iteratively until the algorithm performs at a satisfactory level.

$$\mathbf{w}_i := \mathbf{w}_{i-t} - \gamma\nabla J(\mathbf{w}_{i-1}) \tag{4.6}$$

The **w** vector denotes the neuron weights to be optimized, ∇ is the del operator and γ is the descent or learning rate. The algorithm checks how each weight affects the loss function individually. By calculating the partial derivative in this way, the algorithm can find the direction to which the multidimensional vector **w** should move to improve the algorithm. In a deeper neural network model the amount of neurons to be optimized can become very high, resulting in very high dimensional loss functions whose minima are computationally costly to reach.

4.2 *Overfitting*

Overfitting is a phenomenon usually in supervised training and it happens when the Machine Learning model becomes too good at recognizing the data that is has trained on, therefore failing to generalize to the new data. It usually happens when complex models are trained on simple data. The models are naturally trying to find more evidence to connect the data to the outcome. Usually they do that by learning the noise in the data. Since noise is random, learning it on the train data does not mean it will help predicting the test data.

An important factor influencing the risk of overfitting is the number of data points. A small dataset is more likely to force the algorithm to learn from low or negative impact features. A large dataset is therefore more robust to overfitting. There are numerous way that overfitting is prevented and monitored. Two common ways to prevent overfitting are using a validation set for early stopping and cross validation.

The validation set, as was mentioned before, is like a test set for decision making during the learning process. In this case it is used during training to monitor when the model's performance starts to decrease. That point might be before the pre-decided training rounds and so it is decided to terminate the training at the point during training where the validation loss was lowest. This technique is called early stopping and it is one of the most effective measures against overfitting in all Machine Learning models regardless of structure or function since it simply pulls the plug on the training of the model.

Cross validation does not actively prevent overfitting. It is used mostly to decide if the evaluation of the algorithm is reliable. A cross validation of k folds in this case would split the dataset in k splits and would train and evaluate the model k times. Each fold would use a different split as the test set and the rest k-1 splits are used as training and validation sets. The final performance of the model is the average performance of the models. The certainty of the prediction increases with the number of the folds but so does the computational time since the model trains k times to create and average evaluation.

4.2.1 *Hyperparameter optimization*

Each machine learning algorithm has a set of parameters that are optimized while the algorithm is running. However there are additional parameters that need to be tuned, called hyperparameters. These are the parameters that are defined upon the creation of each model. Each machine learning model has different types of hyperparameters, since they can differ very much in the purpose of use and operation. Some common hyperparameters for a deep neural network, such as the one analyzed above, can be the *number of hidden layers*, *number of nodes in each hidden layer*, the *learning rate* γ, the *type of loss function*, the *dropout rate*, *momentum*, *number of epochs* or *batch size*. The list is inexhaustible.

We end up again with a multidimensional space in which we need to find the set of parameters that produce the best results i.e. minimizes the loss funcion again with respect to the hyperparameters. Using gradient descent in this case is impossible since calculating the gradient for each parameter would require to run the model at least twice. The most popular methods to search for hyperparameters are by performing either a grid or random search or a Bayesian optimization.

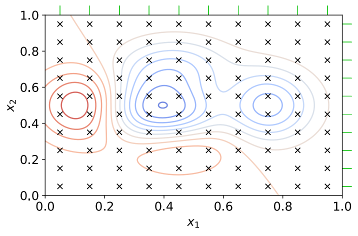


Figure 4.4: 2 dimensional grid search hyperparameter optimization. Image credited to [Elvers, 2019].

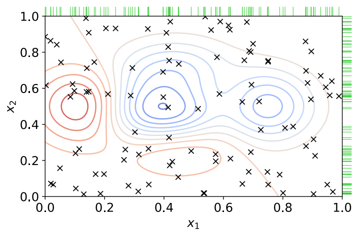


Figure 4.5: 2 dimensional random search hyperparameter optimization. Image credited to [Elvers, 2019].

The first one splits the range of values in each dimension to a desired amount and then searches across the hyperparameter space in a gridded configuration so that the whole space is covered with equal probability. The second picks random combinations of hyperparameters. It can outperform Grid search, especially when only a small number of hyperparameters affects the final performance of the machine learning algorithm.[7] Both of these algorithms are embarrassingly parallel, meaning that the individual searches are completely independent and can be run in parallel. Last one is the Bayesian optimization, a probabilistic model that gathers information from previous iterations, through which it reveals information about the function and location of the optimum. It tries to balance the cost of exploration of hyperparameters for which the outcome is most uncertain with exploiting the known values that are expected to be close to the optimum. Bayesian optimization can provide better results in fewer iterations than grid or random search.[8] It is important to note that hyperparameters can be tuned in groups that are relevant or affect each other, reducing this way the dimensionality of the problem.

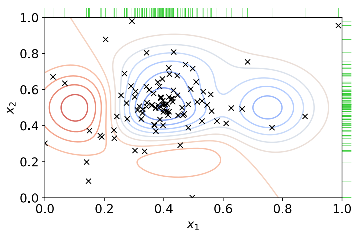


Figure 4.6: 2 dimensional Bayesian hyperparameter optimization. Image credited to [Elvers, 2019].

[7] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012. ISSN 15324435. URL `http://scikit-learn.sourceforge.net`.

[8] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-Parameter Optimization. 2011

4.2.2 *Decision trees*

The decision tree is a supervised learning algorithm. It works both in regression and classification problems and is able to learn very complex, non linear relations but can also overfit the data relatively easily if not optimized. They consist of two kinds of elements: branches and nodes. In the following fictional survey 150 people were asked at noon what time they had breakfast and if they were hungry. A made up example of a decision tree with a depth of two nodes will be built to try classify the relation between breakfast time and hunger.

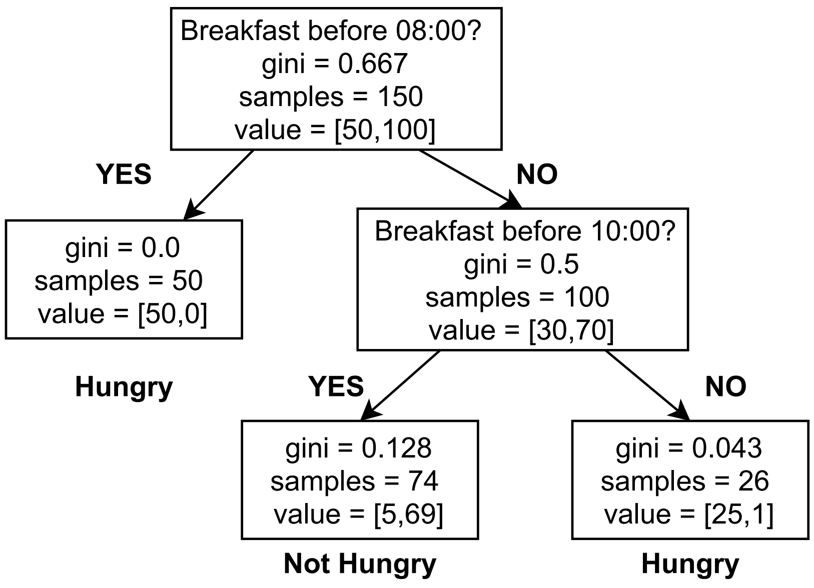


Figure 4.7: Made up tree example for classifying when people become hungry depending on when they had breakfast. *This figure was created with diagrams.net*

The first node is called the root node of the tree. The tree finds at each node, the feature that best splits the data into correct categories. The Gini categorical metric[9] in this example gives the tree a measure of which feature is the most capable of making the best split and which value should be the threshold for that split. In this case the only feature we have is the time the participants had breakfast. The terminal nodes specifying the participants as hungry or not hungry are called leaf nodes. If we wanted a better approximation we could allow the tree to split further, until it would create a leaf node for every individual case, thus creating an overfitting case on the data and reducing the ability of the tree to generalize. Ways to avoid that can be by setting a maximum depth for the tree, the total number of leaves or the minimum number of data points belonging in a leaf node.

[9] Loss function equivalent for decision trees. There exist more classification and regression loss functions for trees. In this case, the lower the Gini metric, the better the split in the node.

4.2.3 *Ensemble methods*

With ensemble methods one can increase the performance of decision trees. Bagging and gradient boosting are the two main types of ensemble learning methods.

Bagging or bootstrap aggregating relies on the idea that a collection of weaker learners is more efficient than a very specialized, strong learner. The algorithm creates subsets of the original data with replacement and for each of those subsets a decision tree is trained. The results of the trees are then averaged. The bootstrapping procedure results in better performance because it decreases the variance of the model without increasing the bias.[10] That happens because even though a single tree might be sensitive to noise in the training set, a collection of trees is not. Random forests are another type of ensemble algorithms that improve on the idea of simple bagging. In bagging it is common that trees can become correlated if the training data is

[10] David Opitz and Richard Maclin. Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999. ISSN 10769757. DOI: 10.1613/jair.614

similar because the best feature is used to make the node splits. In random forests the features that can be used at each split are random, averting the algorithm to persist predicting only with some strong feature.

Boosting is another form of ensemble method. It iteratively combines weak learners[11] to build a strong learner that can predict more accurate outcomes. Each new tree in boosting is based on a modified version of the original dataset. Gradient boosting combines gradient descent with the boosting algorithm. The gradient boosting algorithm is best explained by exploring the AdaBoost. The AdaBoost Algorithm starts by training a decision tree with equal weights for each observation. Following the evaluation of the first tree, we increase the weights of the difficult-to-classify data and decrease the weights of the easy-to-classify observations. As a result, the second tree is based on the weighted data. The gradient descent part optimizes the weights of each of the weak learner in order to minimize the loss function.

[11] Weak learners can be a simple, not very deep tree or a stump i.e. a tree with just one node or split.

4.2.4 *Extreme Gradient Boosting*

Since AdaBoost, more advanced boosting algorithms have been created. The state of the art gradient boosting algorithms include Extreme Gradient Boosting(XGBoost), Light Gradient Boosting(LGB) and CatBoost. In this thesis XGBoost was mainly used to perform the forecasting.

XGBoost differs from regular gradient boosting by introducing some extra features. It uses either L1 or L2 regularization to prevent overfitting as well as sparsity awareness and weighted quantile sketch for approximate tree learning. Apart from mathematical improvements, it utilizes the system resources much more effectively making it a very fast algorithm compared to the original gradient boosting algorithms.

The most important hyperparameters to tune in XGBoost are *eta*, *min_child_weight*, *max_depth*, *max_leaf_nodes* and *gamma*. *Eta* is the equivalent of learning rate in Gradient Boosting Machine(GBM), *min_child_weight* is the minimum weight required to create a new node in the tree[12], *max_depth* is the maximum depth of a tree, *max_leaf_nodes* is the number of terminal nodes in a tree and *gamma* is the reduction threshold in the loss function above which the node is further split.

[12] Or number of samples if all samples have a weight of 1. Each new stump that is added, weights data samples differently if they were badly predicted by the previous stumps up to that point

4.3 *Kriging*

Kriging is a spatial interpolation method. It was initially used by Danie G. Krige to estimate the gold distribution in soil based on samples from a few boreholes. The theoretical basis for the method was formalized in 1962 by French engineer Georges Matheron. [13] I mainly used Kriging to interpolate missing values in the wind farm dataset.

[13] Georges Matheron. Traite de geostatistique appliquee. Tome I, 1962

Usually interpolation methods are deterministic. With Kriging one can associate some probability with one's predictions, which means that the values are not perfect or exact predictions such as from a deterministic statistical model. Kriging methods rely on autocorrelation. The concept is based on an axiom of geography. Things that are closer tend to be more similar or related than distant things. How quickly this correlation can decay is a function of distance.[14] Knowing the value of a variable at different locations allows you to compute all the distances between the observations and with that compute the auto-correlation as a function of distance. A plot containing this information is called a variogram.

[14] Ricardo A. Olea. Geostatistics for Engineers and Earth Scientists. *Technometrics*, 42(4), nov 2000. DOI: 10.1080/00401706.2000.10485748

A variogram gives the analyst some information on how the quantity being studied behaves. When wanting to calculate a variogram, the distances between known data points are called lags. Since not all distances will be the same, a single distance is chosen which is no smaller than the minimum distance between measurement points. Optimally it is close to the mean distance between measurements. Since the measurements are rarely equally spaced, lags have a range in which distances are grouped as the same lag. The width of this interval is called lag tolerance and is usually set to half of the distance between lag distance. A representation of a variogram can be seen in Figure 4.8. Each red point is the variance of a group of data points whose distance belongs in this interval. One can see how the variance of the values increases as the distance increases. This means that the values of points that are further spaced apart usually have a larger dispersion from the average, meaning that it is more likely not to be related.[Olea, 2000] After creating the variogram plot, the next step is usually to fit a function that models the behaviour of the variogram properly. In this case the relation is linear and is characterized by 3 variables. The range is the distance at which the data points are no longer considered correlated, the sill is the variance at this distance and the nugget is the intercept of the variogram slope with the y axis. The nugget exists because even if the distance between two points is infinitesimally small, there are still microscale variations and a slight deviation from the topical mean. Once these values have been calculated, one can estimate the dispersion of the values of that quantity at various distances.

There are many different models of Kriging. The one that was used in this thesis is called Ordinary Kriging. In Ordinary Kriging one assumes that the mean at a certain location is an unknown constant. This can be problematic at times, especially if the assumption is not logical. In our case, assuming that nearby turbines have a similar power output with their neighbouring turbine is logical, although there are complications and phenomena that could see a more complex model implemented. Ordinary Kriging assumes that at location x_0 the quantity Z is described by equation 4.7

$$Z(x_0) = \mu + \epsilon(x_0) \tag{4.7}$$

meaning that it is constant and only some topical fluctuation error exists that differentiates it from the neighbouring points. However

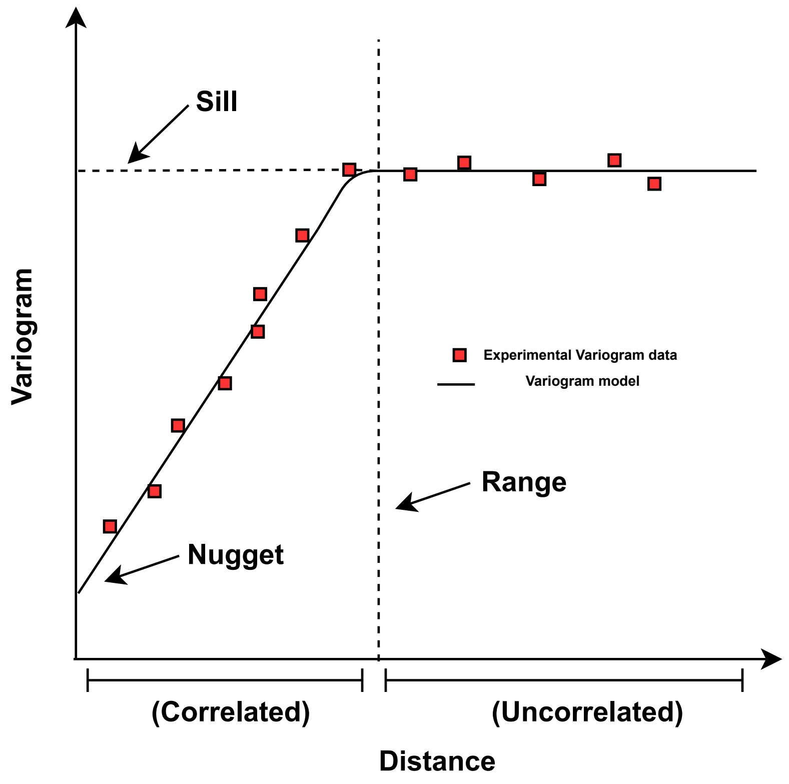


Figure 4.8: This is a representation of a variogram. The purpose is to model the behaviour of the variance with respect to the distance and extract the nugget, sill and range parameters after fitting the proper function type to the points.

the difference with other Kriging methods is that a constant μ is **only** assumed in the neighbourhood of x_0.

The error is defined in the following way:

$$\epsilon(x_0) = \hat{Z}(x_0) - Z\left(x_0\right) = \left[W^T - 1\right] \cdot \left[Z(x_1) \cdots Z(x_N) Z(x_0)\right]^T \quad (4.8)$$

which can be written in the form of a sum as

$$\epsilon\left(x_0\right) = \sum_{i=1}^{N} w_i\left(x_0\right) \times Z\left(x_i\right) - Z\left(x_0\right) \quad (4.9)$$

where $\hat{Z}\left(x_0\right)$ is the estimator value at x_0, $Z\left(x_0\right)$ is the actual value at x_0 hypothesized by equation 4.7 and W a vector with the weights for the values of the neighbouring turbines. The weights need to sum up to 1 and that leaves us with our first constraint.

An estimator can have a mean of errors equal to zero since in a large sample they can cancel each other out. The best estimator between two can therefore be the one with the lowest variance in its errors. Taking into account that the variance that $Var(cX) = c^2 Var(X)$, the variance of our estimator's errors at x_0 is

$$\text{Var}\left(\epsilon\left(x_0\right)\right) = \text{Var}\left(\left[\begin{array}{cc} W^T & -1 \end{array}\right]\left[\begin{array}{cccc} Z\left(x_1\right) & \cdots & Z\left(x_N\right) & Z\left(x_0\right) \end{array}\right]^T\right) \quad (4.10)$$

$$= \begin{bmatrix} W^T & -1 \end{bmatrix} \mathrm{Var}\left(\begin{bmatrix} Z(x_1) & \cdots & Z(x_N) & Z(x_0) \end{bmatrix}^T\right) \begin{bmatrix} W \\ -1 \end{bmatrix} \tag{4.11}$$

By calculating the variance of a vector we get its auto-covariance matrix i.e. a square matrix in which each entry is the covariance, calculated for every combination of components of the vector. This gives rise to the following form for the estimator's error variance:

$$\mathrm{Var}\left(\epsilon\left(x_0\right)\right) = \begin{bmatrix} W^T & -1 \end{bmatrix} \begin{bmatrix} \mathrm{Var}_{x_i} & \mathrm{Cov}_{x_i x_0} \\ \mathrm{Cov}^T_{x_i x_0} & \mathrm{Var}_{x_0} \end{bmatrix} \begin{bmatrix} W \\ -1 \end{bmatrix} \tag{4.12}$$

We are left with a system of equations that can be solved to estimate the value at Z_{x_0}. We need to minimize the following equation with the constraint that $1^T \cdot W = 1$

$$\mathrm{Var}\left(\epsilon\left(x_0\right)\right) = W^T \mathrm{Var}_{x_i} W - \mathrm{Cov}^T_{x_i x_0} W - W^T \mathrm{Cov}_{x_i x_0} + \mathrm{Var}_{x_0} \tag{4.13}$$

Solving this problem with the use of Lagrange Multipliers gives us the Kriging system.

$$\begin{bmatrix} \hat{W} \\ \mu \end{bmatrix} = \begin{bmatrix} \mathrm{Var}_{x_i} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathrm{Cov}_{x_i x_0} \\ 1 \end{bmatrix} \Rightarrow \tag{4.14}$$

$$\begin{bmatrix} \hat{W} \\ \mu \end{bmatrix} = \begin{bmatrix} \gamma(x_1, x_1) & \cdots & \gamma(x_1, x_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma(x_n, x_1) & \cdots & \gamma(x_n, x_n) & 1 \\ 1 & \cdots & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} \gamma(x_1, x^*) \\ \vdots \\ \gamma(x_n, x^*) \\ 1 \end{bmatrix} \tag{4.15}$$

The covariances represented with $\gamma(x_i, x_j$ have already been calculated for the variogram mentioned in case that was desired, and the additional parameter μ is introduced by the Lagrange multipliers to minimize the error in Krigging.

The method is explained in depth in the book Geostatistics for Engineers and Earth Scientists [15].

[15] Ricardo A. Olea. Geostatistics for Engineers and Earth Scientists. *Technometrics*, 42(4), nov 2000. DOI: 10.1080/00401706.2000.10485748

4.4 *Percentile bootstrap confidence intervals*

In statistics, a confidence interval is the range of values for an unknown parameter e.g. a population mean. The interval has an associated confidence level that is chosen during the analysis. Generally, a confidence interval is based on sampling the distribution of an estimator. The most common confidence levels are 90%, 95% and 99%. They represent the theoretical frequency at which values from the unknown distribution would fall in the respective confidence intervals.

The percentile bootstrap interval is simply the range between the $100 \times (\frac{a}{2})$ and $100 \times (1 - \frac{a}{2})$ percentiles of a distribution of resampled θ estimates, where θ indicates the parameter of interest and a denotes the level of significance (e.g., 0.05 for 95 percent CIs).[16] The following sequence indicates how to get a bootstrap percentile CI of $\hat{\theta}$ ($\hat{\theta}$ denotes an estimator of θ):

1. B random bootstrap data are drawn with replacement from the original distribution of estimates. The size of each draw should be approximately the size of the original sample.
2. Each bootstrap sample yields a parameter estimate
3. All B bootstrap parameter estimates are sorted from lowest to highest

where B is the number of rounds data will be sampled. The confidence interval is then calculated as follows:

$$\left[\hat{\theta}_{\text{lower limit}}, \hat{\theta}_{\text{upper limit}}\right] = \left[\hat{\theta}^*_j, \hat{\theta}^*_k\right] \tag{4.16}$$

where $\hat{\theta}^*_j$ denotes the jth quantile which is the lower limit and $\hat{\theta}^*_k$ denotes the kth quantile which in this case is the upper limit. The indices j and k are given respectively by
$j = \left[\frac{\alpha}{2} \times B\right], k = \left[\left(1 - \frac{\alpha}{2}\right) \times B\right]$. In order to bring an example, a 95% percentile bootstrap CI with 1,000 bootstrap samples is the interval between the 25th quantile value and the 975th quantile value of the 1,000 bootstrap parameter estimates.

[16] Bradley Efron. *The Jackknife, the Bootstrap and Other Resampling Plans*. Society for Industrial and Applied Mathematics, jan 1982. DOI: 10.1137/1.9781611970319

5 Data Preprocessing

In this part, the reader will be introduced to the different datasets that were used in this work. The origin and acquisition of the data will be explained as well as the necessary work for making them comparable with each other. In order to familiarize the reader further with the dataset, some basic analysis on the wind farm will be performed.

5.1 ECMWF forecasts

5.1.1 ECMWF model and data

The European Centre for Medium-Range Weather Forecasts(ECMWF) is an independent intergovernmental organisation that provides its products to the national weather services of its member states. There are more institutes providing the same forecasts with slightly altered conditions or parameters so there is no single best or right way to do it. At its core, ECMWF uses Numerical Weather Prediction(NWP) models which solve the governing equations of the atmosphere. [1]

[1] Conor Sweeney, Ricardo J. Bessa, Jethro Browell, and Pierre Pinson. The future of forecasting for renewable energy, 2020. ISSN 2041840X

NWP models first run a simulation covering the whole planet and then more localized models are run in order to be able to increase the resolution and drive computational time down.[Sweeney et al., 2020] ECMWF offers a horizontal resolution of 9km on the localized models. That is the resolution in the data that we have as well.

The ECMWF data that we have is provided by Whiffle because their simulations are run on top of that data. It has an hourly temporal resolution and is arranged in a 20x20 grid centered around the wind farm at its native resolution (0.125deg) which translates to 9km at the wind farm latitude, as mentioned above. A representation of just a part of the ECWMF forecasting grid over the wind farm can be seen in Figure 5.1.

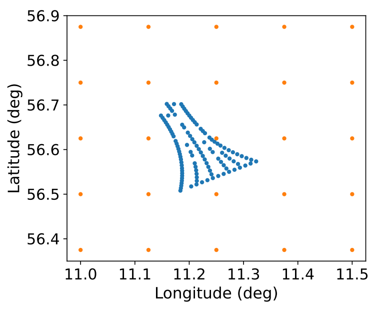


Figure 5.1: The blue dots in the middle represent the turbine arrangement of Anholt wind farm. The orange dots is the 5×5 ECMWF grid around the wind farm on which the weather is forecasted. The original grid in the dataset is a 20×20 grid in which the farm is barely visible. More on the wind farm arrangement in subsection Anholt wind farm details.

The variables included in the dataset are the longitudinal and latitudinal wind (m/s), time(Gregorian calendar) and air density(kg/m^3).

5.1.2 ECWMF data preparation

Since the data in this work will be analyzed at the turbine level, there is need for further processing in order for the quantities to be comparable with the other datasets, both in time and space.

First of all, the grid data points have to be interpolated at each individual turbine. Linear interpolation was chosen which is straightforward and did not require tuning like other spatial interpolation methods require, e.g. Kriging. We chose linear instead of some higher order interpolation method because our values were bounded. Interpolated values in a cubic approach may provide more smooth and accurate results but lay on second order polynomial curves. Therefore the curve fitting can result in the interpolated point being outside the range of accepted values for some quantities (negative power or wind values).

Both GRAPS and Ørsted forecast datasets provide the wind speed magnitude and direction. This means that for ECMWF, the wind magnitude and direction had to be calculated from the x and y components of the wind speed. There are two ways this can be done, both of which are correct logically. First interpolate each component to each turbine and then add or in the reverse order. The result is different because the component addition is not a linear operation. It was decided to first interpolate the x and y values and then calculate wind direction and magnitude. After having interpolated the quantities at each turbine location, they were linearly interpolated in time so that we had 6 data points in each hour instead of 1. That is because the temporal resolution from the GRASP simulation was 10 minutes so this was necessary to match the data temporally.

5.2 *Whiffle forecasts*

5.2.1 *GRASP model and data*

Whiffle utilizes their GPU-Resident Atmospheric Simulation Platform (GRASP for short) in order to perform calculations of the wake effects at Anholt wind farm. In the heart of GRASP lies an LES model commonly referred to as DALES or Dutch Atmospheric Large Eddy Simulation. [2] The difference is that most of the routines are run on Graphics Processing Units(GPUs) making the once time consuming LES models much more realizable.

[2] T. Heus, C. C. Van Heerwaarden, H. J.J. Jonker, A. Pier Siebesma, S. Axelsen, K. Van Den Dries, O. Geoffroy, A. F. Moene, D. Pino, S. R. De Roode, and J. Vilà Guerau De Arellano. Formulation of the Dutch Atmospheric Large-Eddy Simulation (DALES) and overview of its applications. *Geoscientific Model Development*, 3(2): 415–444, 2010. ISSN 1991959X. DOI: 10.5194/gmd-3-415-2010. URL `www.geosci-model-dev.net/3/415/2010/`

Large-eddy simulations (LES), compared to usual NWP simulations, have a typical resolution of 10-100 meters or less and therefore directly resolve turbulence and boundary layer clouds. The boundary conditions are taken from the ECMWF deterministic forecasts [3] and wind turbines have been modelled in the simulation as described by [Meyers and Meneveau, 2010]. Since this additional layer introduces physical processes such as turbulence that were not resolved before, additional information can indeed be created in the process.

[3] Ciaran Gilbert, Jakob W. Messner, Pierre Pinson, Pierre Julien Trombe, Remco Verzijlbergh, Pim van Dorp, and Harmen Jonker. Statistical post-processing of turbulence-resolving weather forecasts for offshore wind power forecasting. *Wind Energy*, 23(4): 884–897, 2020. ISSN 10991824. DOI: 10.1002/we.2456

Every 12 hours ECMWF releases a simulation for the next 10 days for commercial use. The GRASP model retrospectively creates each daily simulation by using the most recently released ECMWF forecast as if it were released then.[4] The GRASP model is run for a whole day and is re-updated after 24 hours to run the next day. Since different parts of the day are distanced differently from the update point, their predictive capability also varies. This means that we need to handle each day individually as we cannot make a continuous prediction.

[4] This will be explained in more detail in section Day ahead prediction concept of the Methods chapter.

5.2.2 GRASP data preparation

The GRASP data is provided in NetCDF files that are optimal for containing multivariate, multidimensional data. Each day was stored in a separate NetCDF file making handling more difficult, inciting us to concatenate the files in a single one. The different files were combined in a 3 dimensional array with day of prediction, hour of prediction and turbine identity being the three dimensions in this order. A separate array was created for each quantity in the original NetCFD file. The quantities relevant to the wind were the *free stream wind speed* and the *rotor disk-average wind speed*. The former calculated the wind speed as if no wind turbines were affecting the wind stream and the latter included how the wind interacted with the wind turbines. [5] The *free-stream wind direction* and *rotor disk-average wind direction* held the wind direction information in a similar manner as the previous ones. Lastly the *rotor disk-average air density* contained information about the air density after the wind had interacted with the turbine.[6] The aforementioned quantities were offered at each turbine location. There was another file containing general meteorological information for the greater wind farm area. It held variables such as *wind speed*(single value), *meteorological wind direction*, *air density*,*temperature*, the *inverse obuhkov length* and *friction velocity*. The most useful ones were the last four which were concatenated, each one in a single array and used

[5] Mnemonic tip: free stream refers to the quantity as if no turbines were present in the simulation while rotor disk refers to the the quantity as if turbines were taken into account in the simulation and the quantity was affected by the interaction with them.

[6] The turbine blades create low pressure points behind them as they move through the air and high pressure in front of them.

5.3 Ørsted wind farm

5.3.1 Data collection

This endeavour would not be possible if there was no way to validate our assumptions. We were in the privileged position to have access to the confidential[7] data for Anholt wind farm, which is operated by Ørsted. Ranging from January 1st 2013 to 30th June 2015, the dataset spanned a total period of two and a half years.

[7] We are not allowed to divulge any sensitive data or information related to the wind farm. All data is property of *Ørsted A/S*.

Several physical quantities are measured at each wind turbine. Each quantity was measured continuously but the data has been processed into 10 minute intervals. Each interval is characterized by a minimum, average and maximum and standard deviation value describing each quantity. The most important variables in the dataset are: *TimeStamp*, *Wind Speed*, *Yaw Position* and *Active Power*. *TimeStamp* indicated the time at the end of the 10 minute period. *Wind Speed* indicated the wind speed measured in m/s by the turbine's corresponding instrument. [8] *Yaw Position* determined angle the nacelle was facing which is usually the direction of the incoming wind since the nacelle alligns with it. It is measured in degrees from the North Pole. *Active Power* measured the power the wind turbine produced or consumed. There are additional meteorological or turbine quantities being measured but since they were not included in the analysis, they will not be mentioned here. There is also additional information on the profile of the turbine such as the hub height, the nominal capac-

[8] That can be either an anemometer in older wind turbines as depicted in Figure 3.5 or a Lidar sensor in our case.

ity of the turbine and some geographical data such as the location coordinates.

5.3.2 *Anholt wind farm details*

Anholt wind farm is located 21 kilometeres North-East off the coast of Djursland peninsula close to Anholt island, hence the name. The wind farm has been erected during the period 2012-2013 and covers an area of $88km^2$. It has 111 operational wind turbines. Each wind turbine has a nameplate capacity of $3600kW$ making the wind farm capable of producing $400MW$ at full power. It became fully operational in Q3 of 2013. [9]

[9] *Ørsted A/S*. Djursland anholt offshore wind farm. URL `https://orstedcdn.azureedge.net/-/media/www/docs/corp/com/our-business/wind-power/wind-farm-project-summary/anholt_uk_2018.ashx?la=en&rev=3b2f175b341e4ca2b906bf2a2ff68c8f&hash=51B4609E53A631D29E0EF49936C33A84`

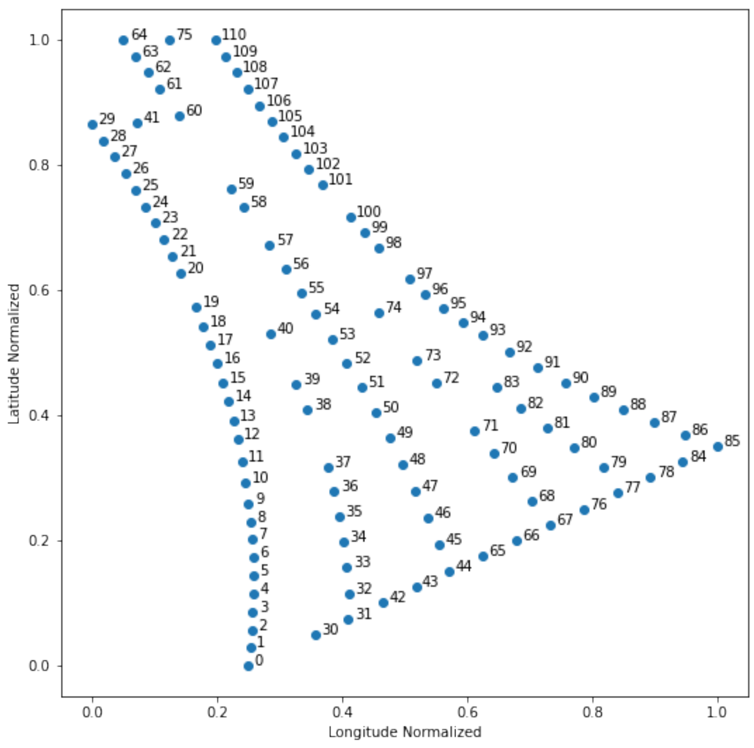


Figure 5.2: The Anholt wind farm turbine arrangement. Each dot and number represents a turbine. As is clear, Anholt wind farm has a rather interesting arrangement of its installed turbines. The standard is to place the wind turbines in a rectangular grid. The coordinates are normalized with the range of values. The numbering of the turbines is something we used that made our analysis easier.

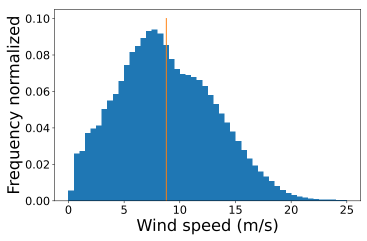


Figure 5.3: This is a histogram of the wind speed intensity throughout the whole year. Extreme values above 25 m/s are not included. The number of bins is 50 and the height represents the normalized frequency of each wind intensity. The mean wind intensity is 8.8 m/s and the histogram has been produces using the actual data measured at the wind farm.

The wind farm has an innovative arrangement considering that the conventional approach was to place the turbines in a rectangular grid. It can be seen in Figure 5.2. The arrangement of Anholt is slightly more efficient than a traditional wind farm because it has been placed in a way that a larger number of turbines is facing the direction of the most common winds that Denmark usually has. The most common wind gusts come from the South-West (Fig 5.4), firstly interacting with turbine number 0. In this arrangement a large number of turbines on the left and bottom rows interact with the southwest wind. This way the amount of turbines that interact second or third in line with the wind are reduced. This is somewhat beneficial for the second and third rows because of wakes that are introduced when a turbine interacts with the wind.

By analyzing some of the wind direction data we can see that the shape of the wind farm is justified. Figure 5.4 is a Wind Rose of the wind data that was collected by the wind turbines. The 360 degrees were split into 36 intervals of 10 degrees and each bar represents the percentage of moments that the wind was blowing from that direction. It is clearly visible that the wind usually blows from the West and South-West as is also verified in the literature.[10]

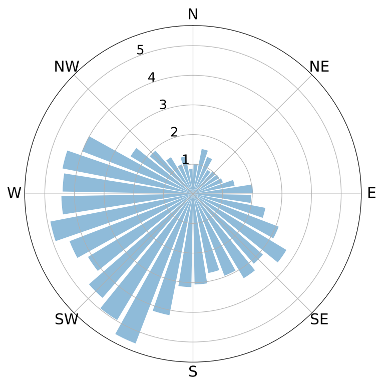


Figure 5.4: This is wind rose diagram of the wind at Anholt wind farm. The angle represents the direction of the incoming wind and the radius represents the percentage of data points that had the respective direction. Each circle represents an increase of 1%.

In Figure 5.5 one can see the normalized annual production of each wind turbine. The invisible centers of the circles represent are actual turbine location. Coordinates are normalized and the color bar is also adapted to the range of the values. We can see that the turbines with numbers between 80-83 and 87-97 which lay in the further NE end, opposite of where the wind usually comes from, have a smaller annual production.

[10] M. P. Van Der Laan, A. Pena, P. Volker, K. S. Hansen, N. N. Sorensen, S. Ott, and C. B. Hasager. Challenges in simulating coastal effects on an offshore wind farm. In *Journal of Physics: Conference Series*, volume 854. Institute of Physics Publishing, jun 2017. DOI: 10.1088/1742-6596/854/1/012046

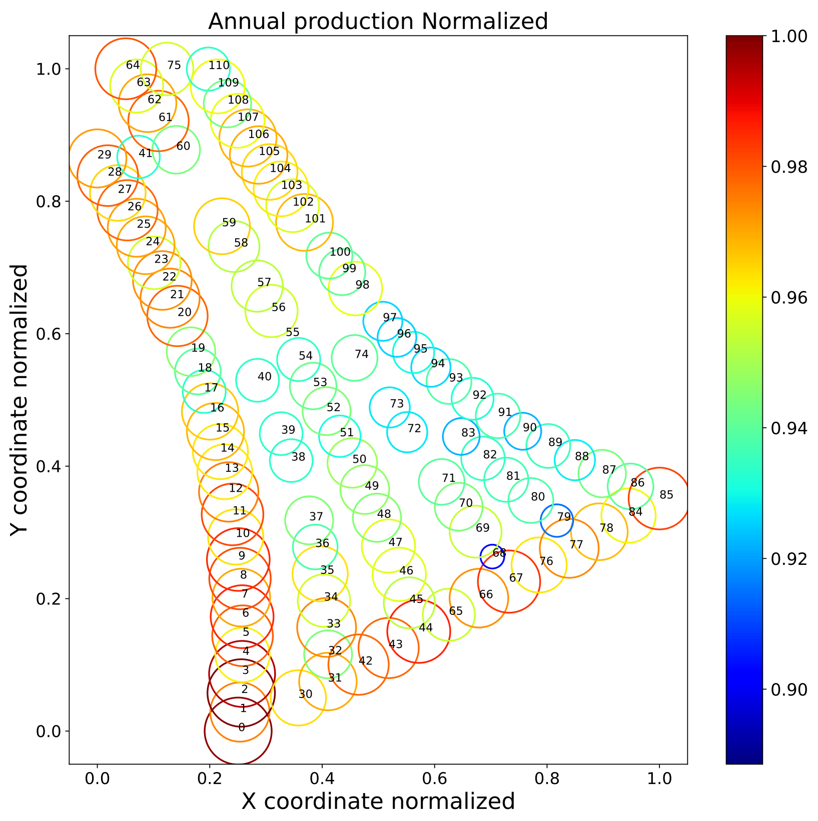


Figure 5.5: The annual production of the wind farm can be seen here. We can see that the arrangement of the wind farm is indeed beneficial because in a square arrangement the more turbines would be placed behind turbines in light blue which would eventually result in even less production. The values have been normalized due to data confidentiality. Turbine number zero has the maximum production.

In the following graphs one can see the average monthly wind speed and average monthly turbine production. This is in order to show that there are many seasonal variations in the behaviour of the wind and the wind farm power output. This also means that temporaly, different parts of the dataset are unbalanced and some months contain on average certain conditions more than others. Since the analysis of our objective is time dependent it means that there might be imbalances in the test sets and train sets when we implement our Machine Learning algorithms.

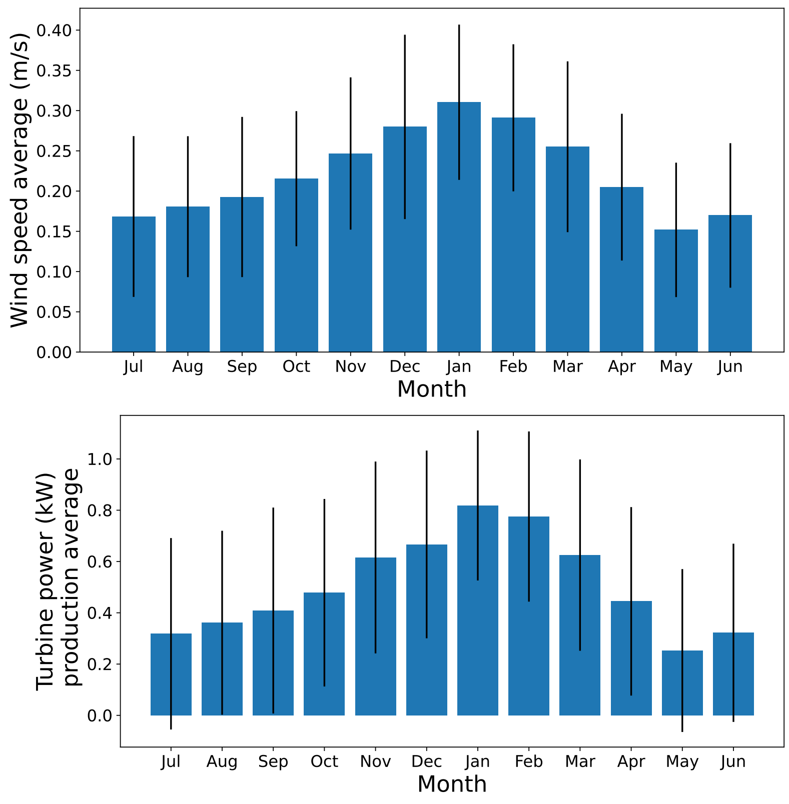


Figure 5.6: These are the monthly averages for wind speed and wind power. Winter months have higher winds as is usual resulting in more energy production per turbine. The error bars are the standard deviation of the values from the mean for each month. The values have been normalized due to data confidentiality.

5.3.3 *Ørsted data preprocessing*

This subsection will analyse why only a small part of the data was chosen and how it was prepossessed. This is an important step and will be explained thoroughly because it was important for us to find the optimal period to run our analysis. The reason for that is because our original dataset is not perfect and the forecast data that we use is produced externally. It takes a some time to set up the algorithms, run the simulations and lastly each run costs a lot of money. All the above meant that we had very limited amount of times we could ask for data so the remedy was to make sure we would ask for the right data periods to be run each time.

The ECMWF forecast datasets were available for many years and could be easily acquired. Then the GRASP forecast simulation could just be run on top of that, so these two datasets did not impose any constrains on the amount of data or the period that would have to be. However the first dataset that was available was the Ørsted dataset and being finite meant that a lot of decisions would be made with regard to the Ørsted wind farm dataset.

The Ørsted dataset was given in 5 files, each containing 6 months of measurements. The different files had to be sewed together as with the other forecast simulation data, to make handling more efficient. Initially, each quantity of interest was stacked from the 5 different files in an 2D array with time and turbines being the rows and columns respectively. By looking through the dataset we come across many missing(NaN or Not_a_Number) values as well as negative values. Figure 5.7 shows the sum of NaN values across our 111 turbines for each moment in the dataset. The plots are made by using the power measurements but the behaviour is similar for the other quantities as well. It is immediately visible that in the beginning of the dataset (January 2013) the sum of missing values in the farm at each moment is very big. The number is declining and practically drops to zero around July of 2013. That observation is very logical since it is around that time that the installation of the last wind turbine finished and the whole wind farm became operational. It is clear that this is a not a good period to be included in our analysis. NaN values that appear further in the dataset are probably maintenance periods, damage to some instrument, failure to record data, etc.

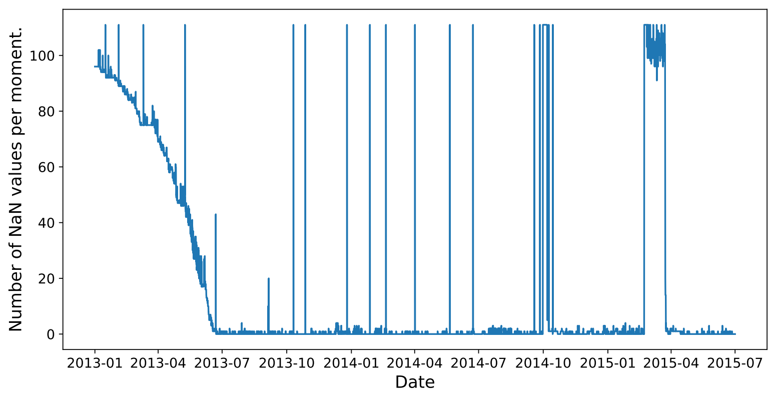


Figure 5.7: Amount of missing values for each quantity at each moment. In this case number 40 means that 40 turbines were not operational yet or that the respective quantity was not measured at that moment in time.

This initial period of 6 months with that amount of missing data put some extra constraints on our ability to use the whole dataset. In addition to that the dataset contains negative power values to a large extend. These negative values are when the turbine uses more power than it produces. This might happen when the wind is low and the turbines do not produce more than they consume[11] or when the wind is too strong where, again the turbine stops rotating as a measure of safety. Negative values are equally distributed throughout the dataset as this is weather dependent(lack or strong presence of wind). Figure 5.8 shows the sum of turbines in each time stamp that had negative values, throughout the whole dataset.

[11] The turbines consume energy for power electronics, cooling systems, blade rotation mechanism etc..

It might look that the amount of moments where most of the turbines had negative power output values were too many or even more than the positive ones. That is not the case. The amount of negative values in the complete dataset is around 11%.

What we wanted to was to find a period in the wind farm data that was relatively low on missing values and also had a small number of negative values. We would have to find that period so that we could also get the ECMWF forecast data and run the GRASP model on that period as well. This way we would be able to compare if

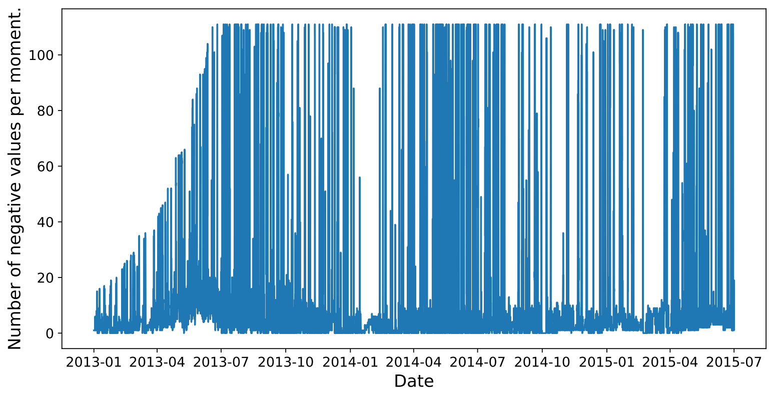


Figure 5.8: Amount of negative values for each quantity at each moment. In this case a point with magnitude 40 means that 40 turbines at the same time have a negative power production in the 10 minute interval that the quantity is measured.

using the GRASP model provides a forecasting advantage over just using the ECMWF forecast data to predict future energy production. What was done in this case was to find the period in the data in which the sum of missing and negative values for the next 6 months was the minimum. That can be seen in Figure 5.9. The amount of negative values was also considered since they are not the desired power quantities we want to predict. In a different and more accurate setup one might have wanted to include the energy consumption in the forecasts since they would theoretically reduce the total energy production of the wind farm. However the energy consumption of the turbines is minuscule compared to the amount of energy the turbines produce in an average operation capacity.

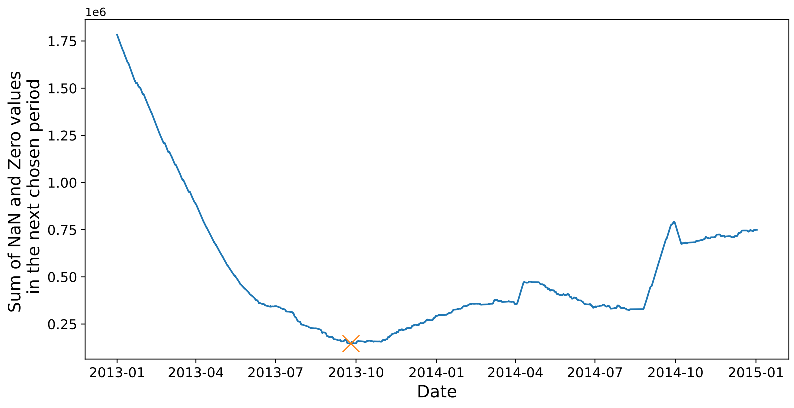


Figure 5.9: Y axis represents the sum of negative and missing values in the next 6 months. The yellow X symbol marks the start date at which the next 6 months contain the least amount of missing and negative values. The plot spans logically 2 years instead of 2.5 which is the actual dataset length, as each value holds the next half year's amount of undesired data points.

The 6 month minima is located at 2013-09-25 06:10:00. Initially, we requested for the ECMWF and GRASP data to stretch from 2013-09-25 to 2014-03-25. We performed our initial analysis on the 6 month dataset and got our first results but in the process we got larger ECWMF and GRASP forecast datasets. [12] That happened because it can be time consuming to retrieve the ECMWF data from the servers, parse them and then run the GRASP simulation on top of them. Running the latter can also become expensive as the amount of data increases, so it was important to make sure that we would have

[12] In the beginning we used 6 months to calculate where the best period in the wind farm data would be. Later we increased the amount of data to 12 months and that is why the start date of the set does not match the exact date in Figure 5.9.

some initial results before proceeding to a larger dataset. Since we succeeded in doing that, I am going to present just the latest results from the 1 year long forecasting period.

The 1 year long dataset was created by adding three months before the start and three months after the end of the original set. The final dataset spanned the period from 2013-07-01 to 2014-07-01. This 1 year period included both missing and negative values as was expected. The negative values would be simply set to zero as we did not want to increase the complexity for our Learning Algorithm.[13] That is so, because apart from predicting the power production when the wind is blowing, it would need to predict the amount of energy a wind turbine would consume for its operation when it would not produce energy. This whole part(negative energy production prediction) of the learning process is now reduced to 1 single case, just zero produced power.

[13] However that was done only after the missing NaN values had been interopolated since they could have been both positive or negative, had they been measured properly.

The negative values in the power represent one specific case. That the wind turbine does not produce more than it consumes. The missing(NaN) values on the other hand mean that the quantity was not measured or is simply missing from the dataset at that moment for whatever reason. That value could have been either negative, zero or positive at full power capacity. Figure 5.10 shows the amount of turbines that had NaN values at each moment throughout the whole dataset. One immediately sees that for some moments no power measurements had been saved at all and that generally there are just a few missing values evenly occurring throughout the dataset. These values could not be simply set to zero because they would influence the coherence between the measured wind farm data and the simulated data on both the individual as well as the aggregate level. In the Methods section I will explain how I interpolated these missing values with the use of the Kriging interpolation method .

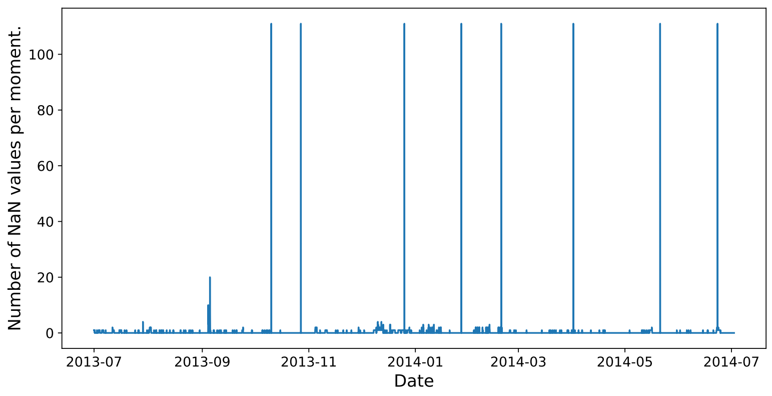


Figure 5.10: Plot is similar to Figure 5.7 with the difference that it only represents the final 1 year selected period from the whole Ørsted wind farm dataset.

The following table gives a feeling of the dataset before and after selecting the 1 year period and the type of values that govern the dataset.

	Complete dataset	Selected Period
% of NaN values	17.15	0.11
% of Negative values	11.11	10.86
% of Negative values to Positive	12.51	12.20

The first line in the table represents how much of the corresponding dataset consists of NaN values. We can see that the complete dataset has a very large percentage of missing values primarily due to the first part 6 months of the data. The 1 yead period that we extracted has only 0.11% missing values. The amount is quite small and interpolating those values is deemed a safe process. The second line is the amount of negative values over the amount of finite values.[14] That means that in the complete dataset almost 30% are values that we are not interested in since we want to predict actual power production(positive values). The percentage is similar also in the 1-year selected period since we said this is a weather dependent occurrence. The last line in the table represents the amount of negative values over the positive ones. This gives us a final measure of the amount of values that are beneficial for our Machine Learning analysis. In this case, the amount of positive values that are of interest in the final period, if we want to be exact, is $(1 - 0.0011) \times (1 - 0.122) = 0.877 = 87.7\%$. That is a satisfying percentage to work with.

[14] Both positive and negative values but not missing.

6 Methods

This chapter is devoted to analysis and explanation of the methods, techniques and implementations what were used to obtain the results. All steps that were of core importance to reach the final results will be explained here.

6.1 *Spatio-temporal interpolation with Kriging*

Section Ørsted data preprocessing in the Data Preprocessing chapter ended with the selection of a smaller part of the original data that was 1 year in duration. It was chosen because it contained the smallest amount of negative and missing values of all the possible 1 year periods throughout the dataset. Figure 5.10 shows the amount of missing values in the selected period for the power values. Similar are the missing values for all the other variables with some minimal differences. NaN values prevent us from being able to use the dataset with confidence and ease and therefore it was decided to interpolate them already early on.

Consider a single moment in time in the wind farm as in Figure 5.2. If 2 turbines have missing power values, [1] the best and most correct way to interpolate them is to use the values of neighbouring wind turbines to do that since they hold the most recent information of the power output. In this case Ordinary Kriging is used to interpolate the values spatially. In Figure 5.10 one can see that the vast majority of the moments had between 0 and 5 missing values throughout the whole wind farm, about 5% of the total turbines number, which is considered safe to interpolate. This is safe to do as long as the amount of missing turbine values in each moment remain below a certain threshold. That threshold was set to 10 turbine values missing simultaneously at a single moment.

[1] Could be either wind speed, wind direction, power output or any of the features. Each quantity is treated individually in the same way. However in this case the most relevant is the power since this is the quantity we use to evaluate and train out ML algorithms.

However, there are some moments that all turbine values are missing. Those moments have to be interpolated in time, which means using the past and future as an extra spacial dimension. There are some logical constraints on this as well, such as the fact that if all turbines are missing for 2 consecutive days, it is not reasonable to say that the intermediate period would have the same weather pattern as well. In this sense, one cannot be certain if the weather would be the same in the next 30 minutes, however the assumption is much more reasonable and the possibility of that being close to reality is high. In our case the maximum amount of all 111 turbines missing consecutively is 2. Since every moment covers a 10 minute period the maximum period without any data is 20 minutes and it occurs only once in the first spike. All the other spikes are individual moments missing. Summarizing, up to 10 turbines at each moment were interpolated spatially in 2 dimensions and above that the temporal dimension was also use as the 3rd dimension in 3D interpolation.

As explained in the theory, Ordinary Kriging is a 2D or 3D spatial interpolation method. So interpolating for less than 10 missing turbines is not an issue. In the case of using time as the third spatial dimension the following problem arises. How far should 10 minutes be placed as actual distance in the third dimension? If the distance between moments in meters is too small then the algorithm would interpolate only temporally and otherwise it would interpolate only spatially. We wanted to find the optimal way to do that. Figure 6.1 is a representation of the problem.

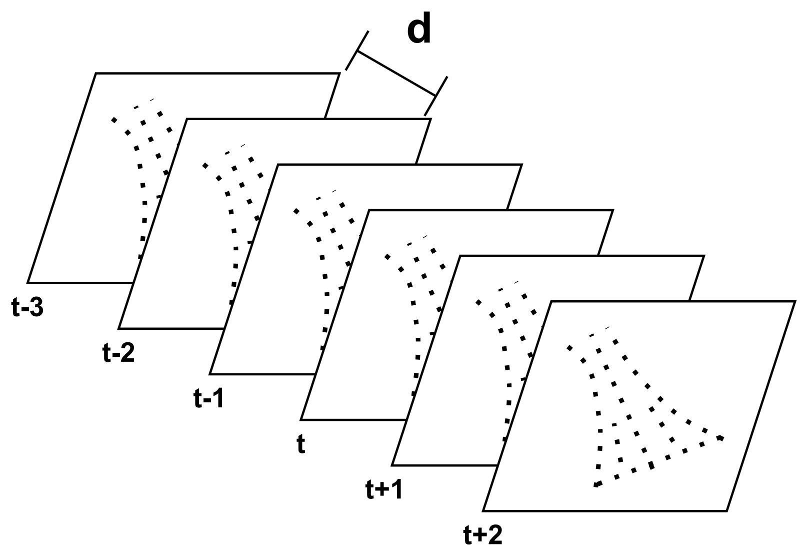


Figure 6.1: This is a representation of the farm where time is the extra 3rd dimension. The distance d denotes the distance of the temporal planes as if they were a spatial dimension. *This figure was created with diagrams.net*

Since the distances in time are spaced equally, a parameter d was tuned that lied between 0 and 1 that would space the temporal planes in the optimal distance from each other. All actual distances in the wind farm were normalized between 0 and 1 before fed to the algorithm so that it would make sense for parameter d to be between 0 and 1. To find the optimal value of d, a random distance was chosen, and then all the temporal planes e.g. up to time $t-1$ were given to the algorithm. Then an attempt to predict all the turbine values at planes t,$t+1$ and $t+2$[2] was performed. The experiment was repeated for multiple d values and the results were compared with the actual values at the corresponding planes. Figure 6.2 shows the results for all three lead times. In each lead time MAE is calculated between all predictions and actual values for multiple values of the time step parameter d. As is logical, by predicting further in time, the MAE minimum value in each lead time is higher because we distance our prediction further from the last known value(time $t-1$). However the

[2] The first moment in the future can also be called first lead time etc.

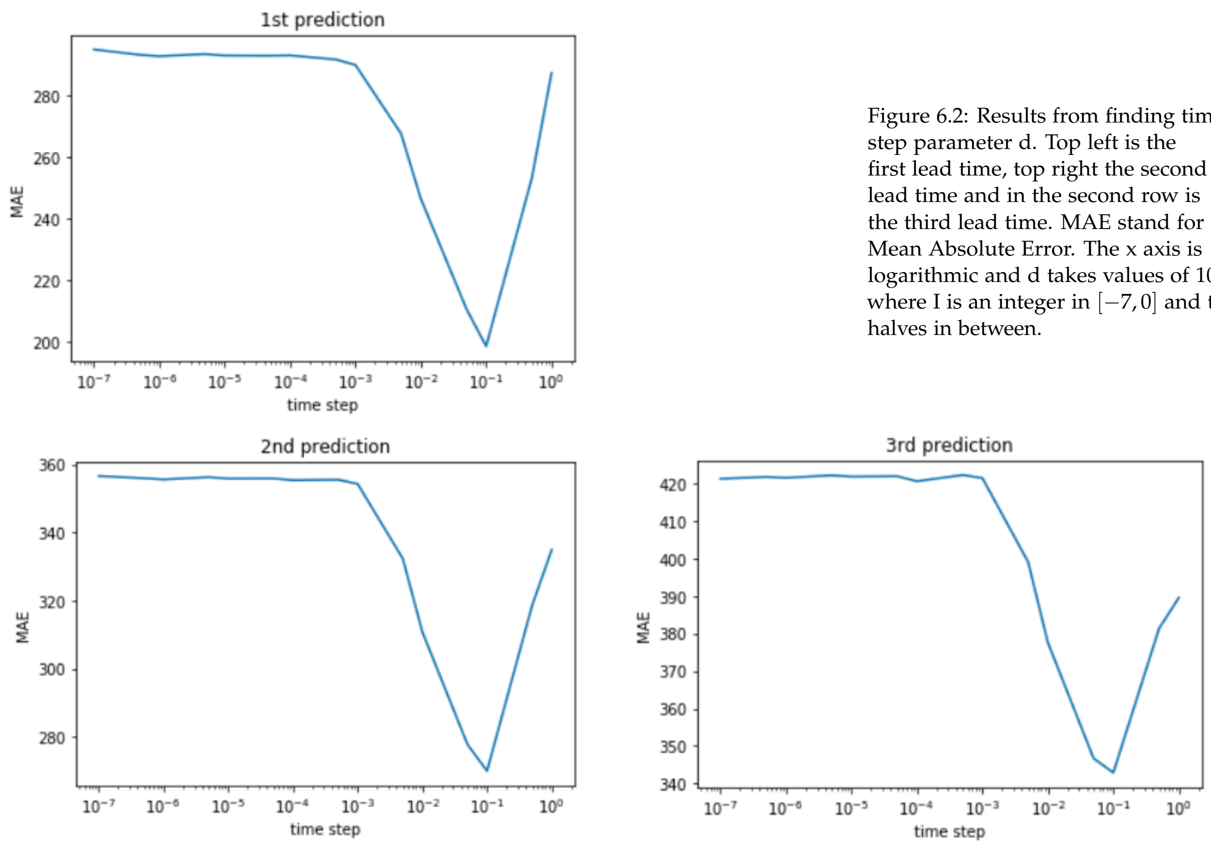


Figure 6.2: Results from finding time step parameter d. Top left is the first lead time, top right the second lead time and in the second row is the third lead time. MAE stand for Mean Absolute Error. The x axis is logarithmic and d takes values of 10^{i} where I is an integer in $[-7, 0]$ and the halves in between.

minimum still indicates that the temporal planes should be placed at d=0.1 from each other.

After finding the optimal value for parameter d, it was possible to temporally place the planes at the optimal distance. This way the moments that contained a lot of missing turbine values could easily be interpolated from the past and future moments in time. In the case of a single, whole plane missing, 3 planes from the past and 3 from the future were used for predicting the missing one. [3] The final dataset we have spans the period from 013-07-01 to 2014-07-01 and is finally ready to be used for predicting the day-ahead power.

[3] It has to be noted again that the negative values had not been set to zero before the Ordinary Kriging interpolation was finished. Some missing values could have been negative. If all negative values were set to zero at a certain moment, then the presence of a positive value could turn a missing value to positive that would otherwise be negative and eventually zero.

6.2 *Data flow sequence*

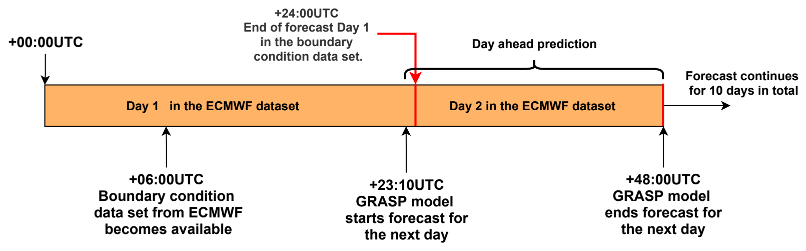
This section explains to the reader how the algorithm was structured in order to simulate the flow and arrangement of data to replicate a real world scenario.

Day ahead prediction means predicting the power for the coming day. In a real world scenario, usually the forecast data is made available at some point in time, each day. In our case, ECMWF forecast data are released every day at 00:00UTC and 12:00UTC with hourly temporal resolution up to lead time t+90, 3-hourly from t+93 to t+144, 6-hourly from t+150 to t+240. [4] This means that every 12 hours ECMWF releases a new prediction for the next 10 days from the release moment. After 6 hours, the forecast data for the

[4] 3.1 Product Schedules - Forecast User Guide - ECMWF Confluence Wiki. URL `https://confluence.ecmwf.int/display/FUG/3.1+Product+Schedules`

Boundary Conditions (BC) optional programme (for members using ECMWF IFS model forecast values as boundary conditions for their own local areas models) is released by ECWMF. [5] This boundary condition dataset is retrieved by Whiffle and used to predict the day ahead weather. The GRASP forecast simulation can of course be run anytime between the data acquisition and the beginning of the next day. In Figure 6.3 one can see the sequence of events and at which point in time each one takes place.

[5] 3.1 Product Schedules - Forecast User Guide - ECMWF Confluence Wiki. URL `https://confluence.ecmwf.int/display/FUG/3.1+Product+Schedules`



+00:00UTC
+24:00UTC
End of forecast Day 1 in the boundary condition data set.
Day ahead prediction
Forecast continues for 10 days in total
Day 1 in the ECMWF dataset
Day 2 in the ECMWF dataset
+06:00UTC
Boundary condition data set from ECMWF becomes available
+23:10UTC
GRASP model starts forecast for the next day
+48:00UTC
GRASP model ends forecast for the next day

Figure 6.3: The orange rectangle represents the 10 day forecast starting at 00:00UTC. However only two days are shown. The red line represents the finish of each day within the 10 day dataset. The GRASP model starts predicting at 23:10UTC, that is 1 hour before the day of interest starts so there is a smooth transition over to the next day.*This figure was created with diagrams.net*

6.3 *Day ahead prediction concept*

As we said before, the goal is to predict the power production for the next day. One universal model could have been created that would be trained on all the data and would be able to predict all the lead times. However it was decided to create different models for different lead times because they would be able to better specialize, compared to having a single model trying to capture all that information. Therefore, the day ahead power is going to be predicted one lead time at a time. For example, the model predicting at lead time t will be given all features relevant to it. The features for each lead time are split into two categories. The features related to the past and to the future. Future related ones are wind speed prediction, wind direction prediction, temperature etc which originate from forecasting models. The features related to the past are the last known power measurements, provided in the wind farm data.[6]

[6] We could provide other quantities such as wind related to the past but power is the one directly connected to what we want to predict so the others are not very useful.

As mentioned before we will try to compare two different ways of providing the features to predict the power at each lead time in our day ahead prediction. The first way consists of a model that takes in all features relevant to each lead time and predicts directly the aggregate power production in the wind farm. We refer to it as an aggregate-power-predicting model. Each lead time model is given the last known aggregated power produced by all 111 turbines and all features of each turbine for the corresponding lead time. Since the complete day ahead power prediction consists of n lead times, n models will be trained each one specializing on 1 lead time.

The second type of model will be referred to as an individual-turbine-predicting model. This time to predict each lead time, a separate model is dedicated to each turbine. So in total lead times$\times$111 turbines to predict the complete day ahead. To predict the aggregate for each lead time the results from the different turbine models are aggregated. Each turbine model is trained only on what it should produce itself, with just the features that relate to that specific turbine, hence the name individual-turbine-predicting model. Both models predict the same thing in a different way.[7]

[7] The reason for this has been explained in subsection Research questions of the Introduction chapter.

There are two ways one can provide the last measured power. Both of them serve a different purpose. Let us suppose that the current time is t=0 and we want to predict the power at t=+1 until t=+n. Each lead time is given the last known power measurement which is at t=0[8] and then we predict until the end of the day(t=+n, n depending on temporal resolution). This way we estimate the power once at the beginning of the day, for the whole day. The lead times such as e.g. t=+10 or t=+100 can only be given the power at t=0 because that is the last actual measured value at time of prediction. It is logical that the lead times closer to t=+n will benefit less from this since the last known power value is an indicator of how the wind farm was behaving shortly before and therefore loses its importance as a feature the further ahead you use it in the future.

[8] It is possible to provide the power from more moments in the past such as t=-1,t=-2,t=-3 etc. Depends on how many known power measurements benefit the learning algorithm.

The second approach differs because it applies the previous technique as new power measurements become available throughout the day. It starts at time t=0 by making the exact same prediction for the whole day. As measurement at time t=1 becomes available it predicts anew. This time in order to predict the rest of the day, we need to predict from time t=+2 until t=+n, which is one prediction less. The last measured power value is now at t+1. As more measurements become available, the fewer predictions need to be made each time to complete predicting the day and the more accurate the last predictions are compared to the other method.

These are two very different methods to predict for the day ahead power. In this work we have experimented more with the first one where one prediction is made at the beginning of the day. Having explained all of the above, the reader should be able to understand how the algorithm that could produce the desired result would have to be built and how the data should be given as features to XGBoost.

6.4 *Data structure and augmentation*

In this section, the process of data structuring and augmentation leading to better predictions will be explained.

Figure 6.4 shows a 3D container array representation of the structure of data for each quantity(measured power, wind speed, wind direction etc.) Since we want to predict each lead time in each day with a different model, the amount of data that we have in our disposal for each lead time is the amount of days, which in our case is 365. This approach was tested and the variance in the results was large. 365 data points is not enough to properly train a model that performs well.

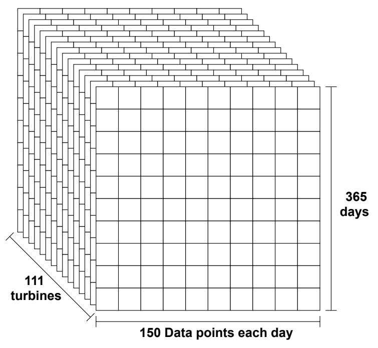


Figure 6.4: Data structure of each variable. Each column in days contains the values that will be used to predict the respective lead time. That holds for all quantities(wind speed, wind direction, free stream wind etc.) *This figure was created with diagrams.net*

In order to get around this hindrance what was realised is that some type of data augmentation technique had to be used. One such technique is the Sliding Window technique in which one uses the same data to create slightly altered versions of the same data. It is analogous to sliding a smaller window over each days data, seeing just a part of it every time. The cropped views are perceived as new data points for each lead time, new days in this case. Creating more examples from the same data by making use of the sliding window technique is not an ideal solution. It is always preferred to have enough data to train on because each new example brings forth something that the algorithm can learn from. However in our case the algorithm was under-performing to a large extend for the type of prediction that we wanted to evaluate. Therefore, the optimal solution is to create the least amount of copies of the data necessary so that the algorithm performs at a satisfactory level. This was evaluated on the test set, so it is possible that the algorithm would need less data to reach that point if new examples were given.

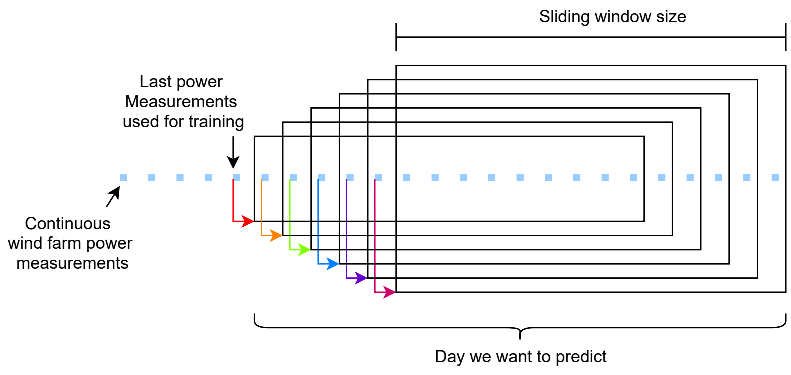

Sliding window size
Last power Measurements used for training
Continuous wind farm power measurements
Day we want to predict

Figure 6.5: Sliding window method for data augmentation. Rectangles represent the window slides over the data. Each window besides the features that are relevant to the lead times it encloses, is also given a copy of the last known power measurement(can be more than one) which is right before the start of each window slide. The intersection of all the boxes is the part that is most heavily replicated. *This figure was created with diagrams.net*

In Figure 6.5 there is a graphical representation of how the sliding window technique was used. The blue dots represent the continuous power measurements of the wind farm.[9] The bottom curly bracket marks the day ahead prediction period. The intersecting daily periods in the ECMWF and GRASP datasets start at 23:10 UTC and end at 48:00UTC with a 10 minute temporal resolution.[10] So in our case each day is 150 data points long. After multiple tests it was found that 5 slides were enough to make the algorithm more reliable. This augmented the original 365 data points for each lead time to $365 \times 5 = 1825$ data points. That was enough data to split among the train, validation and test set.

[9] Each blue dot represents 111 turbines...

[10] Each day can be considered completely separate from the previous one. Therefore it poses no problem that the end of each day overlaps for 1 hour with the start of the next one.

When the algorithm was tested, after data augmentation, it was noticed that there was a decrease in the performance at the lead times close to the beginning. When analysing further what could be the case for that, we discovered some days in the original wind farm data that were unusual. To be more specific, there were 7 days in the wind farm dataset where there was a sudden drop in the energy production, situated usually in the early hours of the day. In Figure 6.6 these sudden drops are visible. Since these drops are surrounded in time by full wind farm production, one could think that the wind became too strong in that period and the turbines stopped for safety reasons. However the wind measured by the sensors of the turbines indicate that the wind lied within operating limits. Therefore we don't exactly know why these sudden drops exist but they were removed from the dataset to prevent this decrease in performance. Of course the additional windows created by those "bad days" were also removed meaning that we removed $7 days \times 5 slided windows = 35$ less data points. This leaves us with $1825 - 35 = 1790$ data points to split among the train, validation and test dataset.

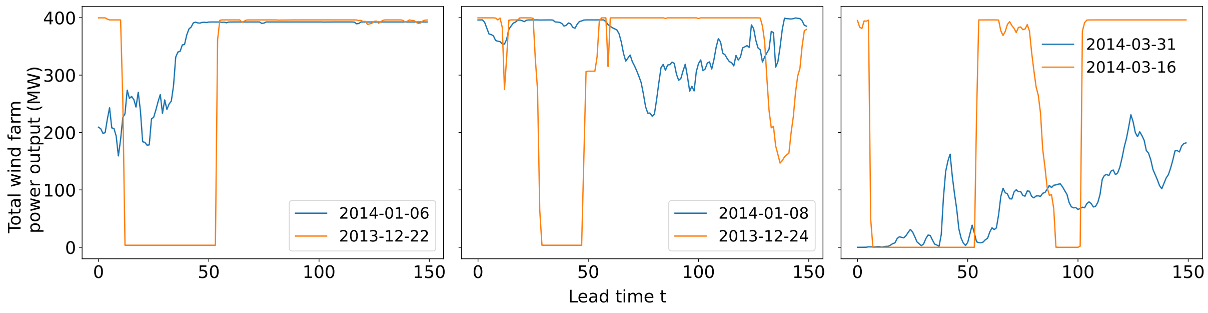


Figure 6.6: Example of three "bad days" in orange and three "good days" in blue throughout the year with their corresponding dates. The total amount of "bad days" exhibiting similar behaviour was 7.

6.5 Algorithm design, setup and prediction evaluation

In this section I will explain a little bit more about the structure of the algorithm and how everything is combined.

The structure of the algorithm used to run all Machine Learning models was built with versatility in mind. Just by changing a few parameters the user can run either an aggregate-power-predicting or individual-turbine-predicting model which can take a variable amount of features or previous known power values. Therefore the same algorithm can be used on predicting aggregate or individual models with input from either ECMWF or GRASP forecast data.

The first thing someone has to decide when running the algorithm is if an aggregate-power or individual-turbine model is desired, the amount of window slides and the features that are going to be provided during training of the model. After the sliding window technique is applied to augment the data(sliding 5 times), we ended up with 1790 data points for each lead time. Those 1790 data points would now be in the place of the 365 days as in Figure 6.7. Those 1790 days are split into train, validation and test sets with each one getting 1260(70%), 265(15%) and 265(15%) respectively[11] as seen in Figure 6.7.

Based on Figure 6.7, an explanation on how how each variable is input to each of the two model set-ups, will be given below. Each quantity available, whether that is wind or power, has the same structure as in Figure 6.7. In an aggregate-power-predicting setup the last known turbine power values that will be used will be summed across the turbine axis. That leaves us with 1 column per lead time which will be the first feature in the train set. The rest of the features from each quantity is the plane that is created if we take a slice of the

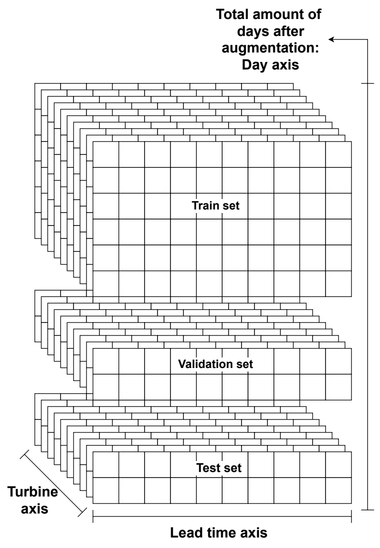


Figure 6.7: Visual representation of how the original data was split and how it was manipulated to produce the desired result. *This figure was created with diagrams.net*

[11] The amount of data in the splits is always multiples of 5(number of window slides) in order not to leak information from one set to other sets

data at lead time t. In the individual-turbine-predicting model each column combination of lead time/turbine is a separate model. The algorithm will train, aggregate and evaluate the model afterwards.

The result is evaluated with the Root Mean Square Error(RMSE) metric. The RMSE metric serves as measure of how good each prediction is. It gives an estimate of how close the estimation and the true value are. It is defined as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2} \tag{6.1}$$

The RMSE of each lead time is calculated on the aggregate wind farm actual output and prediction. In this case the summation in Equation 6.1 happens across the Day axis where the different predictions for each lead time exist. The day ahead prediction is now characterized by a number of RMSE metrics that are as many as the number of lead times. If a single number that characterizes the whole prediction is needed then the sum is calculated throughout the day and lead time axis.

The RMSE values should normally have power units in this analysis. However we decided to present the loss result as a percentage with respect to the maximum power output of the whole wind farm. For example, since the full aggregate power production of the farm is 400.000kW, an RMSE loss of 40.000kW would be presented as 10% RMSE loss with respect to nominal capacity.

It has to be noted that the extensive feature selection part only applies to the part of this work where we would like to see how much we can improve our prediction when we predict by using GRASP forecast data, as they come with a variety of different quantities. Since the ECMWF dataset that we have in our disposal provides only *wind speed* and *wind direction*, these are the two features that will be used when comparing the predictive ability of the two forecast datasets. It would not be fair to train the model using the GRASP dataset on wind speed, wind direction, temperature, air density, free stream wind speed etc. where for ECMWF only the two mentioned above could be used. Since ECMWF offers in general more quantities, this does not mean in any way that we are not running an accurate comparison. These two quantities are perhaps the most important and if these two are adequate to show that there is a sensible difference, that rule should apply as well in presence of more features.

6.6 *Over-fitting and early stopping*

In order to avoid overfitting, each model that was trained was subject to early stopping. This means that in an aggregate-power-prediction set-up where each lead time has a dedicated model, each individual model was checked at each training round to see if there was any improvement. The parameter early_stopping_rounds that is used in XGBoost was set to 5, meaning that after 5 consecutive rounds without improvement the model would stop training in order to avoid over-fitting. The same was done in an individual turbine prediction setup. In this setup each lead time consisted of the prediction of 111 turbines. Each one of them also had an early stopping threshold. Since so many separate models had to be trained, it was very difficult to find the optimal number of estimators that would result in the best prediction for each turbine or for each lead time, depending on the model that was being tested. Therefore the number of estimators in XGBoost was set to a high number, 120 specifically which was enough even for the longest case of training. By using early stopping each model would be able to stop training when it would be best for it.

In Figure 6.8 one can see how early stopping is intervening in the aggregate-power-predicting configuration. While the training loss continues to decrease, the validation loss has stopped improving. The fact that the model predicts better on the validation set early on is because of the difference in size of the two sets. The train set contains many more examples and therefore the evaluation when predicting it with a young model will not perform well.

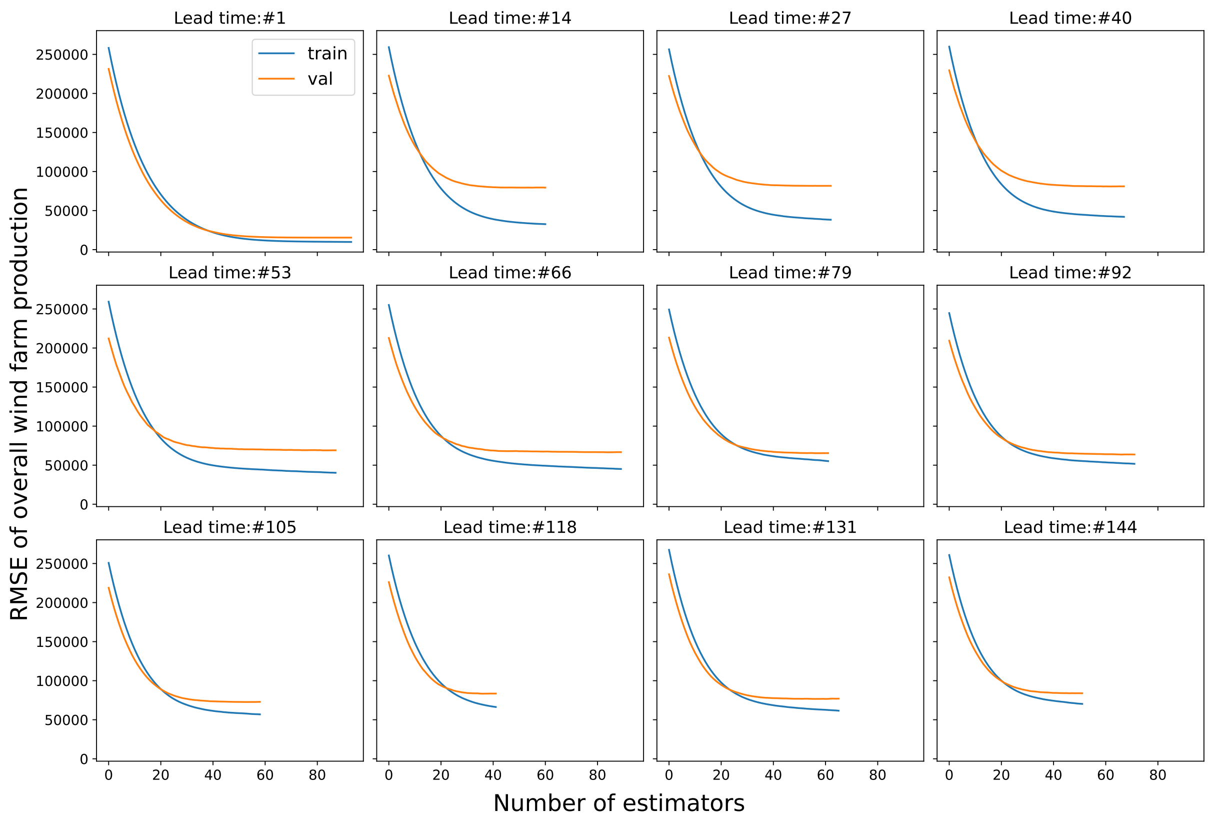


Figure 6.8: Early stopping rounds for the various lead times in the aggregate predicting model setup. The orange line is the RMSE when predicting the validation set and blue is the RMSE when predicting the train set. The lead times that are shown are picked evenly throughout the day that is to be predicted.

6.7 *Hyperparameter optimization*

A core part of Machine Learning is hyperparameter optimization. XGBoost comes with its built in hyperparameters that require optimization. Those are: *max_depth*, *min_child_weight* and *eta*. The first two parameters are used to control the complexity of the trees. It is important to tune them together in order to find a good trade-off between model bias and variance. The *eta* parameter is responsible for the learning rate. It controls overfitting. The hyperparameter tuning was performed with only two physical quantities(last known power measurements and wind speed) since this is the total amount of features that we can run with ECWMF forecast data either in an aggregate or individual-predicting model. Since the aggregate-power-predicting model differs in terms of amount of features from the individual-turbine-predicting model, two separate hyperparameter optimizations should be run in order to find the set of parameters that each one performs best with.

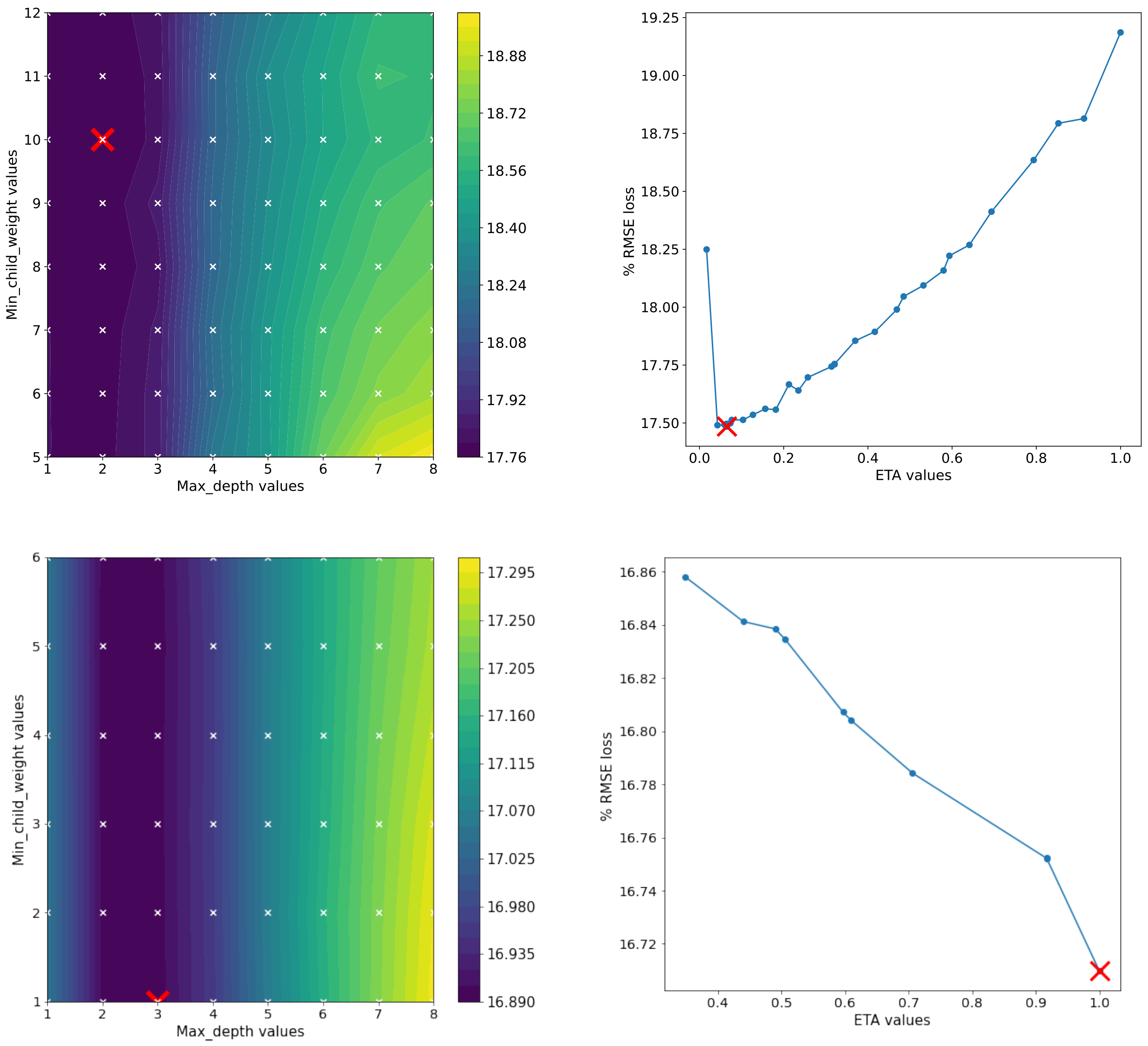


Figure 6.9: Top row is the hyperparameter optimization for aggregate-power-predicting model and bottom row is for the individual-turbine-predicting model. The left column represents the grid search optimization for *max_depth* and *min_child_weight* while the right column represents the Bayesian optimization for *eta*. The red cross indicates the parameters that minimize the loss function for each optimization.

Since *max_depth* and *min_child_weight* are integer parameters and usually take values in the ranges [1,8] and [5,12] which makes a total of 64 combinations, it was decided to use a grid search algorithm to explore all the combinations. In this case, this is the most exhaustive approach, as all of the parameters-space is covered. However the individual-turbine-performing model maxed at the boundary for *max_depth*= 2 and *min_child_weight*=5 initially so another round of hyperparameter optimization was run with in the ranges [1,8] and [1,6] respectively. The graphs in the left column in Figure 6.9 show how the space of the two parameters relates to the model performance. The *eta* value that was used was 0.1 and the rest of the parameters that would not be tuned were all set to their defaults. We can see that the minimum RMSE loss achieved for the two parameters is 17.76 percent for *max_depth*= 2 and *min_child_weight*=10 for the aggregate model. The individual-turbine model minimized for *max_depth*= 3 and *min_child_weight*=1 with a total RMSE loss of 16.89 percent. The red X mark is situated on the minimum for all the results. As we can notice from the vertical contour lines, *max_depth* seems to dominate the importance in the parameter space. Only for larger *max_depth* values, *min_child_weight* starts to impact the outcome as we can see from the bending contour lines. Judging from the result, the algorithm performs better when each of the sequential estimators is a short tree. he opposite case probably creates trees deep enough to overfit the data.

On the right column of Figure 6.9 the optimization for *Eta* can be seen. *Eta* takes continuous values, so it made more sense to explore the space with Bayesian optimization. The package used to run the Bayesian optimization is provided by [Nogueira, 2014]. The parameter values for *max_depth* and *min_child_weight* that were used are the ones obtained from the grid search hyperparameter optimization. The typical range of values for *eta* is [0,1]. We see that the minimum lies at *eta*=0.065 with an %RMSE loss of 17.48 for the aggregate model on the top row and at *eta*=1.0 with an %RMSE loss of 16.71 for the individual-turbine-predicting model.

The above hyperparameter tuning was performed by using the GRASP dataset. It could be argued that the ECMWF dataset would have a different optimal set of hyperparameters. However the guess would be that they will not deviate very much from these values since the same features and model structure would be used to tune that model therefore the only difference is in the provider and acquisition of the data. Table 6.1 holds the results from all the combinations

	Aggregate		Individual	
	ECMWF	GRASP	ECMWF	GRASP
max_depth	2	2	3	3
min_child_weight	11	10	1	1
eta	0.071	0.065	1.0	1.0

Table 6.1: Results from all the hyperparameter tunings. It is visible that the different forecast datasets do not have particularly different taste in hyperparameters, as is logical.

of hyperparameters, forecasting models and forecast datasets. The difference is not considered very big and a single set of parameters will be chosen, specifically that of the GRASP forecast dataset, in order to create a single model that will evaluate the different forecast datasets in each set-up.

6.8 *Parallelization*

One very important step that made this work possible was the parallelization of repeatable processes in the code. As we said before, in order to train an aggregate-power-predicting model to predict for the next day we would need to train 145 different XGBoost models, 1 for each lead time. While XGBoost is built with efficiency in mind, given the amount of features that were input to the algorithm, [12] it is logical that it would take a lot of time to run. In fact the average run time was about 6 minutes. The individual-turbine-predicting model has 111 models for each lead time, one for each turbine.[13] The average runtime for that set-up to run was 10 minutes.

[12] In a quick example of 111 last measured power values and two features per turbine means that the model had to train on $111 + 111 \times 2 = 333$ features. That is a large amount of features and a different model like that had to be run 145 times to complete training for the whole day.

[13] In a similar example like the one above, this corresponds to 111, 3-feature models for each lead time. That is in total $145 \times 111 = 16096$ models with 3 features each. That is a lot of models to predict a single day.

All these models are of course independent from each other so it is possible to run them in parallel. In the aggregate-farm-predicting model, the models for each lead time could be assigned to be executed at different cores and in the individual-turbine-predicting set-up each turbine/lead time combination could be sent to a different core to be executed. The results were then gathered together in the correct order to be evaluated.

The cloud computing service offered by the Copenhagen University allows each user to have a maximum of 8 computing cores and 16Gbs of RAM. The 8 cores were made up of 4 logical cores, each one having 2 threads, so 8 in total. That was also the amount of processes that were used to parallelize the loops. After parallelization, the average execution time for the aggregate-power-predicting model reduced to 1 minute and 15 seconds which yields a speed-up of 4.8, almost 5 times faster. The average individual-turbine-power-predicting model execution time reduced to 2 minutes and 15 seconds which results in a 4.5 times speed-up. Speeding up the code made it much easier to run more experiments, try different techniques and not regret a logical mistake at the end of each run.

7 Results and Discussion

This chapter is dedicated to presenting all the results produced in this work that were considered insightful, interesting or a milestone towards the previous two.

7.1 Persistence benchmark

Persistence forecasting is the simplest form of forecasting. In Persistence forecasting the last known measurement is used as the next prediction in the immediate future. That is of course very accurate for the first lead times and usually very hard to beat but its predictive ability decreases rapidly when used to forecast deeper into the future.

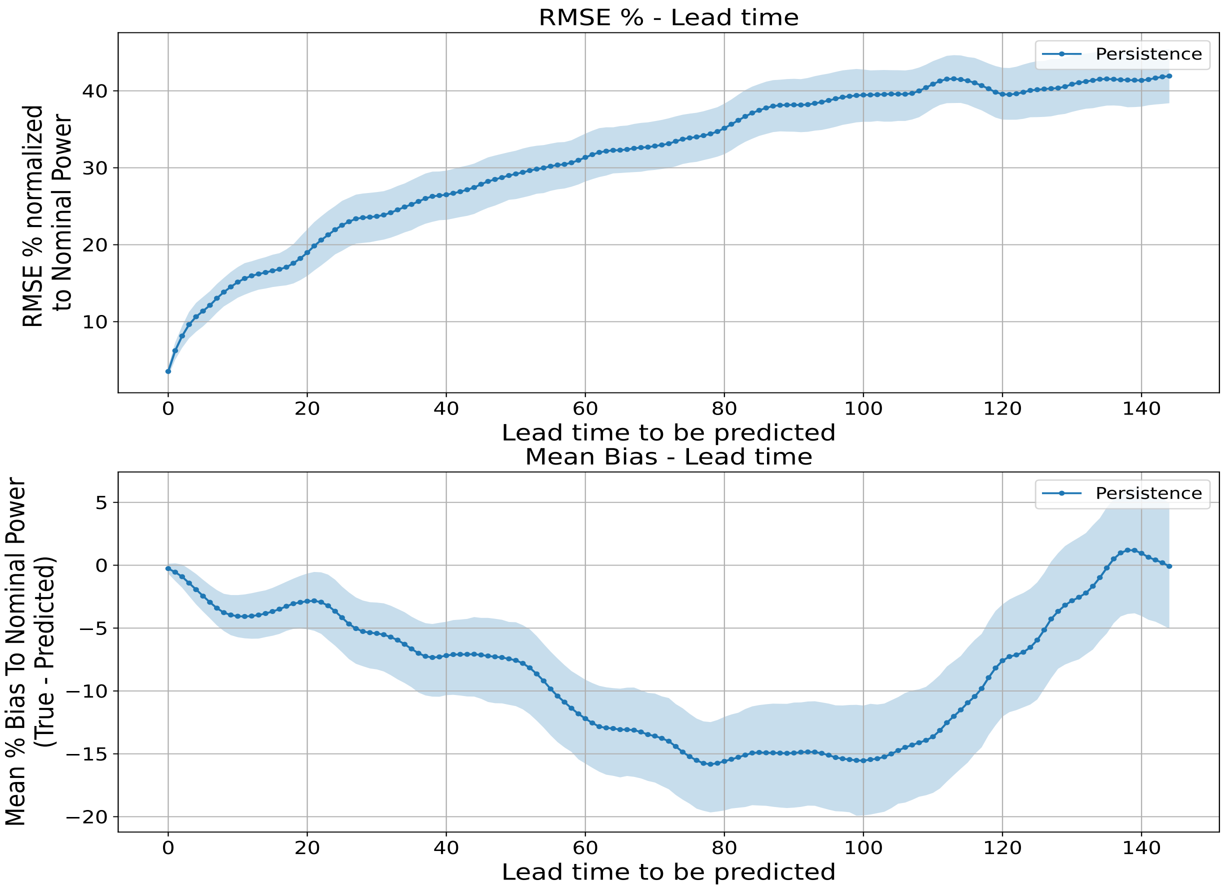


Figure 7.1: The persistence benchmark as calculated from our data. The blue line represents the % RMSE persistence result for each lead time. The light blue contouring marks the 95% confidence interval of the persistence benchmark. The lead times are 10 minute intervals from +23:10UTC to +48:00UTC the next day.

In Figure 7.1 we can see how the Persistence benchmark behaves when used to predict the day ahead power. The middle line represents the result of the corresponding graph and the semi-transparent blue margins represent the 95% Percentile Bootstrap Confidence Intervals. The top graph shows the RMSE throughout the day as a percentage of the nominal power.[1] The second graph shows the mean difference(bias) of the actual power production minus the prediction from our forecast method, in this case persistence. The mean bias for each lead time is presented again as a percentage of the nominal power so that the numbers are easier to comprehend and compare.[2]

[1] The way the metrics are calculated for each result has been explained analytically in section Algorithm design, setup and prediction evaluation

[2] All graphs in the next sections will hold similarly calculated results. Differences might include the axes over which the metrics are calculated but the principles remain the same.

The Confidence intervals are calculated as explained in section Percentile bootstrap confidence intervals of the Theory chapter. In this case the sample set is the power values that are predicted by our estimator, the persistence model in this case. The size of the test set is 265 "days", or slided windows to be more accurate. That is the sample size as well. 265 random samples are drawn with replacement, meaning that a sample can be chosen multiple times. The overall RMSE % is calculated for that sample and this process is repeated 1000 times. That happens to get a good enough distribution of the possible results from the different sample sets. The RMSEs % from all the different bootstrap samples are sorted and the 25th lower value is chosen as the low boundary of the confidence interval and the 975th value is chosen as the top boundary of the confidence interval.

As we can see the persistence forecast is doing quite well in the beginning, bounded by narrow intervals, meaning the predictions are reliable. However the RMSE prediction loss steadily increases throughout the day and peaks at the end of the day.[3] The overall RMSE % throughout all lead times is 32.86%. The reason why the RMSE increases throughout the day is intuitive but the reason why the bias in the bottom graph is forming a clear valley between the start and the end of the day is not. That valley means that persistence always predicts that less energy will be produced throughout the day with a peak minimum(between 10% - 20% less power) around the middle of the day.[4] Therefore we can assume that since the last measurement is always taken close to midnight, the power produced closer to 00:00UTC in the night is statistically almost always higher that the rest of the day except for when approaching the next night.

[3] It would be possible that the prediction accuracy of the persistence benchmark could decrease when reaching the end of the day again because there could be similar patterns of weather e.g. during the night. However, as it seems this is not the case because the weather behaviour in total could be completely different in the next day. A bigger test set could probably prove the previous hypothesis.

[4] One can think of lead times as evenly spaced, 10 minute intervals throughout the day. Lead time 0 is at 23:10 while lead time 60 is 10 hours after, at 09:10 in the morning, the next day.

The persistence forecast will serve as a basic benchmark against our forecasting methods, as, a model should not be considered at all if its predictive ability cannot surpass the simplest of forecasting benchmarks.

7.2 *Aggregate-power-predicting model*

In this section the aggregate-power-predicting model, that has been explained in the Methods chapter, is going to be tested and evaluated.

Since this section is a part of the comparison between GRASP and ECMWF forecast datasets, the aggregate-power-predicting model for GRASP will be given only the same two features that the ECMWF forecast dataset provides, the last known power and the wind speed.

Wind direction in this case had to be provided in the form of two features to account for the angular symmetry of the quantity. The rotational symmetry of degrees is something that is not straightforward for any machine learning algorithm. When dealing with degrees it is impossible for an algorithm to know that 360 and 0 degrees are the same. The solution is to feed the data first through a sin function that scales it between -1 and 1. This way continuity is satisfied however the algorithm will now confuse 0 and 180 degrees since they are both zero in the sin function. The cosine function also encapsulates angular symmetry and its value is -1 at 180 degrees and 1 at 0 meaning that the combination of sine and cosine as features to describe degrees is unique. This was attempted but it proved out that there was no benefit when given to the aggregate-power-predicting model(neither the individual-turbine predicting model), rather a decrease in performance which was unexpected. This is probably due to the consistency of the incoming wind direction in Denmark and the trade-off of having two extra features to try to learn from. It was decided to use only the wind speed as feature besides the last known power measurement.

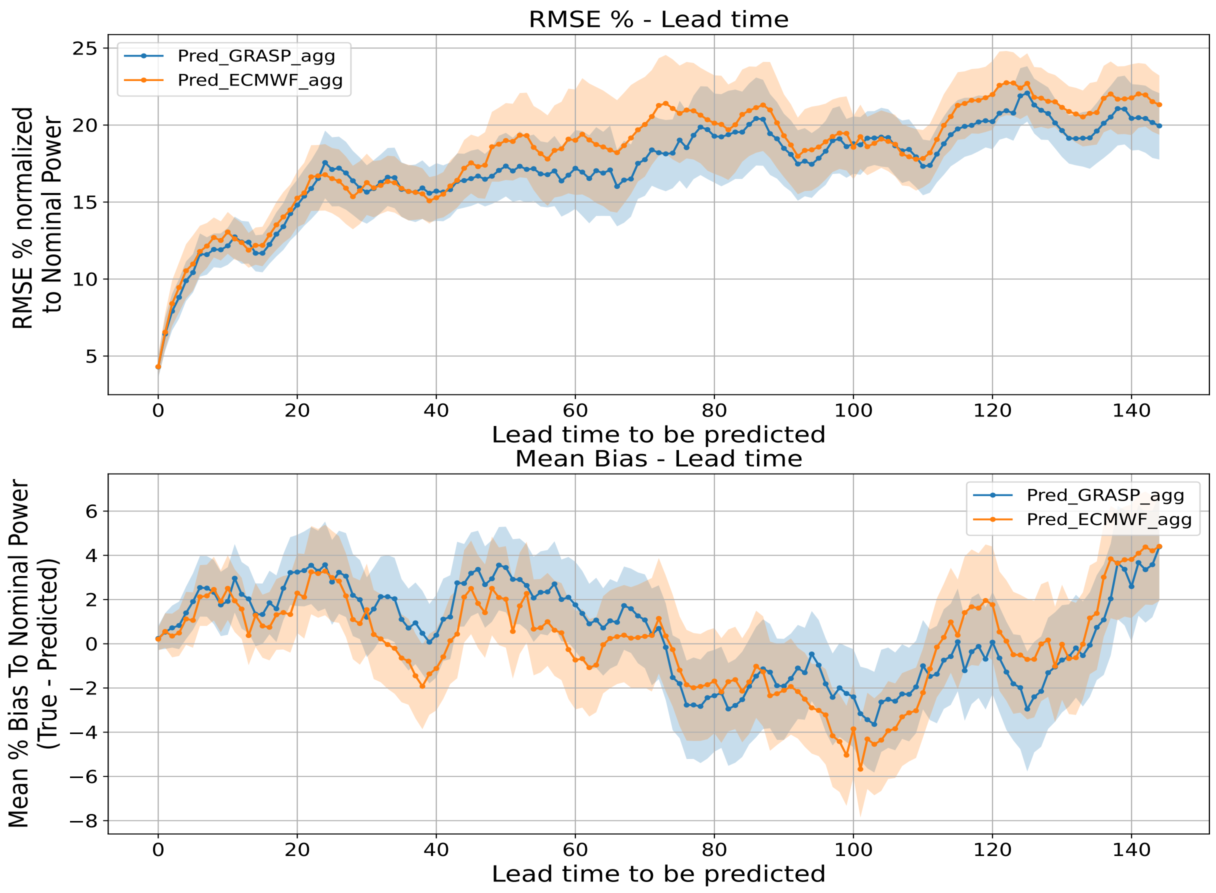


Figure 7.2: Top graph is the % RMSE at each lead time of the aggregate-power-predicting model. Bottom graph is the Mean Bias % with respect to nominal power. Both ECMWF and GRASP forecast datasets are evaluated as inputs to the aggregate model. The total RMSE percentage for all lead times on the test set is 17.66% for GRASP and 18.61% for ECMWF. GRASP forecast set provides an overall improvement of 1% throughout the whole day.

We can see a similar performance in the beginning compared to that of the persistence method. Basically our forecast is almost the persistence method in the early lead times since the last power measurement is the most relevant and dominant feature. One would expect an even better prediction than the persistence forecast but this amount of data did not allow the model to do much better than the already powerful persistence benchmark for the first lead times.

The two plots present similar metrics and results with the persistence benchmark before which are the RMSE and mean Bias at each lead time as a percentage of the nominal power. The predictions throughout the day however seem much more promising than those of persistence. We can see that the predictions from the two forecasting datasets agree to a large extend throughout the whole day. However in the period from lead time 40 until 90 and 110 until the end, GRASP seems to outperform the prediction capability of the ECMWF forecast dataset. The confidence intervals of the two predictions overlap in that period making it difficult to declare the best predictor by visual inspection. It has to be noted however that the ECMWF confidence intervals seem to be much wider in those periods, meaning that there is much greater variance in the predictions compared to GRASP forecast set.

The mean bias is deviating minimally around zero. We could say that the mean bias in the first half of the day seems to be lower for ECMWF compared to GRASP forecast data.

We can see that the prediction is not exactly smooth for either of the forecast datasets. The prediction looks a bit "noisy". In other cases where all of the lead times would be an input to a single model, one could suggest that the model suffers from high variance in the results and that it might be over-fitting to some extend. In our case each lead time is predicted by a separate model so this is not a continuous prediction. Additionally, each one of those models has measures such as early_stopping_rounds set, to rather conservative values(5) to prevent overfitting because of the variety of data that we have in our disposal.

7.3 *Individual-turbine-predicting model*

This section is dedicated to the results of the individual-turbine-predicting model in which every turbine is a different model. The concept, training and evaluation of the model have all been explained in sections 1.3,1.4 and 1.5 of the Methods chapter.

The features that were used here, are again the last power measurements and the wind speed. The wind direction was not used as a feature as it was not used in the aggregate-power-predicting model as well. The individual-turbines model did not improve when it was given the wind direction as a feature. It did not perform worse, it just did not improve as was expected. Since we had each turbine train separately each model would be capable of learning how the incoming wind direction could affect its performance. The reason behind this is that each turbine produces different amount of energy if the incoming wind stream interacts with our turbine before of after interacting with other wind turbines because of the wakes that are produced after each interaction. Turbine e.g. number 60 will experience much different winds when the wind comes from South-West or South-East. That is why it is even more surprising that the individual-turbine-predicting model does not benefit from the wind direction.

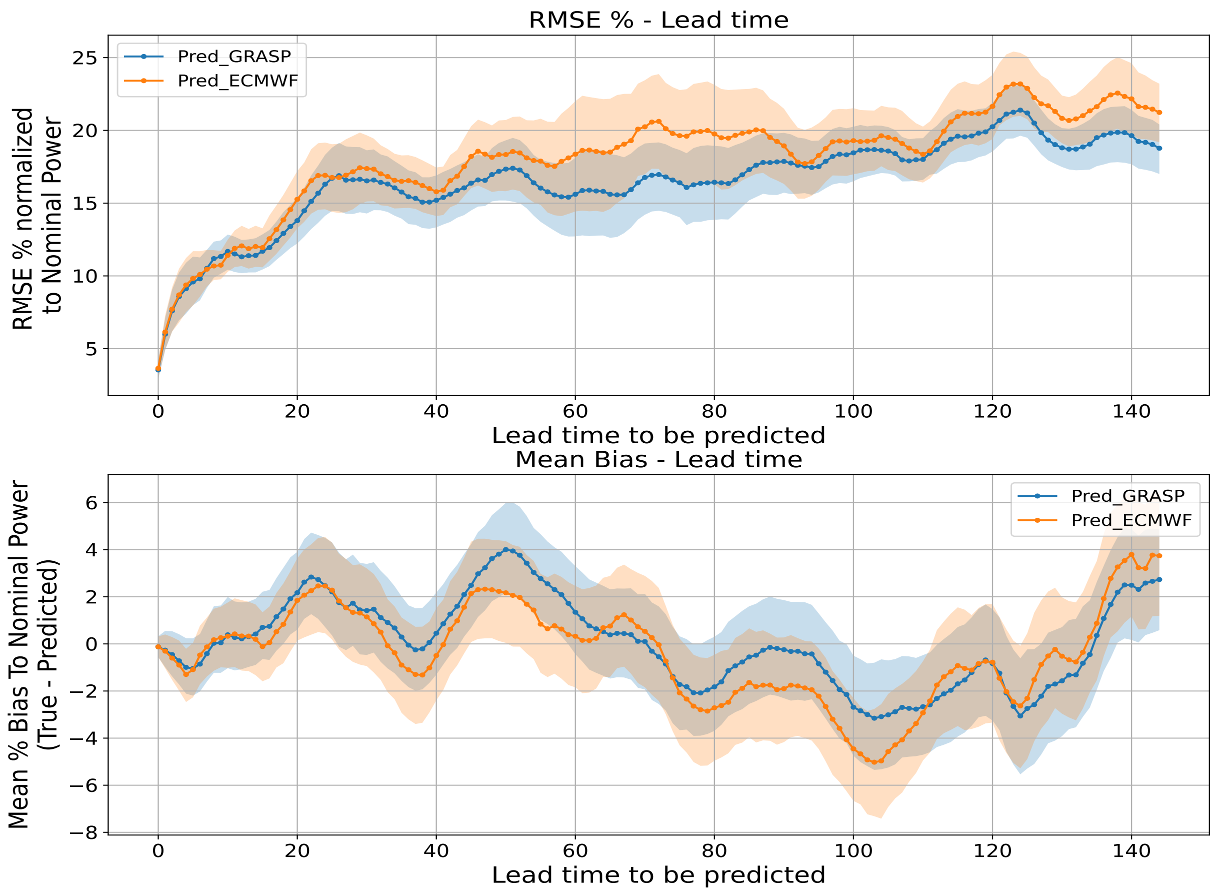


Figure 7.3: This is the % RMSE at each lead time of the individual-turbine-predicting model for both ECMWF and GRASP forecast datasets. The total RMSE over all lead times for GRASP is 16.86% while for ECMWF is 18.54 % on the test set. The 95% confidence intervals again overlap highly.

In Figure 7.3 we can see the performance of the two forecast datasets in the individual-turbine-predicting model. It is immediately visible that the RMSE % prediction loss for GRASP is lower across the whole day compared to the ECMWF prediction loss. The overall RMSE performance of GRASP dataset is 16.86% while the similar metric for ECMWF is 18.54% on the test set. The difference is now at almost 2%.

Apart from that we can see that the predictions are much more smooth. That is a direct result of the structure of the model. Building 111 distinct models for each lead time and then summing the results to get 1 aggregate power estimation results in the cancellation of individual errors that come with predicting each turbine. In a simple analogy each prediction is the actual value plus a small error sampled from a Gaussian distribution. The total prediction $\hat{y}$ is the sum of the individual turbines so: $\hat{y} = \sum_{i=1}^{1} 11 \left(y_i + \epsilon_i\right)$. With a big enough sample the error is cancelled out statistically and the resulting prediction is much less noisy. The aggregate-power-predicting setup in this case could be considered as a simple model from the summation that is very sensitive to each error that comes with it. The 2% increase and the RMSE loss being constantly lower for almost all lead times, are promising results.

The Mean Bias of the predictions of the two sets seem to be in good agreement with each other. That confirms the fact that GRASP uses the ECMWF forecast data for its boundary conditions. A slight periodicity can be observed in the bias of both predictions with a rough peak to peak distance of approximately 20 lead times(2 and a half hours) which I am unable to explain and could be random.

7.4 *Comparison of Individual and Aggregate models: Answering the 1st Research Question*

In this section we will compare the results from the previous two sections and see if there is a model that is superior to the rest.

Now that all of the results have been presented once, a combined plot would help to make the comparison easier. In Figure 7.4 we can see the persistence benchmark with its intervals, the aggregate and individual setups using both GRASP and ECMWF forecast data to predict the day ahead power. The purpose of this plot is to show that our forecast methods are always outperforming the persistence benchmark besides the very start of the model which is logical. The 95% confidence intervals of the persistence benchmark are also not overlapping with the confidence intervals of the rest of our forecasting methods. This makes it clear that all of them are superior to the simple benchmark we set to beat in the beginning.

The confidence intervals of our forecasting methods are overlapping to a large extend which together with the scale of the graph makes it really difficult to distinguish how they differ from each other. Figure 7.5 is a zoom in on the previous graph, excluding persistence and all confidence intervals. In this graph we can visually compare the performance of the forecasts.

The beginning of the forecast is similar in all models. After lead time number 40 there starts to be a differentiation up to almost lead time 90 and then again from 111 until the end. Within these periods we can see that the best performing models are the ones using GRASP forecast datasets as inputs. So, even though the aggregate and individual predicting model vary a lot in their structure and operation, it is the dataset that was used as input that made the difference. The two models with the highest loss at these periods both use the ECMWF forecast data. We can therefore conclude that the use of GRASP data over the use of ECWMF provides some advantage when forecasting, at least during those periods which in return drives the overall average loss to be lower.

To compare how different model setups perform we should see how the aggregate-power-predicting model performs against the individual-turbine-predicting model when using the same forecast dataset as input. Regarding the ECMWF dataset, the two predictions seem to be very close to each other and their predictive capability

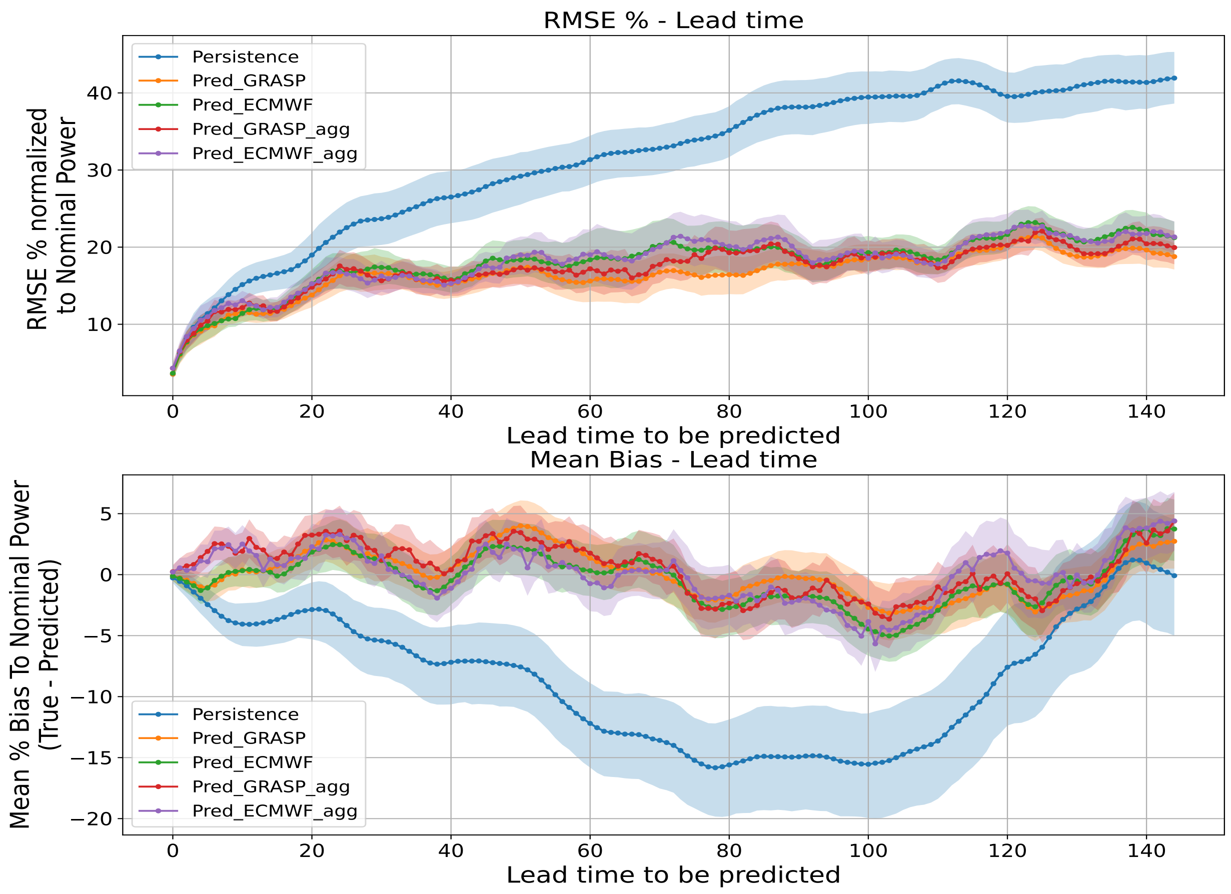


Figure 7.4: This figure contains all different prediction methods that were examined with their respective intervals. It clearly shows that all of our forecasts beat the persistence benchmark.

looks similar visually. By looking at the GRASP forecast data as input, we can see that the individual-turbine-predicting model seems to have a lower % RMSE loss that is not negligible compared to the aggregate-power-predicting model.

Overall day ahead % RMSE	Aggregate	Individual
ECMWF	18.61	18.54
GRASP	17.65	16.86
Persistence	32.85	

Table 7.1: Table holding the values of the best forecasts from all forecasts methods and forecast set combinations.

In table 7.1 we can see the overall % RMSE results throughout the day from all the different forecasts. The table also agrees with the visual assumptions made above. We therefore come to the following two conclusions:

- **The individual-turbine-predicting model, on average, performs better than the aggregate-power-predicting model when predicting the day ahead power at Anholt wind farm.**

- **The GRASP forecast data has, on average, a greater predictive capability compared to the ECMWF forecast data when used to predict the day ahead power at Anholt wind farm.**

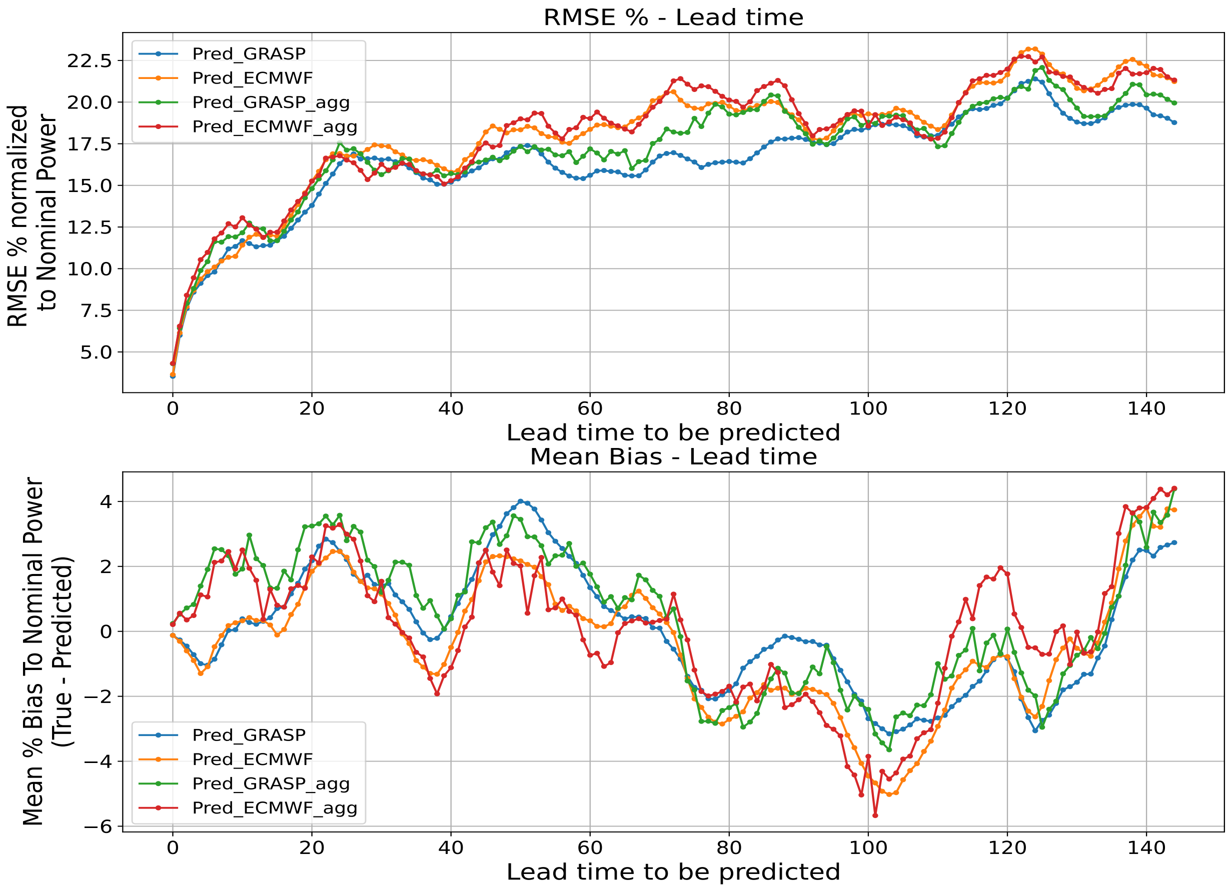


Figure 7.5: This graph shows the performance of all the forecast models in a single graph. We can see that the forecasts differentiate from lead time 40 almost up to the end. These splits are mostly responsible for the various performances of the models.

This section ends with two of our goals achieved. The first one regards the comparison of the two forecast datasets and the second one the first research question that we wanted to answer. We have managed to show that the use of the GRASP forecast dataset for predicting the day ahead power has more predicting potential than the ECMWF forecast dataset. The first research question we wanted to answer was if the use of an aggregate-power-predicting model is better than the use of an individual-turbine-predicting model. Our results suggest that an individual-turbine predicting model is able to perform better on average.

7.5 *Locating GRASP forecast data superiority: 2nd Research question*

The aim of this section is to see if we can find why the GRASP forecast dataset is performing better than the ECMWF foecast dataset for predicting the day ahead power.

Finding the reasons why the GRASP model produces forecast data that is better than the ECMWF forecast data at predicting the day ahead power at Anholt wind farm is a complicated process and beyond the scope of this thesis. A more complete approach would include finding the correlation or the effect of some of the internal physical processes in each model with the power production prediction ability. This is a very costly and time consuming process. Apart from that these forecast datasets are produced externally so there is no possibility to run such simulations to produce the data with different parameters etc.

What we can do is to check where the the forecast data from each of the forecasting models performs best. That can be done by checking where each forecast dataset performs best with respect to each quantity. This was done by splitting the range of the input value into a sensible amount of bins with a width of $quantity\_range/\#\_of\_bins$. Once that was done, the cases that belonged in each bin were grouped and their average predictive performance was calculated. Figure 7.6 shows how each setup and forecast dataset combination performed at each wind intensity.

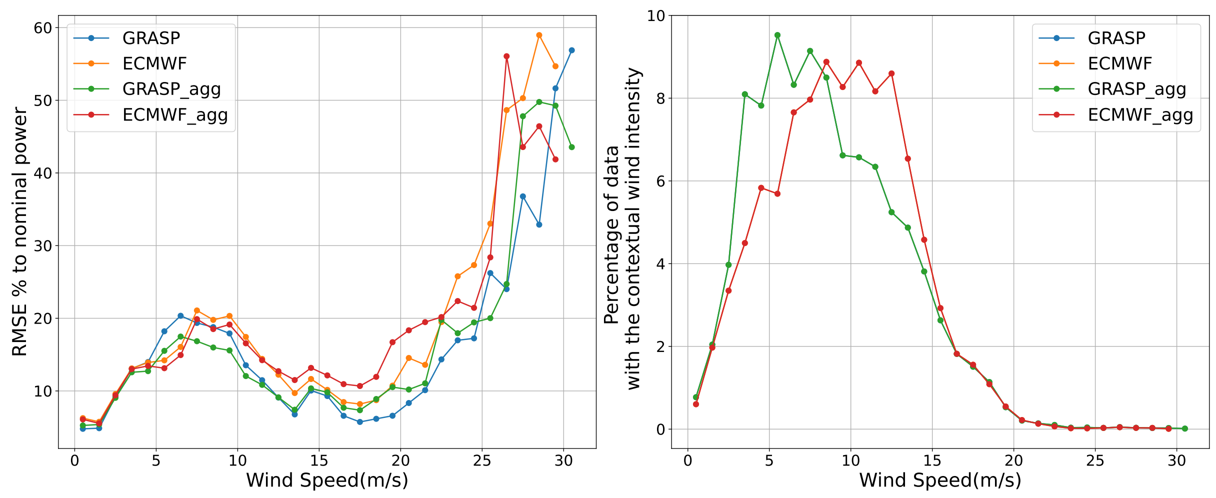


Figure 7.6: The left figure represents the performance at each wind speed and the right graph represents the percentage of data points in the evaluation period with the corresponding wind speed. The maximum wind was 35 m/s so this graphs contains 34 bins of 1m/s width. Values are centered at the middle of the bins and the wind speed for the two models on the right are overlapping since they are calculated in the same way exactly. The models using the GRASP forecast dataset seem to predict better. The frequency of each wind intensity and the mean agree, more or less, with that of Figure 5.3. These results are sampled from the forecast datasets.

The right graph gives a measure of what percentage of wind speeds each forecast model contains. The GRASP forecast dataset seems to contain more low intensity wind speeds(as they would be measured at the turbines), where the ECMWF forecast dataset contains slightly more higher wind speed predictions. One reason to why this might be happening could be that GRASP calculates the wind speeds within the farm by taking into account turbulence and the wakes that are introduced after the wind interacts with the turbines. These physical phenomena cancel out some of the wind's flow "laminarity" therefore resulting in more chaotic and less intense winds and consequently measured wind speeds.

The left graph shows how the each wind intensity affects the performance of each model. We see a great variety in performance throughout the wind speed range, partly because of the data points available at each range. However the overall behaviour among the models is more or less the same. The models performing the best are the aggregate and individual models using the GRASP forecast dataset. Therefore we see again that the specialized wind predictions from GRASP are doing a better job at predicting the power across the whole wind speed spectrum than the coarser ECMWF wind predictions.

Results in Figure 7.7 were sampled in a similar way, only this time RMSE is compared with respect to the aggregate power that was being produced every time. We can see in the right graph that low and full power production take up a large percentage of the probable production, roughly around 60%. [5]. We see that the model's performances with respect to the power output, differentiate mostly in the medium power output ranges where the amount of examples in each case is low. The performance of all the models seems to be similar with respect to the power output especially in the ranges where the examples were sufficient(low and high power outputs). Therefore we cannot really conclude that some forecast dataset has a better ability at predicting at some power output range than the other forecast dataset.

[5] Since there are 20 bins from 100 - 300 MW of an average 2% each one, that makes a total of 40% of the power produced being in medium power outputs.

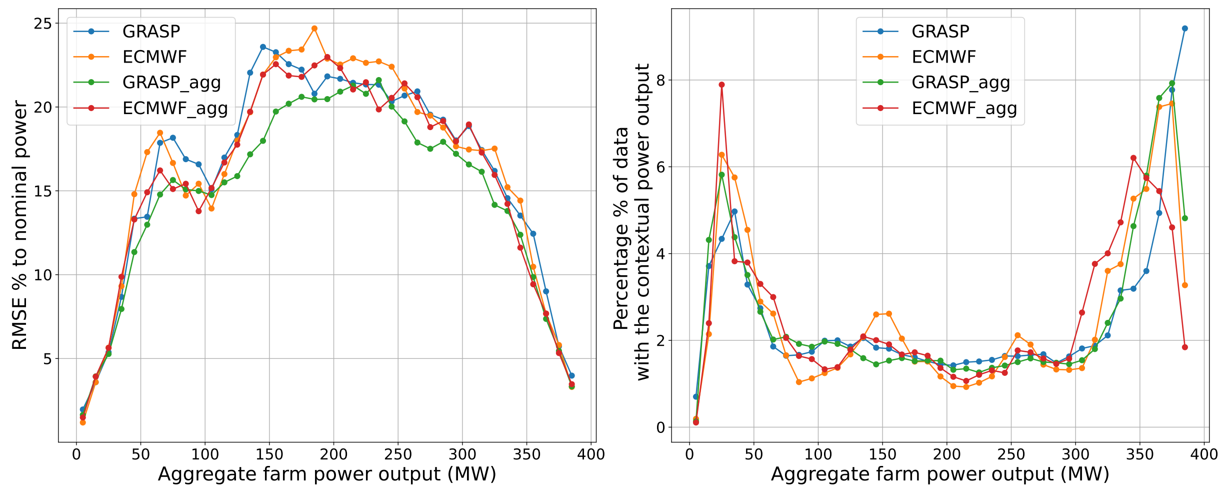


Figure 7.7: Similar plots as in Figure 7.6. The RMSE % is calculated with respect to the aggregate power production of the wind farm. The bin size is 10MW, the total number of points is 40 covering this way the output range of 400MWs. The best performing model is the aggregate model that is using the GRASP forecast dataset.

Even though wind speed values in both GRASP and ECMWF forecast datasets as seen in the left graph of Figure 7.6 look normally distributed around the mean, [6] the power output values are not distributed in the same way. That is because the power curve of wind turbines has the shape of a sigmoid function, therefore pushing the values of the power towards the extrema.

[6] The mean wind speed at Anholt wind farm is 8.8 as and the distribution of wind speeds can be seen in Figure 5.3).

GRASP and ECMWF could have additional spatial differences that the graphs above cannot encapsulate. That means that a model could be performing better than the other at predicting the power at the middle of the wind farm or at the edges. One such example would be that the GRASP dataset performs a bit worse in the boundary turbines of the wind farm because the turbulence that is calculated from the domain boundaries until the wind reaches the turbine is overestimated and the ECMWF's smoother forecast data is better in that.[7] On the other hand since turbulence plays an important role in the wind behaviour within the wind farm, GRASP forecast data could possibly have an advantage over ECMWF forecast data. Figures 7.8 and 7.12 are used to show the spatial difference of how the RMSE % and mean absolute Bias % behave spatially in the wind farm.

[7] Another reason that could be happening is because GRASP uses ECMWF as the boundary conditions for its model so a deviation from the truth in a smoother model would be doubly penalized in a simulation that strives to be more temporally and spatially detailed

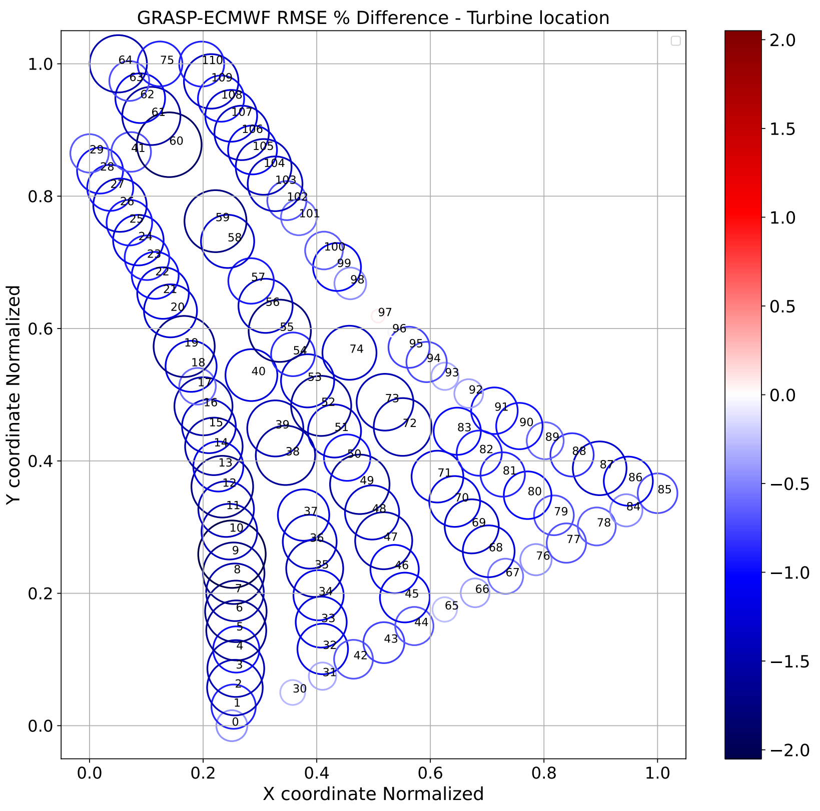


Figure 7.8: In this plot the GRASP RMSE minus the ECMWF RMSE percentages are presented. The RMSEs can be calculated across the turbine axis instead of the lead time axis in order to produce these results. This can only be calculated by comparing the individual prediction models. Red color means that the RMSE % of that turbine is higher for GRASP and blue is higher for ECMWF. The results indicate that across the whole wind farm, individual models predict better the day ahead power for each turbine when using the GRASP dataset instead of the ECMWF dataset. Only turbine 97 is red, although difficult to distinguish because the values are very close to zero.

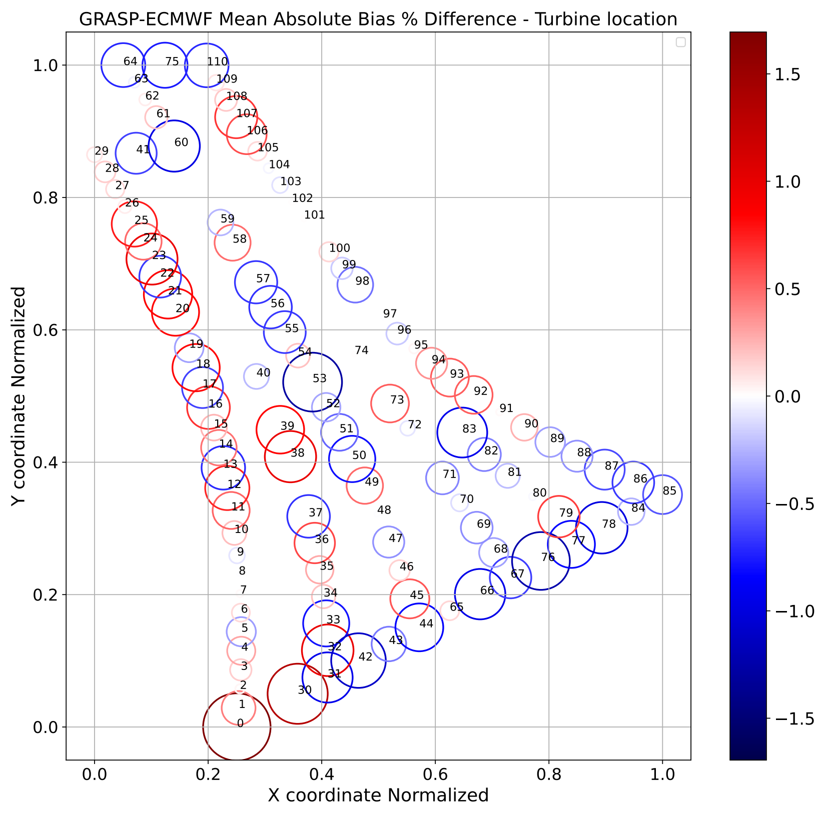


Figure 7.9: This plot shows the GRASP minus the ECWMF mean absolute biases % over the turbine axis. Since the bias originally includes both negative and positive values on its own, the absolute values of the biases had to be compared. This means that the red values indicate a higher absolute value for the GRASP forecast data at the respective turbine while a blue value indicates a higher absolute value for the ECMWF dataset. The results when comparing the two sets suggest that a lot of the turbines(very light color shade or white) have very similar absolute biases. Turbines in the first row seem to have a higher GRASP bias while the bottom row and middle row tend to have a higher ECMWF bias. Both of those behaviours could be explained by the example given above although more research is necessary.

7.6 *Summary and answer to Research Question 2*

In the last section we have tried to locate where or why the GRASP forecast dataset is superior to the ECMWF forecast dataset and argue about the performance of the two sets in a qualitative way. A summary of the previous 3 graphs will follow, that also constitutes the answer to Research Question 2.

In Figure 7.6, the GRASP forecast dataset is performing better in both the individual and aggregate models, throughout the whole wind speed spectrum. Since the models that were used to compare the two forecast datasets were identical, the difference in the performance has to do with the actual predictive ability of each forecast set at each wind speed. This means that the GRASP model has an overall advantage on predicting the localized wind speed over the ECMWF model at Anholt wind farm.

When comparing with respect to power the results are not as clear as before. The models in Figure 7.7 seam to perform equally well where the percentage of examples is sufficient enough. Since wind and power are connected through the power curve, one would expect that the performance of the GRASP dataset with respect to the power would have the same results as it did with respect to the wind. However in this case there is not a universal power curve that is used. Our models predict the power curve each time and the fact that there is a difference in the performance of the models with respect to the wind but not with respect to the power means that it is succeeding quite well in predicting a power curve that suits the needs of each lead time or turbine.

Figures 7.8 and 7.12 are produced with results from the individual models since the aggregate-power-predicting models do not produce predictions for each turbine. The RMSE % difference in the first figure shows that the GRASP forecast data is the undisputed winner when it comes to comparing the prediction performance of the individual model turbine-wise(spatially). The fact that in Figure 7.8, performance is dominated by GRASP while the bias results over turbine are mixed is because the two metrics are calculated and penalize in different ways(Bias is linear while RMSE is quadratic).

7.7 *Improving the individual-turbine-predicting model*

The best performing model for predicting the day ahead power that was created was the individual-turbine-predicting model that used the GRASP forecast dataset as input. This section is dedicated to improving that model.

7.7.1 *Feature selection*

The following list depicts the available features that we have from the GRASP model. We will try to find the combination that performs the best. The list of all available features is the following: [8]

1. *last known power measurements*
2. *rotor disk-average wind speed*
3. *rotor disk-average wind direction*
4. *free-stream wind speed*
5. *free-stream wind direction*
6. *rotor-disk average air density*
7. *Temperature*
8. *air density*
9. *inverse obuhkov length*
10. *friction velocity*

The first step however in improving a model is to make use of some feature selection technique. That usually means finding the features that will help the model predict in the best way. This process usually starts with some feature importance method that gives an insight to which features have the largest significance and effect on the performance of the model. To do that usually a model is created with all the features and then a tool like the Shapley Additive Explanations[9] could be used to explain which features are the most

[8] The first three features have been mentioned in the beginning of this chapter. They've been explained in section GRASP data preparation of the Data preprocessing chapter

[9] Scott Lundberg. Shap (shapley additive explanations), 2020

important. Since our individual-turbine-predicting model predicts the day ahead power by creating 111 models for each lead time(145 in total), it is impossible to somehow see with some mathematical tool which features would be the best for the complete model. That is because each turbine in each lead time has its own best features. Figures 7.10 and 7.11 show how two turbines [10] in the course of three different lead times(first, middle and last) change their feature preference. A model with all features for each turbine was created.

[10] Numbers 0 and 55. One is the southern-most turbine and the other one is in the middle of the farm so that it can represent most of the turbines and different physical phenomena.

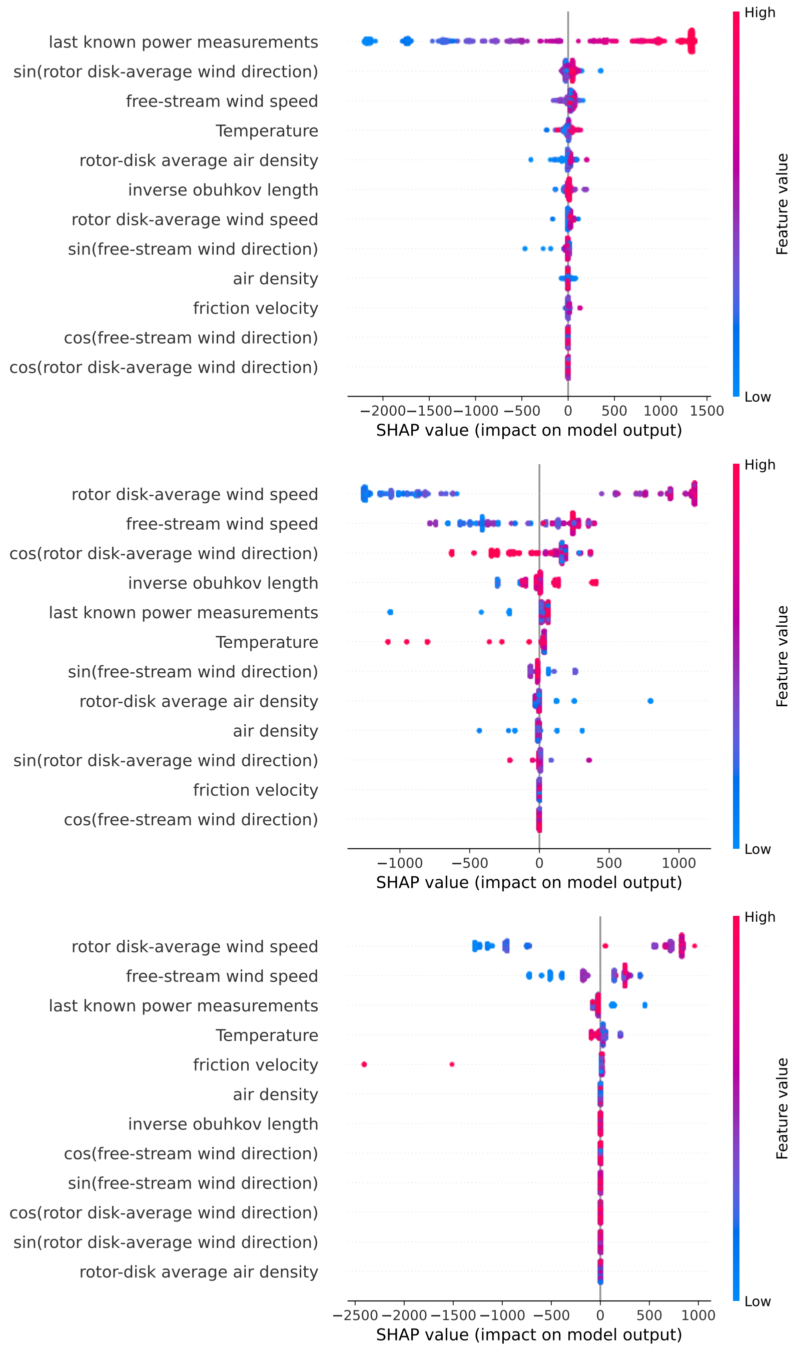


Figure 7.10: This is the Shapley additive explanation of turbine number 0 that is located at the bottom of the wind farm. The plots represent lead times 1,75,145 from top to bottom. We can see that the first lead time is dominated by the previously known power measurement feature. The 75th lead time uses the two wind speed prediction features as well as the cosine of the angle of the incoming wind. Near the end of the predicted day, at lead time 144 the model still relies on the wind speed prediction features. That is the case for wind turbine number 0.

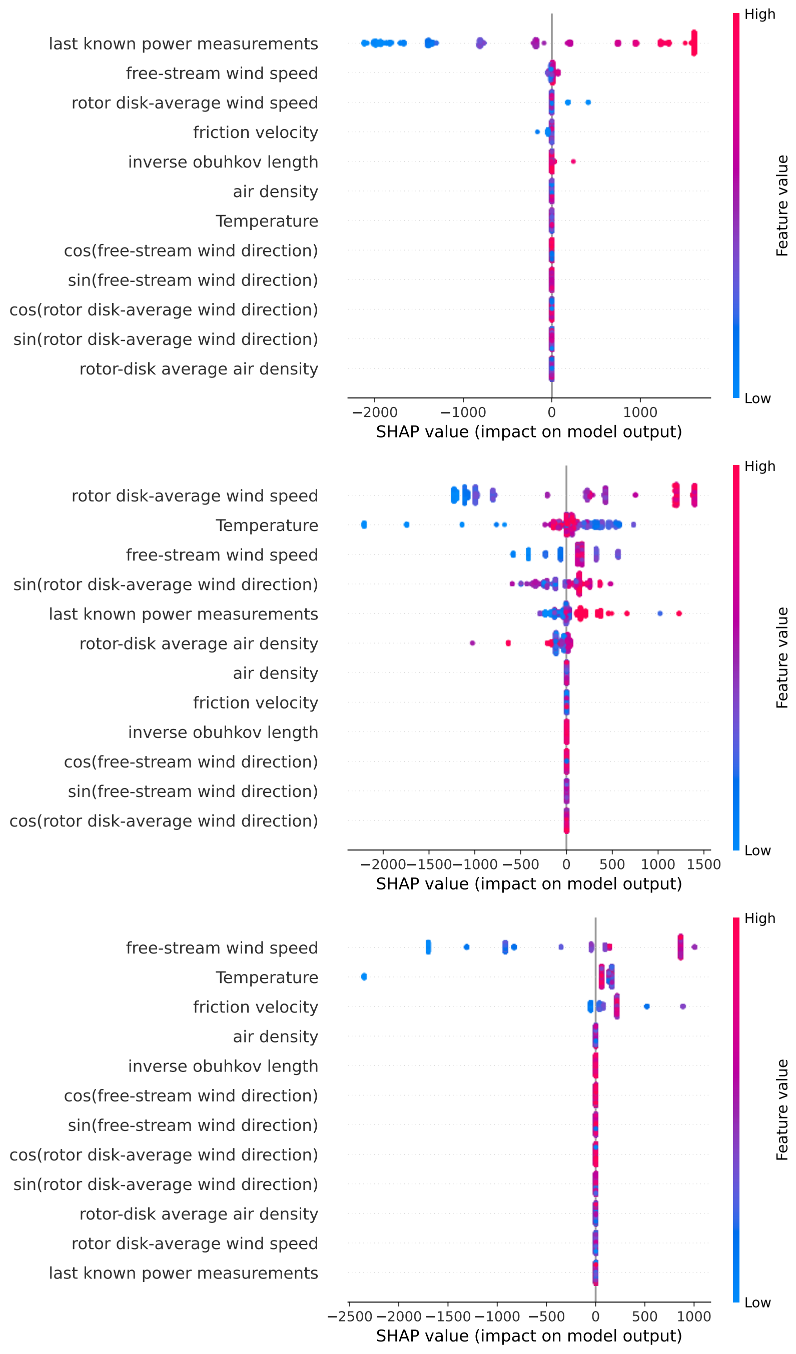


Figure 7.11: This is the Shapley additive explanation of turbine number 55 that is located in the middle of the wind farm and chosen so that it represents more turbines, probably. The plots correspond to lead times 1,75,145 from top to bottom. Top feature choice at lead time 0 is the last known power value with wind speed prediction having little to no importance. At the 75th lead time wind speed predictions, temperature and free stream wind define the models behaviour. At the last lead time wind, temperature and friction velocity are the most important ones. Notice how friction velocity had absolutely no importance in the 75th lead time.

The conclusion is that there is not an easy way to find which features are the most important for a complete day ahead prediction since each turbine in each lead time finds different features to have different importance. We can be certain though that the last power measurement and some wind speed prediction are necessary for the models to do well. The approach at this point will be to try which features maximize the complete prediction to see if we can get an improvement through that. In order to find the optimal number or type of features, we would have to try different combinations of features and numerically evaluate for the complete day ahead prediction model.

In the beginning of this chapter we talked about how the first three features of the numbered list were common in both GRASP and ECMWF datasets and how all of them would be used in the training and evaluation of the two sets. The best result the individual-turbine-predicting model achieved with using the first two features was an overall RMSE of 16.71% on the validation set for the complete day ahead prediction. We will be using the validation set since we want to find the best features still and then test on the test set. Next attempt is to try and see if each feature can bring some extra value to the overall prediction. Table 7.2 shows how each of the features individually improve or don't improve the overall day ahead prediction. The features that bring value to the prediction of run # 1 without any further engineering are *air density* and *friction velocity features*. Those were included together in run # 11 and produced the lowest result of an average RMSE % of 16.64 on the validation set. The test set gave an overall prediction of 16.79%. That is also an improvement from the results of Table 7.1.

Run	Features' numbers used in training	Overall day ahead % RMSE
1	1,2	16.71
2	1-10	17.25
3	1,2, sin(3),cos(3)	17.11
4	1,2,4	16.72
5	1,2, sin(5),cos(5)	17.10
6	1,2,6	17.67
7	1,2,7	16.86
8	1,2,8	16.68
9	1,2,9	17.31
10	1,2,10	16.69
11	**1,2,8,10**	**16.64**

Table 7.2: This table holds the overall RMSE% performances in the day ahead prediction of different combinations of features. Since run #1 gave a better result than all of the input features together in run #2 the next thing to try would be to find which feature if added to features of run #1 can bring value to the overall performance. The extra features that did better than run #1 were features 8 and 10, namely *air density* and *friction velocity*.

7.7.2 *Introducing aggregation weights*

So far we have been predicting the power at each individual turbine and then summing up the results to get the aggregate power value. There is also the possibility of adding an extra learning layer in between that can be trained to summarize the individual results in a weighted manner, which should theoretically result in some improvement. The type of learning model that will be used in this case is a simple neural network that has 111 inputs and 1 output layer exactly like the one in Figure 4.1. In this case the model has 111 weights to tune plus a bias parameter that we included as well.

The model weights of the turbines at each lead time have to be initialized with all being 1 because otherwise the algorithm is keeping the initial weights and just increasing the weights that were positive and decreasing the initial weights that were negative until the sum of the individual predicted weights reached the desired aggregate weight. We also wanted the algorithm to find which turbines are the most reliable predictors or not and increase their weight accordingly. One different such model will be trained for each lead time. The overall RMSE result when using this extra layer on top is 16.75 %. This is an improvement of 0.11% in the overall RMSE performance form the performance of the model at Table 7.1. This results regards the test set as we have not tuned the hyperparameters of this model. However it does seem to improve a little bit on the performance which suggests that it could maybe improve more when tuned properly. The following Figure shows how the weights were initialized and how the model decided to tune them so that it predicts the aggregate power better.

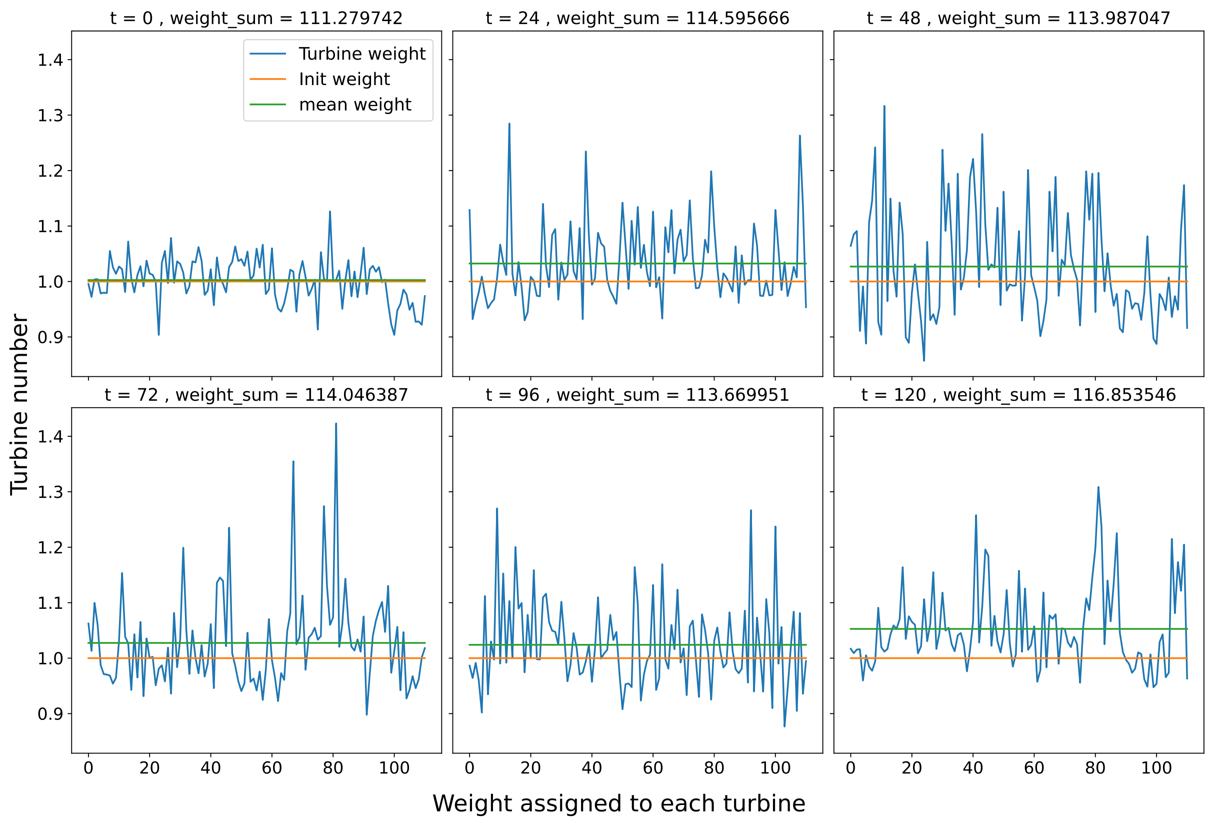


Figure 7.12: Turbine weights for 6 different lead times. The blue line represents the final weights of the turbines the model came up with. The orange line represents the initial weights and the green line represent the sum of the weights. The bias is not included in the summation.

We can see that the sum of weights is almost always above 111 which would be the sum of weights if all turbines had a weight of 1. This suggests that the individual-turbine-predicting model might be under-predicting the total power eventually and that a weighted summation might bring some value to the prediction.

7.8 *Conclusions and future perspectives*

In this work we aimed to showcase whether the use of the GRASP forecast dataset over the use of the ECMWF forecast dataset improves the day ahead prediction of the wind power production at Anholt wind farm. The results suggests that there is an overall benefit in the day ahead prediction of close to 2% of the best model using the GRASP forecast dataset from the equivalent model using the ECMWF forecast dataset.

During this process we have also managed to answer our first research question which was to show that the use of an individual-turbine-predicting model is more beneficial over the use of an aggregate-power-predicting model. In all cases when the same forecast data was used to compare the two models, the individual-turbine-predicting model outperformed the aggregate-power-predicting model. The difference when using the GRASP forecast data was $\approx 0.8\%$ and when using the ECMWF as the forecast data the difference was $\approx 0.01\%$.

Additionally, in the second research question we had to show where or why the GRASP forecast dataset was better than the ECMWF forecast dataset. The findings suggest that the GRASP forecast dataset gives more accurate wind predictions[11] at the turbine level than the ECMWF forecast dataset since our model was predicting better throughout the whole wind speed spectrum. There seems to be a clear advantage also spatially. Each turbine predicted the power much better when using the GRASP forecast set as input. This result suggests also spatial apart from temporal superiority.

[11] Or at least its predictions are more easily connected to the actual power values by our algorithm.

Lastly, we set out to improve the performance of the individual-turbine-predicting model which was the top performing model among all. The model performed an average RMSE % of 16.86 on the test set initially. By trying different features we found the best combination and managed to bring the average RMSE down to 16.79 % on the test set. By adding an additional weighted summation layer between the individual predictions and the aggregation of the individual results we managed to further improve the result to 16.75 % overall RMSE performance. This is an 0.11 % improvement of the overall RMSE.

When improving the final model there are a lot of things to consider still. An extensive Shapley additive explanation could be performed to find the features that most of the turbines seem to not benefit from at all. The impactful features could be used to check whether there would be any improvement. Another, more specialized method, would be to check whether we could supply each turbine model in each lead time with the most important features for it. In my opinion this would improve the performance from the feature selection perspective in the best way. Additionally, a new hyperparameter optimization could be performed in case the models with the new amount of features had a different parameter preference than the hyperparameter optimization performed in the Methods chapter. Also the weighted sum neural network model could be improved with a hyperparameter optimization. Lastly, a model that combined both the features from the ECMWF and GRASP forecast data could be created to see if that would bring further value on improving the day ahead power prediction at Anholt wind farm.

8 Bibliography

3.1 Product Schedules - Forecast User Guide - ECMWF Confluence Wiki. URL `https://confluence.ecmwf.int/display/FUG/3.1+Product+Schedules`.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012. ISSN 15324435. URL `http://scikit-learn.sourceforge.net`.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-Parameter Optimization. 2011.

Jerry Diakiw. Windmills of mykonos, October 2008. URL `https://www.flickr.com/photos/yaroslawd/11317142373`.

D. Djairam, A.N. Hubacz, P.H.F. Morshuis, J.C.M. Marijnisen, and J.J. Smit. The development of an electrostatic wind energy converter (ewicon). In *2005 International Conference on Future Power Systems*, pages 4 pp.–4, 2005. DOI: 10.1109/FPS.2005.204208.

Bradley Efron. *The Jackknife, the Bootstrap and Other Resampling Plans*. Society for Industrial and Applied Mathematics, jan 1982. DOI: 10.1137/1.9781611970319.

Alexander Elvers. Hyperparameter optimization using random search own work, cc by-sa 4.0. `https://commons.wikimedia.org/w/index.php?curid=84255403`, `https://commons.wikimedia.org/w/index.php?curid=84255404`, `https://commons.wikimedia.org/w/index.php?curid=84255405`, November 2019. Three images included in the same citation.

Ciaran Gilbert, Jakob W. Messner, Pierre Pinson, Pierre Julien Trombe, Remco Verzijlbergh, Pim van Dorp, and Harmen Jonker. Statistical post-processing of turbulence-resolving weather forecasts for offshore wind power forecasting. *Wind Energy*, 23(4):884–897, 2020. ISSN 10991824. DOI: 10.1002/we.2456.

T. Heus, C. C. Van Heerwaarden, H. J.J. Jonker, A. Pier Siebesma, S. Axelsen, K. Van Den Dries, O. Geoffroy, A. F. Moene, D. Pino, S. R. De Roode, and J. Vilà Guerau De Arellano. Formulation of the Dutch Atmospheric Large-Eddy Simulation (DALES) and overview of its applications. *Geoscientific Model Development*, 3(2):

415–444, 2010. ISSN 1991959X. DOI: 10.5194/gmd-3-415-2010. URL www.geosci-model-dev.net/3/415/2010/.

By Isjc99. Typical power curve of a wind turbine with cut-in,area of rated power and cut-out point. https://commons.wikimedia.org/wiki/File:Powercurve.png, February 2012.

Scott Lundberg. Shap (shapley additive explanations), 2020.

Georges Matheron. Traite de geostatistique appliquee. Tome I, 1962.

Jakob W. Messner, Pierre Pinson, Jethro Browell, Mathias B. Bjerregård, and Irene Schicker. Evaluation of wind power forecasts—An up-to-date view. *Wind Energy*, 23(6):1461–1481, 2020. ISSN 10991824. DOI: 10.1002/we.2497.

Johan Meyers and Charles Meneveau. Large Eddy Simulations of large wind-turbine arrays in the atmospheric boundary layer. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2010. ISBN 9781600867392. DOI: 10.2514/6.2010-827. URL https://arc.aiaa.org/doi/abs/10.2514/6.2010-827.

Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014. URL https://github.com/fmfn/BayesianOptimization.

Ricardo A. Olea. Geostatistics for Engineers and Earth Scientists. *Technometrics*, 42(4), nov 2000. DOI: 10.1080/00401706.2000.10485748.

David Opitz and Richard Maclin. Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999. ISSN 10769757. DOI: 10.1613/jair.614.

Henk Polinder and Maxime Dubois. Generator systems for wind turbines. *Proceeding PCIM'03*, (May), 2003. URL https://www.researchgate.net/publication/262643332.

Hannah Ritchie and Max Roser. Energy. *Our World in Data*, 2020. https://ourworldindata.org/energy.

Conor Sweeney, Ricardo J. Bessa, Jethro Browell, and Pierre Pinson. The future of forecasting for renewable energy, 2020. ISSN 2041840X.

Ørsted A/S. Djursland anholt offshore wind farm. URL https://orstedcdn.azureedge.net/-/media/www/docs/corp/com/our-business/wind-power/wind-farm-project-summary/anholt_uk_2018.ashx?la=en&rev=3b2f175b341e4ca2b906bf2a2ff68c8f&hash=51B4609E53A631D29E0EF49936C33A84.

M. P. Van Der Laan, A. Pena, P. Volker, K. S. Hansen, N. N. Sorensen, S. Ott, and C. B. Hasager. Challenges in simulating coastal effects on an offshore wind farm. In *Journal of Physics: Conference Series*, volume 854. Institute of Physics Publishing, jun 2017. DOI: 10.1088/1742-6596/854/1/012046.

David J. Yáñez. Viv resonant wind generators. 06 2018.