UiT The Arctic University of Norway

Faculty of Science and Technology

Department of Physics and Technology

Wind Power Prediction with Machine Learning Methods in Complex Terrain Areas

Brynhild Bentsen Sæther

EOM-3901 Master's thesis in Energy, Climate and Environment

June, 2021

Abstract

The increasing amount of intermittant wind energy sources connected to the power grid present several challenges in balancing the power network. Accurate prediction of wind power production is identified as one of the most important measures for balancing the power network while maintaining a sustainable integration of wind power in the power grid. However, the volatile nature of wind makes wind power forecasting a complicated task, and it is known that the performance of already established wind power prediction models decreases for wind farms in complex terrain sites. This thesis aims to forecast the future wind power output for five different wind farms in Northern Norway using methods from statistics and machine learning. The wind farm sites are generally characterized as complex terrain areas with good wind resources. Four different prediction models are developed for short to medium-term, multi-step prediction of wind power, ranging from traditional statistical models such as the ARIMAX process to complex machine learning models. Additionally, two of the models are implemented both using the recursive and the direct multi-step forecasting technique. For each wind farm, the models are evaluated for an entire year and utilize multivariate input data with variables from a NWP model. The results of the experiments varied greatly across all locations. It was seen that the implemented models were outperformed by the persistence model for short forecasting horizons. However, when the forecasting horizon increased, several models showed a lower error than the persistence model.

Acknowledgements

First and foremost a big thank you is dedicated to my supervisor Stian. Without your help and guidance this thesis would hardly be completed. Thanks also to Yngve Birkelund for providing the dataset that was used in this project.

Sincere gratitude and appreciation is directed to my classmates through five years. Through our years of studying we have proved that we work together just as well as we pause together. It has been a pleasure, and the coffee breaks will be missed. A special thanks is directed to my partner in crime, Vilde Jensen. Without you these five never would never have been the same.

Lastly, I want to thank my family and my boyfriend Andreas for your invaluable support through my studies.

Brynhild Bentsen Sæther
Tromsø, 2021

Contents

List of Figures

List of Tables

Abbreviations

ACF	autocorrelation function
ADF	Augmented Dickey Fuller
ANN	artificial neural network
AR	autoregressive
ARIMA	autoregressive integrated moving average
ARIMAX	autoregressive integrated moving average with exogenous variables
ARMA	autoregressive moving average
CNN	convolutional neural network
KPSS	Kwiatkowski, Phillips, Schmidt and Shin
LSTM	long-short term memory
MAE	mean absolute error
MEPS	MetCoOp Ensemble Prediction System
MIMO	multiple input multiple output
MSE	mean squared error
NRMSE	normalized root mean squared error
NVE	Norwegian Water Resources and Energy Directorate
NWP	numerical weather prediction
PACF	partial autocorrelation function
RBF	radial basis function
RF	random forest
RMSE	root mean squared error
RNN	Recurrent neural network
SVR	support vector regression
TCN	temporal convolutional network

Part I

Background

1

Introduction

Underpinned by population growth, economic development, and increasing prosperity levels in emerging economies, the overall global energy demand is projected to reach an increase of 50% by 2050 (US Energy Information Administration, 2019). At the same time, the United Nations Environment Programme states that greenhouse gas emissions must begin to fall by 7.6% each year starting from 2020 to limit global warming to 1.5 °C by 2030, as targeted in the Paris Agreement (United Nations Environment Programme, 2019). To put this number into context, the global lockdown of the world due to the COVID-19 pandemic in 2020 with travel bans and economic slowdowns is only projected to reduce global emissions by 6% this year (Le Quéré et al., 2020). As a result, the installed capacity from renewable energy sources in today's energy system must not only meet the increasing energy demand but, at the same time, displace the role of fossil fuels already accounted for in the energy mix. Fortunately, global efforts on reducing carbon emissions are steadily increasing, and renewable energy generation is on the rise. While hydropower remains the largest renewable source of electricity worldwide, power from solar and wind are today more cost-competitive than the building of new coal or gas power plants in about two-thirds of the world (United Nations Environment Programme, 2019). Hence, intermittent energy sources from wind and solar power generation will play a leading role in accelerating the decarbonization of the energy system.

The global cumulative wind power capacity exceeded 651GW worldwide in 2019, making it the second-largest renewable energy source after hydropower

(BP p.l.c., 2020). Unfortunately, the intermittent nature of wind presents several challenges to wind energy operations. The non-steady mechanical load yields excessive wear in a turbine's drive train and makes the wind turbines prone to fatigue failures. Meanwhile, the randomness in wind power output makes it difficult to accommodate a substantial wind power level in the power grid. Progress in wind turbine technology also enables the design and installation of larger turbines at locations where the wind is more intermittent (Ding, 2019). To speed up the energy transition, a continued focus on solutions that support the integration of wind and other intermittent renewable energy sources to the grid is vital for further development of intermittent renewable energy sources (Lee & Zhao, 2020).

1.1 Motivation

The electric power system forms the connection between energy suppliers and consumers, where the essential function is to meet the energy demand of consumers. Utilities must handle demand and electricity production variations on both long and short timescales to maintain a secure and reliable power system operation. Generally, the power system operation divides into different timescales ranging from seconds to days to balance supply with demand. On a short timescale, load variations during seconds to minutes are handled by automatic regulation of a generator's speed (governor action). On a longer timescale ranging from approximately 10 minutes to several hours, usually referred to as 'load following,' balancing the network involves connecting or disconnecting dispatchable power sources for the purpose of balancing the anticipated load increase or decrease. The planning of generation at this time scale is known as generation scheduling. As wind energy storage is not yet feasible on a large scale, utilities must employ a mix of dispatchable generation assets that can be controlled to deal with unexpected demand fluctuations or loss of generation (Infield & Freris, 2020). High penetration of wind energy in the power system may also threaten the system's stability. If a significant power plant suddenly fails, utilities must have enough flexibility in their systems to compensate for the lost capacity to avoid a system-wide shutdown (M. R. Milligan, Miller, & Chapman, 1995). As a consequence, utilities must plan for operating reserves to maintain a reliable system in case of sudden generation loss or demand prediction errors (Yoder, Hering, Navidi, & Larson, 2014). The more reserves required for a balanced power system, the higher the operating costs will be. Hence, the introduction of renewable energy sources into the electrical power system will impact and incur costs regarding balancing the network and maintaining a reliable system (Infield & Freris, 2020).

A measure to reduce the incurred costs of wind power connected to the power

grid is to forecast future wind power output. The importance of reliable wind power forecasts was first identified in the late 1970s, indicating that it could have applications in maintenance scheduling, load scheduling strategies, and dispatching decisions depending on the forecast horizon (Costa et al., 2008; Wendell, Wegley, & Verholek, 1978). Long-term forecasts, ranging from weeks to months and years, can be used for maintenance scheduling for the system operator. Medium-term forecasts on a daily timescale of hourly wind levels can be factored into the generation scheduling strategy, while short-term forecasting of expected power output from a wind farm ranging from minutes to hours can be applied to decision making for energy trading and dispatching (Costa et al., 2008). Accurate short-term wind power forecasts may help utilities reduce or avoid the need for excess generation and, therefore, make the integration of wind power into the grid more sustainable and cost-effective. With perfect foresight, the utilities can plan the generation schedule accordingly, eliminating the need for backup generation resources to compensate for the uncertainty in wind intermittency (Xiaoyun, Xiaoning, Chao, Shuai, & Xiuda, 2016; Botterud, Wang, Miranda, & Bessa, 2010). Inaccurate forecasts, however, may cause problems to utilities. If the forecast is too optimistic, the backup generation might be too low to maintain an acceptable power level in the grid. In such a situation, regulators' minimum reliability requirements may be too low, which causes system costs to be higher than optimal (M. R. Milligan et al., 1995).

A vast amount of research has been done in the field of wind power forecasting, as will be discussed in chapter 2. However, using standard error metrics such as RMSE and MAE, it has been shown that the performance of popular methods decreases for wind sites in complex terrain (Costa et al., 2008). It follows that accurate wind power prediction is site-dependent regarding the local wind profile, terrain type, and climatic conditions. As a result, finding a universal method for forecasting future wind power output is a challenging task.

In this thesis, short-term wind power forecasting is done for five different wind farms located in Northern Norway. This region can be characterized as a cold climate region, with good wind resources in a complex terrain consisting of high mountains, valleys, and fjords (Byrkjedal & Åkervik, 2009).

1.2 Objectives

The overall objective of this master thesis is to contribute to the improvement of forecasting techniques for wind power prediction in complex terrain by using advanced forecasting techniques from statistics and machine learning. The overall objectives will be achieved by:

1. Implementing several algorithms from time series forecasting, ranging from the naive persistence model via classical statistical methods (ARIMAX) and shallow machine learning models (random forest (RF) and SVR) to deep-learning with artificial neural networks (LSTM)

2. Comparing the performance of different methods on available datasets of produced wind power from five separate wind farms located in northern Norway.

3. Comparing how the relative merit of the implemented methods depends on and changes with the prediction horizon, to test the hypothesis that simple models will perform well at the shortest prediction horizons whereas the more advanced models will show their strength when the prediction horizon increases.

4. Evaluate whether the RF and the SVR models perform better when implemented recursively or when the predictions are made directly for the targeted horizon.

The experimental design will apply theory from machine learning for training, validating, and testing the implemented models. The models will receive measured historical power output data and forecasted weather data from a NWP model as input, including variables such as wind speed, wind direction, temperature, and surface pressure from each of the wind farm locations. Data is provided by the Norwegian Water Resources and Energy Directorate (NVE) and the Norwegian Meteorological Institute.

1.3 Contributions

Previous works on this exact dataset is limited to a master thesis from 2019 (Fossem, 2019), where Marcov Chain modelling were used for the 2-hours ahead prediction of power output at Raggovidda, Kjøllefjord, Havøygavlen, Fakken and Nygårdsfjellet wind farms. The contributions of this master thesis includes:

1. The evaluation of the performance of four different time series forecasting models involving both classical statistical methods, and more complex machine learning methods and one deep learning model, for this dataset.

2. Short- to medium-term forecasting of wind power output at Raggovidda, Kjøllefjord, Havøygavlen, Fakken and Nygårdsfjellet wind farm with prediction horizons ranging from 1-hour to 24-hours ahead.

1.4 Structure of the Thesis

The thesis is divided into four parts: Part I Background, part II Theoretical Background, part III Method, and part IV Results and Discussion. In Part I, the introduction is provided in chapter 1 and previous works on short-term wind power forecasting will be reviewed in chapter 2. Part II provides the theoretical background information for the thesis. In chapter 3, wind power characteristics and the workings of the European power market are introduced. Chapter 4 covers the theory of time series forecasting, focusing on both statistical and machine learning approaches. In chapter 5, the forecasting models that will be used in this thesis are explained. In part III an analysis of the datasets is provided in chapter 6. The preprocessing of the dataset is described in chapter 7, and the methodological decisions made when implementing the forecasting models are outlined in chapter 8. In part IV, the experimental setup and the results from the experiments are given in chapter 9. Lastly, a discussion of the results and a conclusion are provided in chapter 10.

2

Previous Works

Forecasting models for wind power prediction are widely studied, and several state-of-the-art techniques have been identified over the years (Foley, Leahy, Marvuglia, & McKeogh, 2012). Many of the existing power prediction systems used are based on the results of NWP models. A NWP model is a mathematical model that predicts the weather in near future based on the current weather conditions and mathematical models of the atmosphere and oceans. Typically, the models including NWP model data as input are more accurate than those that do not. Hence, all models employed by utilities utilize an approach that includes a NWP model (Giebel & Kariniotakis, 2017). In this chapter, the different methods typically used for wind power forecasting will be introduced, and an overview of previous works on short-term wind power prediction will be given. Given that this thesis will be focusing on the implementation and performance of statistical and machine learning methods for wind power forecasting, the literature review will be focusing on works that use statistical and data-driven methods. As this field is constantly developing, the literature review will concentrate on relatively new papers, excluding older research.

2.1 Prediction Methodologies

Wind power forecasting can be categorized according to the timescale it is covering and which methods are being utilized. Timescales range from short-term, via medium-term to long-term forecasting as described in chapter 1, and the

different methods utilized can be divided into persistence, physical, statistical, machine learning, and hybrid methods. The main differences between the methods lie in the varying level of complexity between models, the required input data, and their recorded accuracy at different timescales.

2.1.1 Persistence Method

The persistence model is typically used as a baseline or reference model. It is based on the simple assumption that the wind speed at time $t + n$ will be the same as it was at time t. This method's accuracy has usually been high for very short-term prediction of wind power, on a timescale of minutes to hours, because of the volatile nature of wind data. However, as the prediction timescale increases, the accuracy of the method quickly deteriorates. Apart from being simple, the main advantage of this method is that no tuning or evaluation of parameters or external variables are required (Hanifi, Liu, Lin, & Lotfian, 2020).

2.1.2 Physical Methods

Physical models use the concepts of the lower atmospheres dynamics and meteorology to carry out spatial refinement of the output of NWP models to the specific on-site conditions of a wind farm, as well as the transformation of the predicted wind speed to the hub height of wind turbines. The predicted wind speed from the NWP model is usually given on a relatively coarse spatial resolution. Consequently, the refinement of the NWP model output must consider several factors from the surrounding area of the wind farm, such as the surface roughness, shelter from obstacles, and factors such as temperature and pressure for calculation of the wind speed in that exact area. The wind speed at hub height is then calculated by parametrization of the wind profile or flow simulations. Lastly, the power output of a turbine or the wind farm is determined by the estimated wind speed at hub height and the manufacturer's wind power curve(Lange & Focken, 2006). The wind power curve is an estimation of the power output of a turbine as a function of the wind speed, and it will be further explained in chapter 3. Physical models usually have high accuracy and can improve modeling of wind flow in complex terrain because they consider the surroundings of the wind turbine or wind farm (Wu & Hong, 2007). Physical methods do not need to be trained on historical data, but are dependent on a detailed description of the terrain surrounding the wind farms.

2.1.3 Statistical and Machine Learning Methods

Statistical and machine learning methods aim to learn the linear and non-linear relationships between the data from an NWP model, such as the wind speed, direction of the wind, temperature, and the generated power from a wind turbine. These approaches generally do not use a pre-defined power curve to determine the power output, but instead use the results from the NWP model as input to a mathematical model. Historical data and dependent variables are used for training the model, and the model is then tuned by comparing the predictions with the on-line measured data. The most common statistical approach for short-term wind power prediction is the autoregressive moving average (ARMA) models, including their extensions and special cases. These models have shown relatively good performance for short-term predictions, but the accuracy tends to decrease as the forecasting horizon increases. Advantages of the method include low computational time and not being dependent on a massive amount of data for acceptable results. It is also one of the most established models used for time series forecasting. However, they need some preprocessing of the data and selection of hyperparameters to achieve optimal results, which require statistical expertise from the practitioner.

Machine learning methods work by learning the relationship between inputs and outputs of the model by non-statistical approaches. The models that have received the most attention among machine learning methods are the artificial neural networks, often referred to as 'black box' models. Typically an artificial neural network (ANN) consists of an input layer, one or more hidden layers, and an output layer. The input data is fed to the model, and the ANN learns the relationship between the variables by adjusting the weights of interconnected processing units until the best result is achieved. The ANN is a complex model, and it will be further explained in chapter 5.5. The accuracy of an ANN depends on many factors, such as the data preprocessing, the network structure, the learning method, and the chosen hyperparameters. Other machine learning methods applied to short-term prediction of wind power include SVR, decision trees, k-nearest neighbors, and others. These methods usually have fewer hyperparameters that need to be tuned than the ANN models, and have different learning approaches for understanding the relationships between variables. Common for artificial neural networks and other machine learning methods is that they often require a relatively large amount of data for the training process. Consequently, the computational capacity required for training of a machine learning model is often larger for machine learning methods than it is for statistical methods.

2.1.4 Hybrid Methods

A hybrid model combines different forecasting models intending to benefit from each model's advantages and obtain an overall better performance. The combinations can be models from different physical, statistical, or machine learning approaches or models aiming to predict different time horizons. It is not always beneficial to combine different forecasting methods. Since it is not the focus of this thesis, it is only briefly included in the literature study.

2.2 Literature Review

In conjunction with the increasing penetration of wind power in the electrical power production, wind power forecasting emerged in the late 1980s. Since then, the number of research papers published in the field has exploded. In this section, some of the most recent and relevant literature for this thesis will be reviewed.

Until the 2000's the literature on short-term forecasting was greatly dominated by statistical models such as the autoregressive (AR) and the ARMA. In (M. Milligan, Schwartz, & Wan, 2003) standard statistical time series models are used to predict wind power output up to 6 hours ahead using an ARMA model. Their work aims to investigate the feasibility of relatively inexpensive statistical forecasting methods that do not require any data beyond historical wind power generation data. The idea is not to be competitive with commercial forecasting methods, but rather to develop statistical models at a lower cost which may be desirable for small wind farms. (Torres, Garcia, De Blas, & De Francisco, 2005) use the statistical ARMA model and the persistence model to predict the hourly average wind speed up to 10 hours in advance. Their study expands to five locations with different topographic characteristics. They found that after a suitable amount of pre-processing of the data, the ARMA model behaves better than the persistence model, especially in longer-term forecasts. (Duran, Cros, & Riquelme, 2007) develops an AR model with exogenous variables using wind speed and historical wind power data from the previous 12 months as input to their model. Comparing the results to the persistence model and a traditional autoregressive model, the AR model with exogenous variables showed significant improvements.

In more recent years, machine learning and hybrid methods have received much attention. (Rohrig & Lange, 2006) utilize an ANN to predict day-ahead wind power in Germany. For training of the ANN, historical predicted meteorological parameters and contemporaneously measured power data are used to learn the physical coherence of wind speed and wind power output. (Heinermann &

Kramer, 2016) first analyze homogeneous ensemble regressors that make use of a single base algorithm and compare decision trees to k-nearest neighbors and support vector regression models. Heterogeneous ensembles that use multiple base algorithms which benefit from diversity among the weak predictors are then created. They show that a combination of decision trees and support vector regression outperforms the state-of-the-art predictors and homogeneous models, and requires a shorter run time. (Xiaoyun et al., 2016) proposes a deep LSTM network using the principal components of NWP data including air density, pressure, temperature, wind speed, and wind direction as input data to the network. This model is referred to as a PCA-LSTM model. The number of hidden layers in their network was three, with 300, 500, and 200 neurons in each layer, respectively. Comparing the PCA-LSTM to an LSTM using raw data, the authors show that the PCA-LSTM achieves higher accuracy than the LSTM using raw data. The PCA-LSTM can also reduce the complexity of the network and enhance the generalization ability of the model. (Lahouar & Slama, 2017) proposes a RF model for an hour ahead prediction of wind power output. Their work focus on choosing the appropriate weather factors by correlation and importance measures. The spatial average of wind speed, wind direction, and historical wind power data are used as input to the model. Their study's main contribution is to demonstrate the random forest's ability to benefit from exogenous input that may contain relevant information. Lastly, they argue that the random forest may be developed without an optimization process, which is beneficial compared to the extensive optimization required by neural networks. (Bilal et al., 2018) develop a multilayer perceptron network to predict wind turbine power output using wind speed as input data to the model. The network is trained using data collected at different sites along the coast of Senegal. They found in their study that the model's performance is different for all sites and that the difference was related to site characteristics and turbine operation.

(Hong & Rioflorido, 2019) presents a hybrid deep neural network for 24 hours ahead wind power generation forecasting. Their method is based on a convolutional neural network (CNN) that is cascaded with a Radial Basis Function Neural Network (RBFNN) with a double Gaussian function as its activation function. The CNN aims to extract wind power characteristics, and the RBFNN deals with uncertain characteristics caused by intermittent wind characteristics and data spikes due to feature extractions. Their results showed that their proposed methods perform better than comparative models. (J. Zhang, Yan, Infield, Liu, & Lien, 2019) uses a deep neural network to forecast the wind turbine power based on an LSTM algorithm and uses a Gaussian mixture model (GMM) to analyze the error distribution characteristics of short-term wind turbine power forecasting. Using NWP data and historical wind power data, the LSTM model forecasts the power and uncertainties of three wind turbines in the wind farm. The study compare the results of the LSTM, RBFNN,

wavelet network, deep belief network (DBN), backpropagation neural networks (BPNN), and Elman neural network (ELMAN). The results show that the LSTM model can significantly improve the forecasting accuracy.

As the development of wind power technology repeatedly presents larger, more effective turbines, the expectations for short-term forecasting are high. An accurate model for short-term prediction of wind power is recognized as means to allow even greater amounts of wind power to compete on equal footing with conventional energy sources in a competitive electricity market. Hence more advanced and cost-effective forecasting methods need to be developed to better forecast generated power from large-scale wind farms. Wind power prediction, however, is not so straightforward. The accuracy of the developed model is highly site-dependent, and to achieve the best results, a considerable amount of effort should be made in tuning a model based on the characteristics of the local wind profile (Costa et al., 2008; Giebel & Kariniotakis, 2017).

Part II

Theoretical Background

3

Wind Energy and the Power Market

Wind is the movement of air through the atmosphere. As solar radiation is heating the Earth's surface, the surface will heat the air above. When the air is heated, it will expand, become less dense, and rise in the atmosphere. As a consequence, colder air is pushed down, and these motions are the cause of convective air motions in the atmosphere, which is what we know as wind. The sun's heating effect is most substantial near the equator, and from here, looped convection currents moving from the equator and towards the poles are generated. This process results in large-scale movement of air masses in the atmosphere that, combined with the dynamic effects from the Earth's movements, generates prevailing wind patterns all around the globe. The velocity of the wind determines its strength which is directly connected to the amount of kinetic energy that exists in the moving air masses. Therefore, wind energy is, like hydropower, an indirect form of solar energy (Twidell & Weir, 2015). Humans have been harnessing the power in the wind since the first sailing boats were used for navigating rivers and lakes as early as 4000 BC (Letcher, 2017). Today, the kinetic energy contained in the wind can be harnessed by modern wind turbines for the generation of electricity. To better understand the relation between the weather and the power output from a wind turbine, the workings of a wind turbine will briefly be introduced in section 3.1 in this chapter. In section 3.2 a consideration of the availability and potential of wind power is provided, and in section 3.3 the dynamics of the

power market and the necessity of predicting wind power output at different time horizons is explained.

3.1 Wind Turbines

Wind energy is, by definition, the energy content of airflow due to its motion. This type of energy is what is known as kinetic energy, which is a function of the fluid's mass m and velocity U given by

$$E = \frac{1}{2}mU^2 \,. \tag{3.1}$$

The power in the wind is the rate of kinetic energy flow. Considering a wind turbine intercepting a cross-section A of wind speed U and density ρ, the maximum wind power produced by the turbine is

$$P = \frac{1}{2}\rho A U^3 C_p \,. \tag{3.2}$$

The efficiency of the wind power extraction by the turbine is quantified by the power coefficient C_p. The power coefficient represents the ratio of power extracted by the turbine to the total power of the wind resource $C_p = P_T/P_{wind}$ (Letcher, 2017). Hence, the turbine power capture can never be larger than the power in the wind, P_{wind}. The exact working's of how a turbine extracts energy from the wind is a complex field within meteorology and fluid dynamics, which is beyond the scope of this thesis and will not be explained in further detail.

All five wind farms considered in this thesis utilize horizontal axis wind turbines for the production of wind power. Horizontal axis wind turbines, also referred to as HAWTs, have three visible components: the blades, the nacelle, and the tower. Placed inside the nacelle are the drive train and the control system, along with the gearbox and generator. The majority of HAWTs use a gearbox to speed up the rotor speed inside the generator, but some turbines have no gearbox, and the rotor directly drives the generator. On top of the nacelle, one or more anemometers measure the wind speed, and a vane assesses the wind direction. The wind speed and direction of the wind are considered the most important parameters for describing the characteristics of the wind. Yaw control responds to changes in the wind direction by rotating the nacelle to where the wind comes from, and pitch control responds to changes in the wind speed by turning the blades in the direction of the incoming airflow. Yaw and pitch control helps the turbine's ability to absorb the kinetic energy that can be harvested from the wind (Ding, 2019). In figure 3.2 the wind turbine and its components are shown. A typical horizontal axis wind turbine tower ranges up to 80m above the ground.

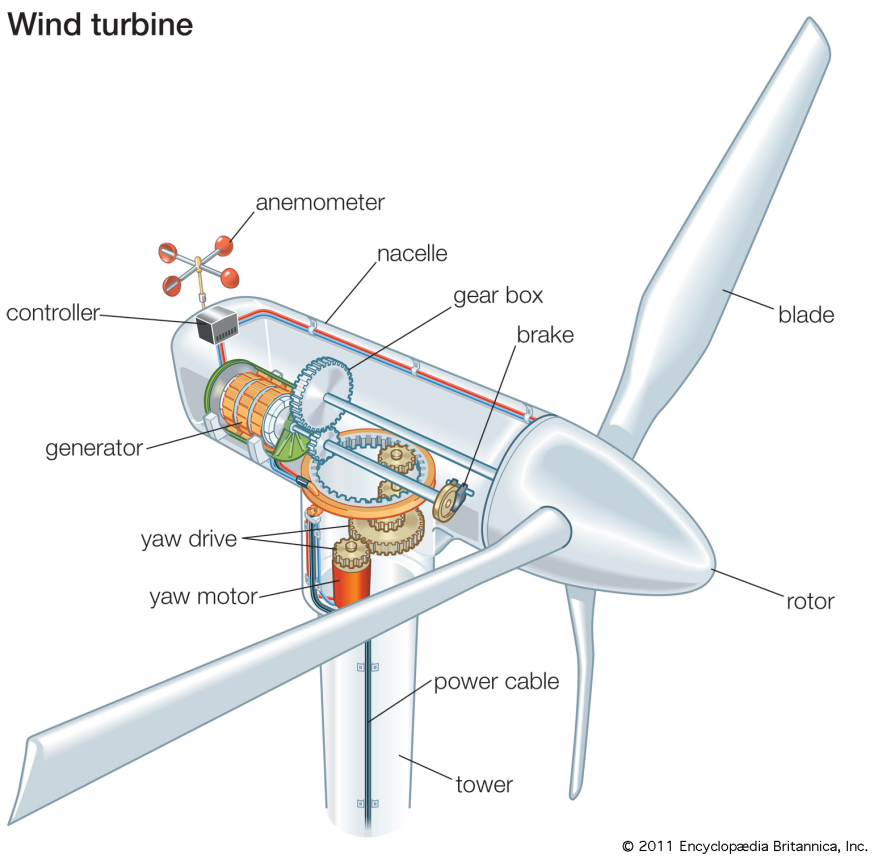


Figure 3.1: The components of a wind turbine. Image retrieved with premission from (Encyclopædia Britannica, 2011)

The relation between the wind speed and the power production from the wind turbine can be studied from a wind turbine's power curve. In the power curve, the minimum speed required for the rotor to start spinning and generate power can be found. This wind speed is referred to as the cut-in speed of the wind turbine. The power curve also gives information about at what point the wind speed is too high for the turbine to produce any energy, and this point is referred to as the cut-out speed. At cut-out speed, there is a risk of damage to the wind turbine, and therefore the breaks will slow the rotor to a standstill at this wind speed. Wind speed data is therefore expected to have significant importance in wind power forecasting with exogenous variables.

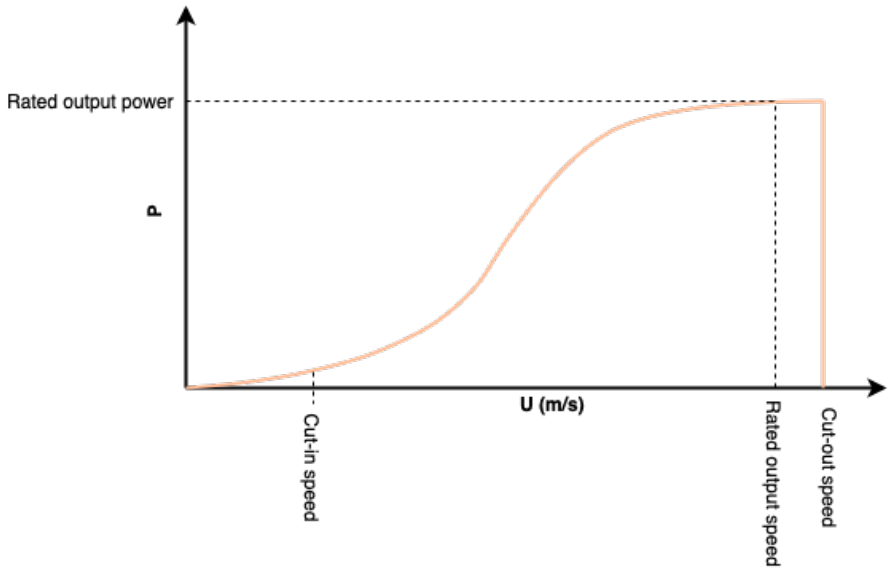


Figure 3.2: The power curve of a wind turbine. The cut-in speed is the minimum wind speed required for the wind turbine to start production of electricity. The cut-out speed is where the wind speed is too strong for the turbine to produce electricity. At the rated output speed the wind turbine is producing its maximum amount of power, which is referred to as the rated output power.

3.2 Potential in the Wind

The global annual source of energy contained in the wind on the Earth's surface is estimated to be approximately 4.04×10^{22} J (Letcher, 2017). This is about 70 times the global energy consumption, as recorded in 2019 (Ritchie & Roser, 2020; BP p.l.c., 2020). A massive amount of resources and efficient wind turbines make wind power production one of the cleanest energy production methods commercially available today. Several studies have concluded that wind energy is the energy source with the lowest life cycle pollution (amount of CO_2eq[1] emissions during the lifetime of a power plant) (Asdrubali, Baldinelli, D'Alessandro, & Scrucca, 2015; Guezuraga, Zauner, & Pölz, 2012). Additionally, wind energy has the second lowest energy payback time of any available energy source after hydropower (Guezuraga et al., 2012). In Norway, the wind power potential is substantial, as some of the highest average wind speeds in Europe are found here. Along the coast, the average wind speed is estimated to lie around 7-9m/s, and inland the average wind speeds are about 3-5m/s

1. Measure of the total amount of greenhouse gas emission presented as the equivalent of CO_2 with the same global warming potential.

(Byrkjedal & Åkervik, 2009). Evidently, Norway has an enormous potential in harvesting clean energy from the wind. However, the complex terrain of hills and mountains that are characteristic of Norway's nature makes installing and assessing new wind farm sites a complicated task. The increased difficulty of predicting the future wind power output in complex terrain might also complicate the wind power integration to the power grid.

3.3 The Power Market

The power market is a dynamic marketplace where power can be bought or sold across areas or countries. In the early 1990s, the Nordic countries, including Norway, deregulated their power markets, introducing free competition while aiming to create a more efficient market and allowing the exchange of power between countries to increase the security of supply. Today the power market is called Nord Pool and covers large parts of Europe, and the power supplied to the grid comes from many different power sources, such as hydro, thermal, nuclear, wind, and solar energy. The varied amount of energy sources connected to the power grid ensures a more liquid power market where large volumes of energy are traded daily, and prices are determined by the balance between supply and demand and the working capacity of power plants across countries. The trading of power happens at two different timescales: the day-ahead market and the intra-day market. The day-ahead market has the purpose of balancing supply and demand on a 24 hour timescale. Customers can sell or buy energy across countries for the next 24 hours in a closed auction. For a country or power grid that relies heavily on wind energy, it can be challenging to know how much energy will be produced 24 hours ahead of time. For this purpose, the intra-day market complements the day-ahead market by favoring trading around the clock for fine adjustments according to unexpected demand variations or power production. Being able to trade power at this timescale is beneficial because transmission system operators reduce the need for operating power reserves and associated costs (Nord Pool, 2021). In Norway, Statnett is responsible for maintaining a balanced and reliable power grid (Statnett, 2018). Suppose the prediction of electricity demand or power production is unreliable and unexpected events lead to a sudden drop or rise in demand/production. In that case, Statnett restores the balance in the grid by operating with power reserves (Statnett, 2016). The power reserves are easily regulated and can provide power on short notice. These reserves are often dispatchable power sources that use fossil fuels and are expensive to use. Therefore an accurate and reliable forecast of the demand curve and the expected power output from power plants are essential tools for balancing the power grid. Following the timescale of power trading and load following dispatching decisions, this thesis will aim to predict the power output from wind farms at different timescales ranging from 1 hour

ahead to 24 hours ahead. Forecasting future demand curves and power output is done by time series forecasting that will be introduced in chapter 4.

4

Time Series Forecasting

In time series forecasting, the objective is to predict the future based on historical and present information. The historical information is obtained from a time series which refers to a sequence of observations recorded at specific time intervals. Statistical analysis of time series is done by studying the internal data structures so that a hypothetical probability model can be set up to represent the data. This model then provides an understanding of the dynamic processes of the time series and can further be used for the prediction of some variable at a specified future time.

A general approach to time series forecasting can therefore be summarized in three steps: Firstly, the statistical properties of the data are identified; After that, a suitable model to describe the data generating process should be properly defined; Lastly, forecasting is done by estimating the future values of the time series based on the model (Chatfield, 2000). To evaluate the forecast, the time series used for prediction is split into a training set and a test set. When setting up the model, the training set is then used to fit the model, and the test set is used to check the accuracy of the forecasts made by the model. An error function is chosen for this purpose; usually, the mean squared error (Brockwell & Davis, 2016), but different error functions for a variety of purposes can be utilized. Another step frequently used in machine learning strategies is to split the training data set into a training data set and a validation data set. Building the model is then changed to fitting a model to the training data and evaluating the model by using the validation data set. If necessary, adjustments are made to the model until the validation of the model gives the desired result. When

the model is perfected, forecasts are made, and the model's performance is evaluated using the test data set.

Forecasting can be done for both univariate and multivariate time series. In the univariate case, the time series depends only on a single observation that changes over time. In contrast, the more complicated multivariate methods are used when the forecast depends on values of one or more additional timeseries variables, called predictor variables or exogenous variables. The exogenous variables may help understand the dynamics of the dependent variable and may improve the accuracy of prediction(Tsay, 2014). This master thesis will focus on the methods that can be used for forecasting multivariate time series.

Traditionally, time series forecasting techniques have been dominated by statistical methods utilizing linear processes for minimization of the error function. In later years these methods have been challenged by machine learning strategies that utilize non-linear processes and offer good generalization properties (Brownlee, 2019). Classical methods may work well when they are presented with stationary data, that is, when the mean and variance of the time series are constant in time, and there is no correlation between aperiodic cycles (Chatfield, 2000). Generally, this is not the case for real-world problems, and choosing a more sophisticated forecasting model may sometimes be desirable. However, preprocessing of the data that involves the elimination of trends and seasonal components resulting in a stationary time series may well lead to better forecasting accuracy both in statistical and machine learning methods (Makridakis, Spiliotis, & Assimakopoulos, 2018).

4.1 Machine Learning for Time Series Forecasting

In this thesis, a variety of machine learning methods for time series forecasting will be implemented and explored for wind power prediction. Machine learning is a field in computer science that uses methods from statistics to give computer algorithms the ability to learn from data and use the knowledge to perform, i.e., prediction tasks. A machine learning problem can be defined as the problem of improving the performance measure of a task through a training procedure (Jordan & Mitchell, 2015). The use of a machine learning algorithm often provides a solution when a problem becomes too complex for simpler statistical algorithms.

4.1.1 Supervised Learning

Supervised learning is an approach to machine learning where a set of measured or preset variables denoted as inputs have some influence on one or more outputs. In a supervised learning approach, the goal is to predict the value of the outputs using the input. The learning task can loosely be stated as follows: given the value of an input vector (X), make a good prediction $(\hat{y})$ compared to the output (y). This task is then repeated several times while adjusting its parameters according to the difference between the predicted output and the labels (Friedman, Hastie, Tibshirani, et al., 2001). In machine learning terminology, the input values X are often referred to as the features, and the output y is called the labels. A general framework for a supervised learning process is illustrated in figure 4.1. To begin with, the data has to be sampled and prepared for the forecasting task. The prepared data is then split into a training, validation, and test dataset. A machine learning model can, after that, be trained using only the training data set. The model is validated on the validation data set according to the performance results of the validation data; the model's hyperparameters can be tuned to achieve a better result. When a satisfying result is reached, the model is used for making predictions using only the test data, and the final performance of the model is then recorded.

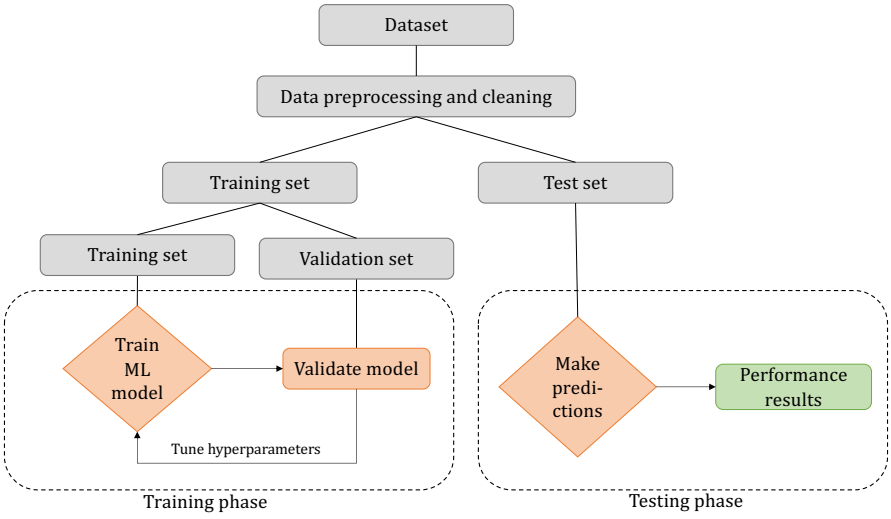


Figure 4.1: The process of a supervised learning method. The dataset is split into a training and a test dataset. The training data is used for validation and training of a machine learning model. When satisfying results are achieved by the model, the model is tested on the test data and the performance is recorded.

4.1.2 Multistep forecasting

In this thesis, the forecasting horizon will be set to 1-24 hours ahead of time. With hourly resolution data, this will require the usage of methods for multistep forecasting. A multistep forecasting task involves prediction of the next H values $[y_{N+1}, \ldots, y_{N+H}]$ of a time series $[y_1, \ldots, y_N]$ with N observations, where $H > 1$ is the forecasting horizon. Performing multistep forecasting is challenging because of the accumulation of errors, reduced accuracy, and increased uncertainty of the prediction. Several strategies can be used for this purpose: recursive, direct, DirRec, and multiple output prediction (Bontempi, Taieb, & Le Borgne, 2012).

The **recursive** strategy trains a one-step model and then uses it recursively for returning a multistep prediction. The strategy builds on producing a one-step forecast and then using this forecast to predict the second forecast and continue using the forecasted values recursively until the desired forecast horizon is achieved. The process can be described as in equation (4.1).

$$x_{t+1} = f_h(x_t, ..., x_{t-n+1}) + w_{t+h} \tag{4.1}$$

A drawback of this strategy is that it is sensitive to estimation errors, since the estimated values instead of actual values are used more and more when predicting further into the future (Bontempi et al., 2012).

The **direct** strategy produces separate forecasting models for each forecasting horizon independently. In this case, the first model is responsible for the first forecast, the second model for the second step forecast, and this continues until there are as many models as forecasting horizons. The process is in (4.2).

$$x_{t+h} = f_h(x_t, ..., x_{t-n+1}) + w_{t+h} \tag{4.2}$$

Since this strategy does not utilize the forecast values to make another prediction, it is not as sensitive to the accumulation of errors as the recursive strategy. However, since the model for each forecast is learned independently, it assumes that there is no statistical dependence between the predictions, which is not usually the case in real-world problems. This method also requires more computation time than the recursive strategy, since it relies on learning as many models as forecasting horizons.

A combination of the two strategies mentioned above is the **DirRec** forecasting strategy. This strategy relies on computing the forecasts using different models for every forecasting horizon. However, each model may use the forecasts produced by the model from the previous time step as input values. By using this method, the weaknesses from each of the independent strategies may be avoided (Bontempi et al., 2012). This process is presented as equation

(4.3).

$$x_{t+h} = f_h(x_{t+h-1}, ..., x_{t-n+1}) + w_{t+h} \tag{4.3}$$

Lastly, the **multiple output strategy** involves using historical data to learn a single multiple output model F (Bontempi et al., 2012). In this case, the estimation is not a scalar, but a time series as long as the number of elements in the forecasting horizon. Using this strategy, one can learn the stochastic dependencies between future values, which may help improve the prediction accuracy. However, this method constrains all the horizons to be predicted with the same model structure and using the same learning procedure. This constraint greatly reduces the flexibility, and the variability of the approach and risks returning a biased model (Taieb, Sorjamaa, & Bontempi, 2010). The single multiple output model F is described by

$$[x_{t+H}, ...x_{t+y}] = F(x_t, ...x_{t-n+1}) + \mathbf{w} \tag{4.4}$$

where **w** is a vector noise term. The estimation of the next H values are then given by

$$[\hat{x}_{t+H}, ..., \hat{x}_{t+1}] = \hat{F}(x_N, ...x_{N-n+1}) \tag{4.5}$$

This strategy is referred to as the Multiple Input Multiple Output (MIMO) strategy and is only a variant of the multiple output strategies.

4.2 Preprocessing methods

A time series is said to be stationary if its statistical properties, such as the mean and variance, are constant in time. Most statistical methods used for time series forecasting are based on the assumption that the data fed to the model is stationary or can be rendered stationary. In doing so, it can be assumed that the future statistical properties of the data are the same as the current statistical properties, and forecasting the data becomes a much simpler task. In real-world problems, it is not usual for time series to be stationary. As a result, a big part of time series analysis involves identifying the characteristics of the data and finding ways to transform it, so it becomes stationary. Essentially, the data is decomposed into trends, seasonality, and residuals, and the first step in determining if any of these characteristics are present is to prepare a plot of the time series (Brockwell & Davis, 2016; Chatfield, 2000).

The trend and seasonality in a time series can be identified if the mean and the variance of the series is a function of time, that is, if the series can be described

as $x_t = m_t + s_t + y_t$, where m_t is a slowly changing function recognized as the trend component, s_t is a function with period d, and y_t is a stationary random noise component, hereafter referred to as the residuals. The aim of preprocessing is to remove the components m_t and s_t to achieve a stationary series $\hat{y}_t$ (Brockwell & Davis, 2016).

4.2.1 Elimination of trend and seasonality

There are a few different methods for elimination of trend and seasonality in the data. For elimination of trends in the absence of seasonality, a moving average filter or other methods can be used for estimation of the trend component in the data, and later the trend can be removed from the original time series. For example, the trend can be estimated by utilizing the method of least squares estimation. By doing so, the growth rate of the trend in the form $m_t = a_0 + a_1 t + a_2 t^2$ can be estimated (Brockwell & Davis, 2016), and the detrended series can be described as $\hat{y}_t = x_t - \hat{m}_t$. Another option is to remove the trend by differencing the data. In that way, any polynomial trend of degree k can be reduced to a constant, and the resulting time series x_t will have a constant mean value.

In the case of seasonality in the time series, the seasonal component of the series can either be modeled directly and be subtracted from the data, or a seasonal differencing can be applied to the data. The seasonal difference is the difference between an observation and the previous observation from the same season $\hat{y}_t = x_t - x_{t-m}$, where m is the number of seasons (Hyndman & Athanasopoulos, 2018). To avoid confusion the differencing applied to remove the trend component of the series is often referred to as "first differences" meaning differences at lag 1, and the differencing applied to remove the seasonal component of the series is called seasonal differencing. A combination of the methods for detrending and deseasonalization can be used to obtain stationary data in the presence of both seasonal and trend components (Brockwell & Davis, 2016).

While most statistical methods used for time series forecasting require stationary data for adequate prediction accuracy, this is not always the case for more complex machine learning methods. In the literature, there are mixed opinions on the relevance of data preprocessing when using machine learning. Some studies state that machine learning methods can effectively model any type of data pattern and can therefore be applied to the original data. One of the main arguments for this claim is that the machine learning methods can learn the underlying data structures and account for trends and seasonalities in the computations (Sharda & Patil, 1992). In contrast, other studies conclude that without preprocessing of data, machine learning methods may

yield suboptimal results (G. P. Zhang & Qi, 2005). In (Makridakis et al., 2018) the forecasting accuracies obtained from a multilayer perceptron model (MLP) applied to a number of 10 different preprocessing methods were compared using the symmetric mean absolute percentage error (sMAPE) and the mean absolute scaled error (MASE). It was observed that seasonal adjustments significantly improve the accuracy of one step ahead forecasts, and a combination of deseasonalization and detrending provides the overall better accuracy using the MLP for forecasting purposes.

4.2.2 Augmented Dickey Fuller and KPSS test

Statistical tests such as the Augmented Dickey Fuller (ADF) test and the Kwiatkowski, Phillips, Schmidt and Shin (KPSS) test can be used as a systematic approach to determine whether a time series is stationary or non-stationary. Both of the tests can be used to inform to some degree whether a null hypothesis can be rejected or fail to be rejected. The ADF test is a unit root test that determines how strongly a trend defines a time series. The null hypothesis of an ADF test is that the time series can be represented by a unit root, that is, it is not stationary. The alternate hypothesis is that the time series is stationary. The p-value of the test interprets the results. A p-value below the threshold suggests that the null hypothesis is rejected, while a p-value above the threshold fails to reject the null hypothesis (Brockwell & Davis, 2016). The null hypothesis of the KPSS test is that the time series is stationary, and the alternate hypothesis states the opposite. The KPSS test is further described in (Kwiatkowski, Phillips, Schmidt, & Shin, 1992). A major setback for the KPSS test is that it rejects the null hypothesis too often (Hobijn, Franses, & Ooms, 2004). Both the ADF and the KPSS test are therefore often used in conjunction with each other, and if both tests indicate stationary time series, the time series is most likely stationary.

4.3 Forecast Evaluation

The perfect predictive model will achieve zero error, which is the best performance. However, because of the volatile nature of wind this is not possible for short-term wind power forecasting. All forecasting models aiming to forecast the wind power of a wind turbine or wind farm will have some error. Many factors can affect the error measure of a wind power prediction model, such as the size of the turbines and the wind farm, the sampling rate of the data, the forecast horizon, the chosen algorithm, the hyperparameters of a model, as well as the wind farm site and local topography characteristics. It can therefore be difficult to compare the performance of forecasting models from different

works, and the accuracy of a proposed model should be compared to one or more robust baseline models instead (Hanifi et al., 2020).

A baseline model in a time series forecasting problem is essential, as it provides a point of reference for the comparison of different models. If a model achieves worse results than the baseline model, the baseline model makes it evident that the prediction model should be improved or abandoned. According to (Brownlee, 2019), a baseline model should be fast and straightforward to implement and be able to reproduce the output given the same input. In this thesis, the persistence model, as described in chapter 2, will be used as a baseline model.

4.3.1 Performance Metrics

In order to assess the forecasting quality, several performance metrics can be used. A typical loss function, which will be further explained in chapter 5, that is used by neural networks for regression problems is the mean squared error (MSE). The MSE is the average of the squared differences between the actual and the predicted values. Because of the squaring, the larger errors will be penalized more than minor errors, so that the model is punished more for making bigger mistakes. The MSE is defined as

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{4.6}$$

where N represents the number of samples, y_i are the observed values from the test dataset, and $\hat{y}_i$ are the forecasted values. The most popular two metrics used for measuring the performance of wind power forecasting models are the RMSE and the MAE, defined respectively as

$$RMSE = \sqrt{\frac{1}{n_{test}}\sum_{i=1}^{n_{test}}(\hat{y}_i - y_i)^2} \tag{4.7}$$

$$MAE = \frac{1}{n_{test}}\sum_{i=1}^{n_{test}}|\hat{y}_i - y_i| \tag{4.8}$$

Both the MSE, the RMSE and the MAE are scale-dependent measures that are useful when comparing the performance of models used on the same set of data (Hyndman & Koehler, 2006). When evaluating models across different datasets, this might be an issue because the base value that is to be predicted may have different impacts on the performance across different datasets, while the relative error across datasets may actually be the same. Additionally, the MSE

and the RMSE are based on the squared error loss function and thus sensitive to the existence of outliers. In contrast, the MAE is based on the absolute error loss and is less sensitive to outliers. A better measure for comparing wind power predictions across different locations may be the normalized root mean squared error (NRMSE). While the RMSE is helpful for the comparison of different models for the same dataset, the NRMSE is not scale-dependent and therefore preferred for comparing model results across different datasets. The normalized root mean squared error is defined as

$$NRMSE = \frac{RMSE}{y_{max} - y_{min}} \tag{4.9}$$

Here $y_{max} - y_{min}$ represent the range of the observed data.

5 Forecasting Models

In this thesis, four different forecasting algorithms will be implemented for the purpose of wind power forecasting. In this chapter, the theoretical background of the models will be explained. The methods range from traditional statistical methods to the complicated ANN models. More specifically, the ARIMAX model will represent the traditional statistical method. The RF and the SVR models will represent the shallow machine learning models, and the LSTM model will contribute from ANN methods.

5.1 ARIMAX

One of the most commonly used methods for time series forecasting is the autoregressive integrated moving average (ARIMA) model (Hyndman & Athanasopoulos, 2018; Brockwell & Davis, 2016). The ARIMA model builds upon three key aspects; autoregression, integration, and moving average, where integration refers to the reverse of differencing (Hyndman & Athanasopoulos, 2018). In an autoregression model, the variable of interest is forecast using a linear combination of past values of the variable itself. An autoregressive model of order p can be written as

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdot + \phi_p y_{t-p} + \epsilon_t \tag{5.1}$$

where ϵ_t is white noise. This process is alone referred to as an **AR(p)** model, where p refers to the number of lags included in the model.

The moving average part uses the dependency between an observation and a residual error from a moving average model on lagged observations. In a moving average model, forecasting is done using past forecast errors in a regression-like model that can be written as

$$y_t = c + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdot + \theta_q \epsilon_{t-q} \tag{5.2}$$

where ϵ_t is white noise. This process is referred to as an **MA(q)** model of order q.

In order to make the time series stationary, the degree of first differencing that is involved in the model is determined by the d parameter. By combining differencing of the time series data, the autoregressive model and the moving average model, an **ARIMA(p, d, q)** model is obtained, where p refers to the order of the autoregressive part, d is the degree of differencing involved, and q is the order of the moving average part (Hyndman & Athanasopoulos, 2018). The ARIMA model does not support time series with a seasonal component, and the seasonal component must be removed by methods as described in 4.2. For time series that contain seasonal components there exist ARIMA models that can account for seasonality. The non-seasonal ARIMA model can be written as

$$y'_t = c + \phi_1 y'_{t-1} + \ldots + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + \ldots + \theta_q \epsilon_{t-q} + \epsilon_t \tag{5.3}$$

where y'_t is the differenced series. Values for p and q are determined by looking at the number of significant lags in the autocorrelation function (ACF) and partial autocorrelation function (PACF) plot of the time series. That is, the p parameter is equal to the number of significant lags as shown in the ACF plot of the series, and th q parameter is equal to the number of significant lags in the PACF plot of the timeseries. The parameter d is determined as the number of times a series has to be differenced in order to become trend stationary. The ARIMAX model is an extension of the ARIMA model, where the ARIMAX model includes exogenous variables in the model.

5.2 Decision Trees

A decision tree is a simple yet effective machine learning algorithm used for both regression and classification problems. As illustrated in figure 5.1 a decision tree can be understood as a flow chart structure where each node in a tree denotes a test on an attribute, each branch in the tree represents an outcome of a test, and each leaf or terminal node holds a class label or prediction ($\hat{y}$).

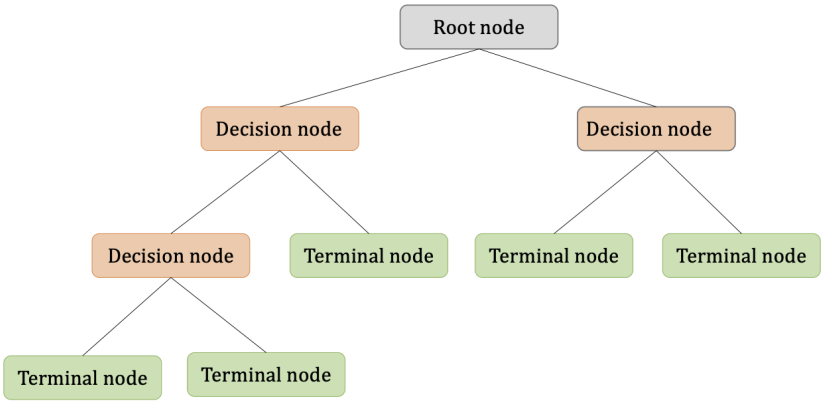


Figure 5.1: A simplified illustration of a decision tree. In the root node, colored gray, the first decision of how to split the training data is made. In the decision nodes, or internal nodes, colored orange, different conditions are decided, and the outcome of the internal nodes are illustrated as connections or branches between the node. The terminal nodes, or the leaves, colored green, represents the final decision of the tree. That is the final regression value.

Since this thesis will handle a regression problem, decision trees for regression problems will be described in this section. A regression tree is built by splitting the training set into unique subsets based on a split point automatically determined by the algorithm. The input data to the algorithm consist of p inputs and responses for N observations. That is (x_i, y_i) for $i = 1, 2, \cdot, N$ with $x_i = (x_{i1}, x_{i2}, \cdot, x_{ip})$. If the training data is split into M regions $R_1, R_2, \cdot, R_M$, the response can be modeled as a constant c_m in each region (Friedman et al., 2001):

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \tag{5.4}$$

Using minimization of the sum of squares as a criteria it can be seen that the best $\hat{c_m}$ is the average of y_i in region R_m

$$\hat{c}_m = ave(y_i | x_i \in R_m)$$

Starting with all the data, a splitting variable j, and a split point s, a pair of half-planes can be defines as

$$R_1(j, s) = \{X | X_j \leq s\} \text{ and } R_2(j, s) = \{X | X_j > s\}$$

Then determining the splitting variable j and the split point s is done by

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right] \tag{5.5}$$

When the best split is determined, the data is partitioned into the two resulting regions, and the splitting process is then repeated for each of the two regions. This process is continued until there is only one node left in the tree, or until a pre-determined number of terminal nodes are reached (Friedman et al., 2001). A couple of disadvantages concerning decision trees is that they are prone to overfitting to the data and small variations in the dataset may result in completely different trees being generated. A solution to these problems are proposed by the Random Forest algorithm that will be presented in the next section.

5.3 Random Forest

The RF algorithm is an ensemble learning technique that was first introduced by Leo Breiman in 2001 (Breiman, 2001) as a modification of bootstrap aggregation, also referred to as bagging. When using bagging for regression problems, the same regression model is fit to several bootstrapped samples of the training data before the result is averaged. This way, bagging reduces the variance of an estimated prediction function. The technique is known for working specifically well with decision trees, since trees can capture complex interaction structures in the data. A multitude of regression trees are constructed in a random forest algorithm, and each tree is trained using a bootstrap sample of the training dataset. Additionally, the random forest algorithm involves randomly selecting a subset of input features and thresholds at each split point in the construction of the trees. The random selection of features and thresholds results in less correlated trees that yield resistance to overtraining and often results in better performance than bagged decision trees (Friedman et al., 2001). A simple illustration of a random forest is shown in figure 5.2.

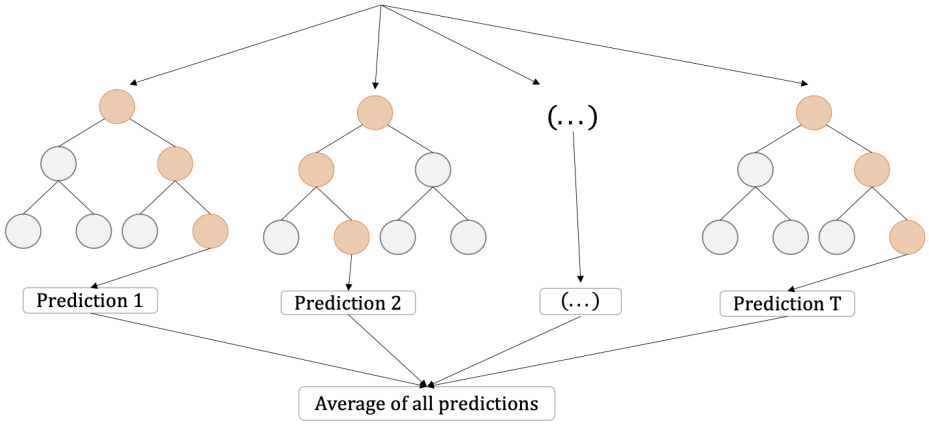


Figure 5.2: The random forest, as illustrated in the figure above, is a collection of several decision trees. All the decision trees calculate an outcome, and the final decision from the random forest model is the average of the leaves from the decision trees.

The algorithm works by extracting a bootstrap sample Z^* of size N from the training data. A random forest tree T_b is then fitted to the bootstrapped data. This is done by selecting m random features from the dataset, a split-point is then determined among the selected features, and the node is split into two daughter nodes where the procedure is repeated. The training is stopped when the terminal nodes has reached the pre-determined minimum number of nodes, n_{min}. The forest of regression trees creates an ensemble of regression values, and the final prediction can be determined by for example averaging over the ensemble (Friedman et al., 2001), or using some other aggregation function.

The performance of regression trees can be controlled by several hyperparameters that needs tuning to achieve the best result. The main parameters are (Babar, Luppino, Boström, & Anfinsen, 2020):

- The number of features considered in each node (m)
- The number of trees in the forecast (T)
- Elements in a node required to perform a split (M_s)
- Elements required to create a node (M_l)
- The maximum depth up to which a tree can grow (L)

(Breiman, 2001) recommends as a default value for m to be set to $p/3$, where

p refers to the number of features in the dataset. However, the parameter is problem dependent, and should be treated as a hyperparameter. The number of trees in the forest is not the most critical hyperparameter, but the more trees the more the computational load increases, and typically an initial increase in accuracy is achieved before a saturation point is reached (Luppino, Bianchi, Moser, & Anfinsen, 2018).

5.4 Support Vector Regression

The use of support vector machines in forecasting problems is often referred to as support vector regression (SVR). The idea behind this method is to map the input data into a higher dimensional feature space by a nonlinear mapping function and perform linear regression for forecasting in this feature space. Therefore, linear regression in high-dimensional feature space corresponds to nonlinear regression in the low-dimensional input space (Müller et al., 1997). The linear function that describes the relationship between x and y in the high-dimensional feature space is described as

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x}) + b \tag{5.6}$$

where $\phi(\boldsymbol{x})$ represent the nonlinear mapping of data $\boldsymbol{x}$ to the high-dimensional feature space from the input space, and b and $\boldsymbol{w}$ are adjustable coefficients. In support vector regression a penalty is introduced for points that lie too far away from the predicted line $f(x)$, but for points that lie within a predefined distance ϵ from the border, there is no penalty (Welling, 2004). In figure 5.3 the epsilon tube is presented, and $f(x) + \epsilon$ and $f(x) - \epsilon$ illustrate the margin. The data points that lie within this margin gives no contribution to the loss, while the two data points laying outside the margin will contribute with ξ or ξ^*.

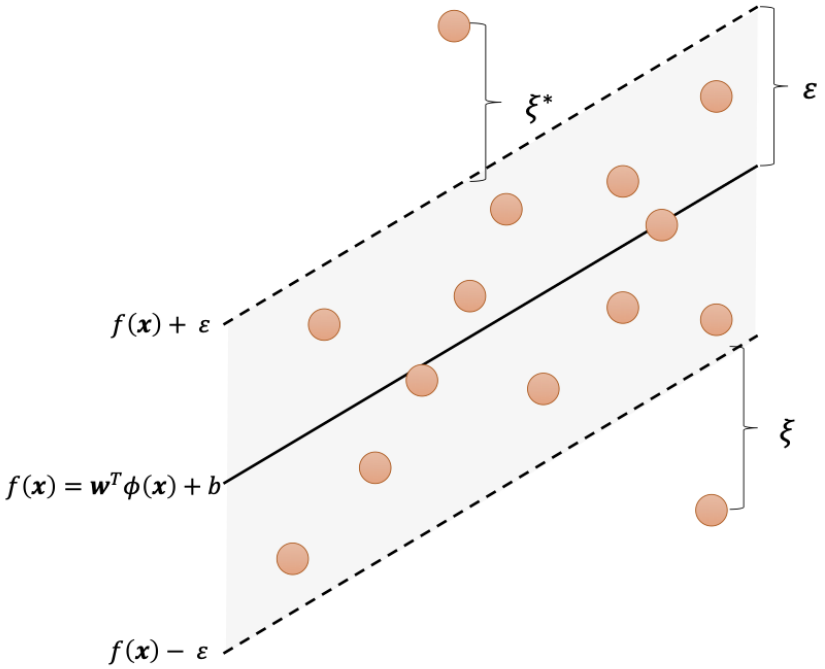


Figure 5.3: The SVR model is essentially linear regression in a high-dimensional feature space. The function $f(\boldsymbol{x})$ describes the relationship between x and y in feature space. The gray area that spans a distance ϵ out from both sides of the function $f(\boldsymbol{x})$ represent the epsilon tube. Any points within this area has no effect on the cost of the model. Points that lie outside the epsilon tube are penalized by some value $\boldsymbol{\xi}$.

The purpose of the SVR method is to maximize the margin while still representing the data well. This is done by minimizing the empirical risk described as (Liu, Chen, & Mori, 2015)

$$\min R(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l} L_\epsilon(y_i, f(x_i)) \tag{5.7}$$

where $\frac{1}{2}\|\boldsymbol{w}\|^2$ is the normalization term related to the size of the margin, C is a punishment parameter, and $L_\epsilon(y_i, f(\boldsymbol{x}_i))$ is the ϵ-insensitive loss function (Liu et al., 2015):

$$L_\epsilon(\boldsymbol{y}_i, f(\boldsymbol{x}_i)) = max(|\boldsymbol{y}_i - f(\boldsymbol{x}_i)| - \epsilon, 0) \tag{5.8}$$

This loss function states that data points $\boldsymbol{y}_i$ that lie outside a distance ϵ from $f(\mathbf{x}_i)$ will contribute to the loss. In order to keep the margin to a reasonable

size while at the same time allowing some outliers in the data, two positive slack variables are introduced, ξ_i and ξ_i^*. The slack will contribute to the error function by penalizing any data point outside the margin and the errors are then minimized by

$$R(\boldsymbol{w}, \xi, \xi^*) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \tag{5.9}$$

$$\text{subject to} \begin{cases} \boldsymbol{y}_i - \boldsymbol{w} \cdot \phi(\boldsymbol{x}_i) - b \leq \epsilon + \xi_i \\ \boldsymbol{w} \cdot \phi(\boldsymbol{x}_i) + b - \boldsymbol{y}_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \tag{5.10}$$

The $\boldsymbol{w}$ parameter (5.11) and the SVR function (5.12) can finally be obtained by solving the optimization problem using the generalized method of Lagrangian multipliers to find a solution that satisfies the Karush-Kuhn-Tucker conditions (Liu et al., 2015). These become:

$$\boldsymbol{w} = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\phi(\boldsymbol{x}_i) \tag{5.11}$$

$$f(\boldsymbol{x}) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)K(\boldsymbol{x}, \boldsymbol{x}_i) + b \tag{5.12}$$

where $K(\boldsymbol{x}, \boldsymbol{x}_i)$ is the kernel function. In using this method the kernel trick explained in (Schölkopf, 2000) is utilized. The model can be trained for the time series data $\boldsymbol{x}_t = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and the input of the model is the number of lags included in the model, which is determined by the user. The ability of SVR and RF to solve nonlinear regression problems makes the models promising for forecasting problems that involves nonlinear and non-stationary time series (Liu et al., 2015). Additionally, both the RF and SVR models are widely used for time series forecasting, and known for being able to achieve good results on different datasets. Collectively, the models cover important aspects of the machine learning domain, where the RF model represent ensemble learning methods, and the SVR model covers kernel learning methods.

5.5 Artificial Neural Networks

Artificial neural networks are models that are designed to simulate the human brain when analyzing and processing information. The idea was first introduced by McCulloch and Pitts in 1943 (McCulloch & Pitts, 1943). The most common

artificial neural network, known as the perceptron or the multilayer perceptron, consist of neurons, usually referred to as nodes, and the connections between them. For each connection there is a weight w_j, and associated with each node there is an activation function $f(\cdot)$. The value v_j of each neuron is calculated by applying the activation function to a weighted sum of the values of its input nodes (Lipton, Berkowitz, & Elkan, 2015). A general representation of a node and its corresponding weights in a perceptron is shown in figure 5.4

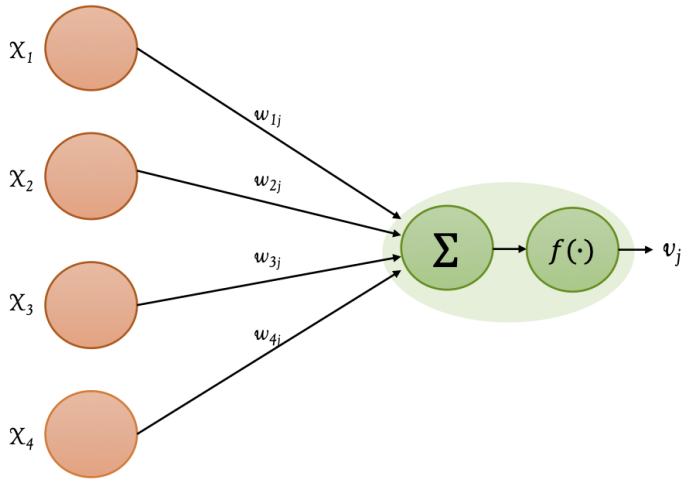


Figure 5.4: Node and corresponding weights in a perceptron model. In the figure $X_1, \ldots, X_4$ represent the input values to a neural network. The weights $\omega_{1j}, \ldots, \omega_{4j}$ are represented as arrows from the input nodes to a neuron/node. The output value of the node v_j is calculated by applying an activation function to a weighted sum of the values of the input nodes.

Every neural network can be classified according to its architecture and the method used for training. The architecture of the network refers to the number of nodes and layers in the network, which activation function that is used, and the neural connections, whereas the training algorithm describes how the network adapts its weights (Weron, 2014). In this section, a feedforward neural network will be introduced to explain the main principles for training and optimizing a neural network, before moving on to concepts of recurrent neural networks that form the basis for the LSTM model.

5.5.1 Multilayer Perceptron

The multilayer perceptron algorithm is, despite the growing number of viable methods, one of the most widely used algorithms within artificial neural

networks (Palit & Popovic, 2006). Multilayer perceptrons are also called feed-forward neural networks because of how information is handled within the network. The goal of such a network is to approximate a function $f^*(\boldsymbol{x})$ that defines a mapping $y = f(x;\theta)$, while learning the parameters θ that yields the best function approximation (Goodfellow, Bengio, & Courville, 2016). The structure of a network consists of an input layer and an output layer interconnected by a chosen number of hidden layers. The hidden layers perform a mapping between the input layer and output layer. In each hidden layer, the input data is processed, and the output of each layer is a function of the input data $f^{(l)}(\boldsymbol{x})$, where l is the number of the hidden layer. The output from the network is the composition of the outputs from the hidden layers in the network $f(\boldsymbol{x}) = f^{(l)}(\ldots f^{(2)}(f^{(1)}(\boldsymbol{x})))$ (Goodfellow et al., 2016). This interconnected chain of approximation functions gives the network the ability to learn characteristic features of the input data and generalize that knowledge, which has proven to be efficient for use in time series forecasting (Palit & Popovic, 2006). Figure 5.5 presents a multilayer perceptron network with four input nodes, three nodes in the first hidden layer, two nodes in the second hidden layer, and one single output node.

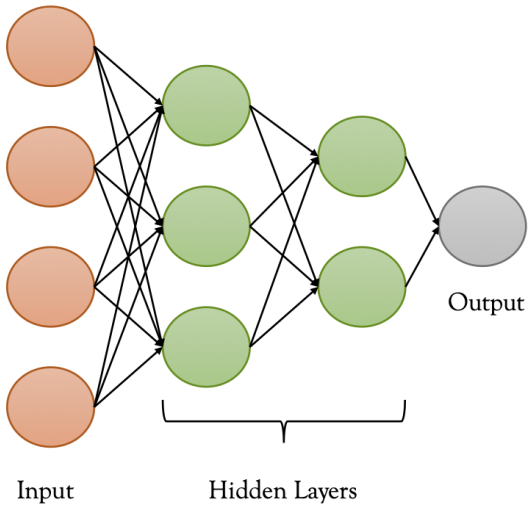


Figure 5.5: Architechture of a multilayer perceptron network. Here the input layer consist of four nodes, two hidden layers with three and two nodes, respectively, form the body of the network, and the output layer is only one node. Every green node in the network represents a neuron, where the output from the node is calculated by applying an activation function to a weighted sum of the values of the input nodes.

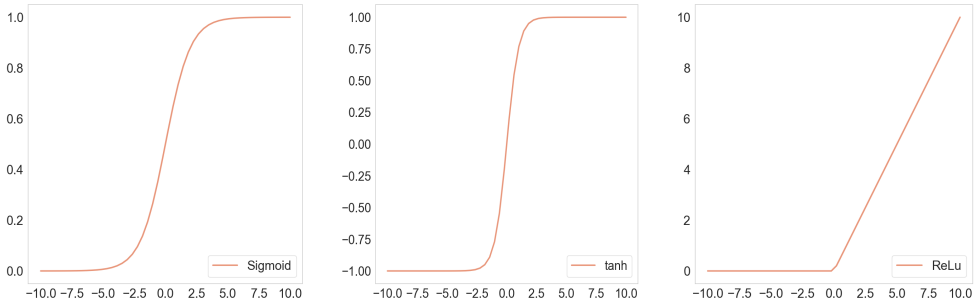
Each node in the hidden layers performs a mapping of the input vector $\boldsymbol{x}$ to a scalar value that will act as input to the next layer. The mapping performed by a neuron i in layer l can be described as (Gonzalez & Woods, 2018)

$$z_i(l) = \sum_{j=1}^{n_{l-1}} w_{ij}(l) v_j(l-1) + b_i(l) \tag{5.13}$$

for $i = 1, 2, \ldots, n_l$ and $l = 2, \ldots, L$, where L is the total number of layers, z_i is the total input to neuron i in layer l, w_{ij} is the weight that connects the output of neuron j to the input of neuron i, $v_j(l-1)$ is the output of neuron j in the previous layer $(l-1)$, and b_i is the bias value associated with the ith neuron. This mapping of the input layer to the output layer is referred to as a forward pass through the network. The output of neuron i in layer l is given by (Gonzalez & Woods, 2018)

$$v_j(l) = h(z_i(l)) \tag{5.14}$$

for $i = 1, 2, \ldots n_{,l}$, where h is an activation function. The output function is an important feature in the network that enables the network to learn non-linear data patterns. Essentially, it converts the output signal of a previous node into information to be passed to the next node. Various activation functions can be utilized in the network and a few of them are shown in figure 5.6.



(a) Sigmiod **(b)** Tahn **(c)** ReLu

Figure 5.6: Three different activation functions. The sigmoid activation function and the hyperbolic tangent activation function has the same shape, but the latter is centered around zero in both dimensions.

During training of the network, the objective is to drive $f(\boldsymbol{x})$ to match $f^*(\boldsymbol{x})$ as closely as possible. Each input sample $\boldsymbol{x}$ is accompanied by a label $y \approx f^*(\boldsymbol{x})$, and the goal of the output layer is to produce a value as close as possible to y_i for every sample in the training data $\boldsymbol{x}$.

Gradient Based Optimization

The training algorithm used in neural networks is almost always built on using gradient-based optimization algorithms that aim to minimize some cost function. These will not be described in detail in this thesis. However, a few examples are the stochastic gradient descent and the Adam optimizer, which is an extension of the stochastic gradient descent. The optimizer used for training the neural networks in this thesis is the Adam optimizer. For further explanation of how the Adam optimizer works, the reader is referred to (Goodfellow et al., 2016).

Cost Function

A common cost function used in training of neural networks for regression tasks is the MSE function, which is defined as

$$J(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{2n} \sum_{m=1}^{n} (y_i(m) - \hat{y}_i(m))^2 \tag{5.15}$$

where $y_i(m)$ is the desired output of the network, $\hat{y}_i(m)$ is the output from the network, n is the number of output neurons, and θ refers to the weights w and biases b of the network (Gonzalez & Woods, 2018).

Regularization

To avoid overfitting of a neural network the cost function is often combined with a regularization term. An overfit network is recognized by a small loss obtained on the training data, and a large loss for the validation data. There are several regularization strategies that can be applied to a neural network model, such as adding restrictions to the parameter values in the network or adding constraints to the parameter values in the loss function. One way to do this is to add a criterion that expresses a preference for the weights to have smaller L^2 norm. Specifically this will be

$$J(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{2n} \sum_{m=1}^{n} (y_i(m) - \hat{y}_i(m))^2 + \lambda w^T w \tag{5.16}$$

where λ is a value chosen ahead of time that controls the strength of preference for smaller weights. When $\lambda = 0$, no preference is imposed, and a larger λ forces the weights to become smaller. Minimizing $J(\boldsymbol{\theta})$ results in a choice of weights that make a trade-off between fitting the training data and being small. Another regularization technique that can be used is the dropout technique.

The dropout technique works by randomly dropping units in the network from the training process in order to avoid units from co-adapting too much, so that overfitting of the network is avoided (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Often regularization is needed for training of deep network as some networks have a tendency to overfit to the training data as the network depth increases (Goodfellow et al., 2016).

Backpropagation

After one forward pass through the network, the total cost of the network is computed. The backpropagation algorithm makes use of the information from the loss function, and feeds the output error obtained in the forward pass back to the network to compute the changes required to update the parameters, that is to compute the gradient. To find a scheme that can adjust all the weights in the network using training patterns, we need to know how the error changes with respect to a change in the net input to any neuron in the network

$$\delta_j(l) = \frac{\partial J}{\partial net_j(l)} \tag{5.17}$$

Using (5.13) and (5.17), the rate of change of J with respect to the networks weights and biases can be found

$$\frac{\partial E}{\partial w_{ij}(l)} = \delta_i(l) v_j(l-1) \tag{5.18}$$

$$\frac{\partial E}{\partial b_i(l)} = \delta_i(l) \tag{5.19}$$

Then the weights can finally be updated using gradient descent or any other optimization algorithm. In the gradient descent algorithm a new weight is proposed by

$$w_{ij}(new) = w_{ij}(old) - \alpha \delta_i(l) v_j(l-1) \tag{5.20}$$

where α denotes the learning rate of the network, that is, the size of the step in the optimization algorithm. A forward pass through the network using the updated weights are then computed again, and this process continues until the loss has reached an acceptable level (Gonzalez & Woods, 2018).

The multilayer perceptron is a heavily parametrized model and the complexity of the model can be controlled in terms of how many hidden layers that are chosen and the number of neurons per hidden layer. The performance of a neural network model heavily depends on the choice of optimizer, loss function, the network architecture and choice of regularization techniques (Goodfellow et al., 2016).

5.5.2 Recurrent Neural Networks

RNNs are different from feed-forward neural networks such as the MLP in the sense that they base their understanding of a subject on previous knowledge from looping inside the network. The loop allows the network to step through sequential data while preserving the state of the nodes in the hidden layers between steps. This way the recurrent neural networks can recognize patterns in sequences of data, as its output at each timestep depends on previous output and past computations. In contrast, the traditional feed forward neural networks assume that all inputs and outputs are independent of each other (Bianchi, Maiorino, Kampffmeyer, Rizzi, & Jenssen, 2017). In figure 5.7 a simple RNN architecture is illustrated. Here X refers to the input layer of the network, h represent the hidden layers and y refers to the output of the network. For each input timestep of the timeseries the recurrent neural network predicts one output, and loops the weights from that calculation back to the network for calculation of the next timestep.

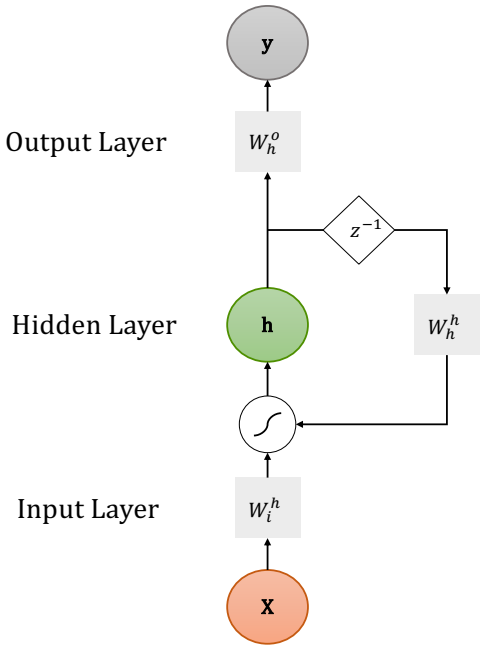


Figure 5.7: Simple illustration of a general RNN. The input layer, hidden layer and the output layer is presented by X, h and y, respectively. W_i^h is the marix representing the weights from the input layer to the hidden layer. W_h^h is the matrix that represent the weights from the hidden layer that is looped inside the network. W_h^o represent the weights from the hidden layer that is passed to the output layer. z^{-1} represent a time shift operator with a time delay of one timestep.

When used for timeseries prediction, a recurrent neural network is trained on the input timeseries X and tries to reproduce the desired temporal output y. As with the multilayer perceptron algorithm, the training procedure is almost always based on gradient techniques and the loss function is often the MSE that was described in the previous section. The difference in training between a feedforward neural network and an RNN lies in the backpropagation algorithm. The RNNs rely on an extension of the backpropagation algorithm called backpropagation through time (BPTT) in order to account for the temporal dependencies in the data. In the recurrent neural network each input of the timeseries has one input timestep, one copy of the network and one output. An illustration of the unrolled RNN is shown in figure 5.8. The backpropagation through time algorithm works by propagating through the whole unrolled net-

work in order to accumulate errors across each of the timesteps. The network is then "rolled up" and the weights are updated according to the total error obtained across every timestep of the input sequence (Brownlee, 2017). When the number of input timesteps is large, the backpropagation through time algorithm is computationally very expensive as the number of derivatives required for a single weight update is the same as the number of input timesteps.

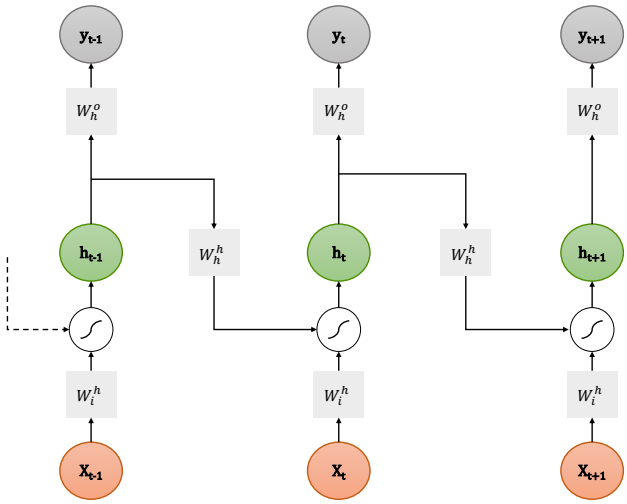


Figure 5.8: RNN unrolled to a feedforward neural network. Each of the inputs $\boldsymbol{x_t}$ and outputs $\boldsymbol{y_t}$ are relative to different time intervals.

To further limit the training process and spare computational capacity, a truncated version of the BPTT algorithm is used for training of the RNN. In the truncated BPTT algorithm the input sequence is processed one timestep at the time and periodically the BPTT update is performed back for a fixed number of timesteps (Brownlee, 2017). The problem with this algorithm is that when the gap between the relevant information and the point of forecasting becomes large, RNNs become unable to connect the relevant information due to the gradient vanishing or exploding. A vanishing gradient refers to the case where the norm of the gradient for long-term components decreases negative exponentially to zero, and gradient exploding refers to the opposite event with exponential growth of weights (Kong et al., 2017).

5.5.3 Long Short-Term Memory Network

The long short-term memory LSTM network is one of the most widely used recurrent neural networks and the idea behind the algorithm was presented in

(Hochreiter & Schmidhuber, 1997). The LSTM model is similar to the general RNN model. However, LSTMs are known for being able to learn long term dependencies in the dataset without facing the problem of vanishing or exploding gradients. The difference is that in the LSTM network the nodes in the hidden layers of the network are replaced by a "memory cell", as illustrated in figure 5.9.

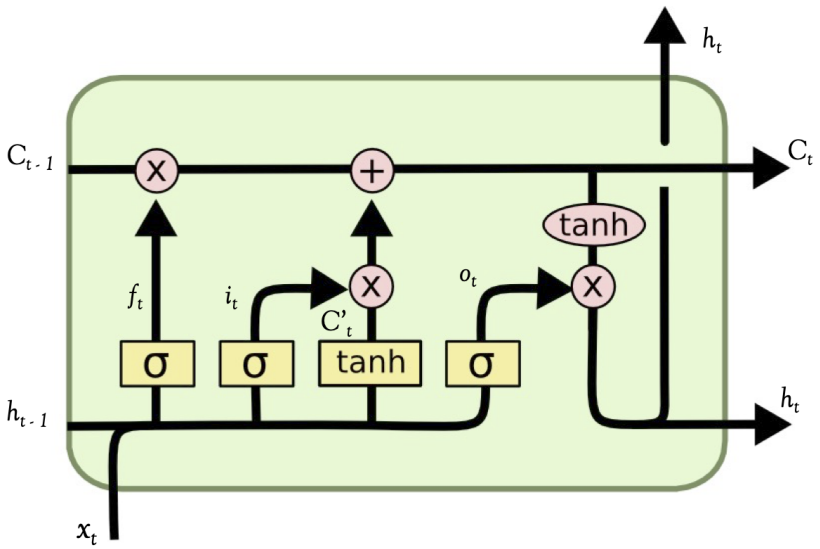


Figure 5.9: Memory cell in LSTM networks [1]

In a memory cell the problems of gradient vanishing or exploding are overcome by its internal state, which is a node with a self-connected recurrent edge with a weight corresponding to the value one. This way the gradient can pass many time-steps without changing the weights (Lipton et al., 2015). In figure 5.9 g_c represents the input node. This node takes activation from the input layer at the current time step and from the hidden layer at time step $t-1$, the summed weighted input is then run through an activation function. i_t represents an input gate. This gate takes the activation from the current data point and from the hidden layer at time step $t-1$. If the value of the activation is one, the value of the gate multiplies the input node, and if it is zero, the flow is cut off. C_t refers to the internal state of the memory cell, where it is decided which elements should be updated, maintained or erased, based on the outputs of the previous time step and input of the current time step. In 2000 the concept of forget gates was introduced by Gers, Schmidhuber and Cummins (Gers, Schmidhuber, &

1. Illustration by MingxianLin, distributed under a CC-BY 2.0 license, retrieved from: https://commons.wikimedia.org/wiki/File:LSTM.jpg

Cummins, 1999). The forget gate is presented as f_t in the figure. They provide the cell with a method for forgetting the content of the internal state, which can be useful in continuously running networks. The use of forget gates was not a part of the original LSTM design, but due to its effectiveness, it has become standard practice in most applications, and therefore it will be included in the model described in this project.

Computations in the LSTM network can be summarized as follows:

$$\begin{aligned} \tilde{C}_t &= \tanh(W_{Cx}\mathbf{x}_t + W_{Ch}h_{t-1} + b_{\tilde{C}}) \\ i_t &= \sigma(W_{ix}\mathbf{x}_t + W_{ih}h_{t-1} + b_i) \\ f_t &= \sigma(W_{fx}\mathbf{x}_t + W_{fh}h_{t-1} + b_f) \\ o_t &= \sigma(W_{ox}\mathbf{x}_t + W_{oh}h_{t-1} + b_o) \\ C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\ h_t &= o_t * \tanh(C_t) \end{aligned} \tag{5.21}$$

Here, $\tilde{C}_t$ represents the input node, and i, f and o represent the input gate, forget gate and output gate, respectively. $W_{gx}, W_{gh}, W_{ix}, W_{ih}, W_{fx}, W_{fh}, W_{ox}$ and W_{oh} are the weight matrices for the networks activation functions and σ represent the sigmoid activation function (Lipton et al., 2015).

Part III

Method

6 Data

The datasets used in this thesis consist of the measured power output from five different wind farms in Northern Norway, and data from a NWP model that includes the temperature at 2m above surface, surface pressure, wind speed at hub height and wind direction at hub height. The power output will act as the dependent variable when making predictions, and the weather data will be provided to the forecasting models as exogenous variables. The measured power output from each wind farm is provided by the Norwegian Water Resources and Energy Directorate (NVE), and the weather data from the NWP model is provided by the Norwegian Meteorological Institute. The data is presented at an hourly resolution from the 1st of January 2017 to the 31st of December 2017. Both the production data and the meteorological forecast data used in this master thesis have been collected and processed by Yngve Birkelund at UiT The Arctic University of Tromsø. A general description of the sites are given in table 6.1 and the locations of the wind parks are shown in figure 6.1.

Name	Capacity (MW)	Location
Raggovidda	45.0	70.76°N/ 29.09°E
Kjøllefjord	39.1	70.92°N/ 27.26°E
Havøygavlen	40.5	71.01°N/ 24.58°E
Fakken	54.0	70.09°N/ 20.08°E
Nygårdsfjellet	32.2	68.50°N/ 17.87°E

Table 6.1: Description of wind farm sites

An individual representation of the five different wind farms and their characteristics is presented in section 6.1. In section 6.2 an analysis of the power output data collected from each wind farm is given, and in 6.3 the dependencies between the exogenous variables and the power output will be explored.

Figure 6.1: Wind parks locations shown in map

6.1 Wind Park Sites

In section 6.1.1 - 6.1.5 the 5 different wind farms and their characteristics are represented.

6.1.1 Raggovidda

The northernmost wind park in figure 6.1 is Raggovidda. The 15 wind turbines in Raggovidda wind farm and some of the surrounding topography of the area is shown in figure 6.2

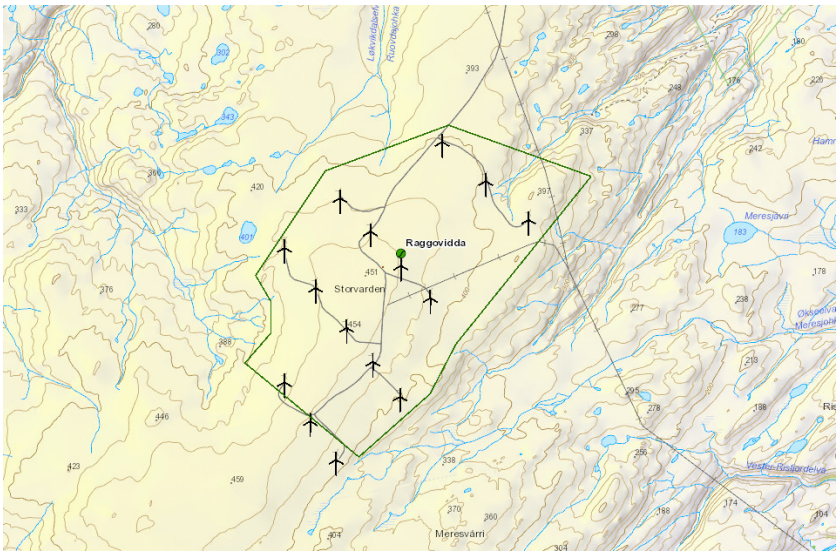


Figure 6.2: Raggovidda wind farm

This wind park has been operative since 2014, which makes it the newest of the five wind farms considered in this thesis. The wind farm consist of 15 wind turbines with an average rotor diameter of 101m. It is located at at about 430m above sea level in a large flat mountain area with good wind resources. The average hub height of all turbines is 80m, and the yearly average power production from Raggovidda wind farm is about 189.0GWh (NVE, 2021).

6.1.2 Kjøllefjord

Kjøllefjord wind farm is located at a low flat mountain area at about 260m elevation. To the east of the wind farm there is a large fjord in the southward direction, and to the north there is another fjord going in the eastward direction. The wind farm consist of 17 wind turbines with an average rotor diameter of 82m. The average hub height of the turbines is 70m. Kjøllefjord wind farm has been operative since 2006 and since then the average yearly wind power production has been 119.0GWh (NVE, 2021).

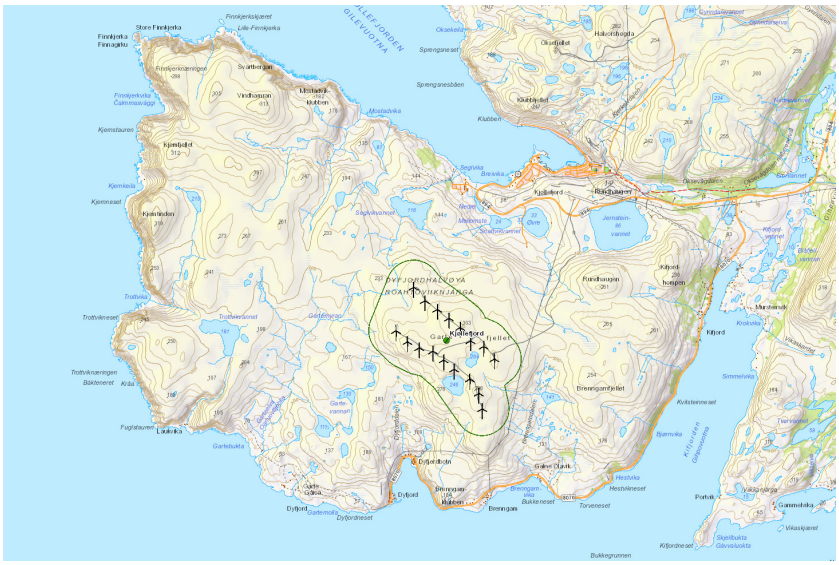


Figure 6.3: Kjøllefjord wind farm

6.1.3 Havøygavlen

Havøygavlen wind farm is situated on a flat low island at about 200m above sea level. The wind farm consist of 16 wind turbines with a rotor diameter of 81.3m. The average hub height is 80m and the average yearly production from the wind farm is 100.0GWh. Havøygavlen has been operating since 2002, which makes it the oldest wind farm considered in this thesis (NVE, 2021).

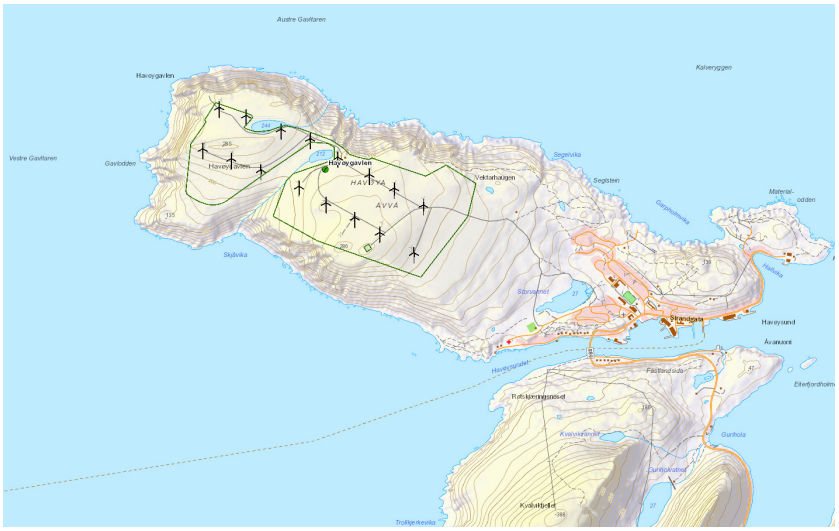


Figure 6.4: Havøygavlen wind farm

6.1.4 Fakken

Fakken wind farm lies on Vannøya island on the coast of Northern Norway. The wind farm consisting of 18 wind turbines lies at 40-200 meters above sea level. The area is surrounded by mountains to the west and the south and open sea to the north. The average rotor diameter of the wind turbines is 90m and the average hub height is 80m. Since 2012, when the wind farm was first established, the average yearly production of the wind farm has been 139.0GWh (NVE, 2021).

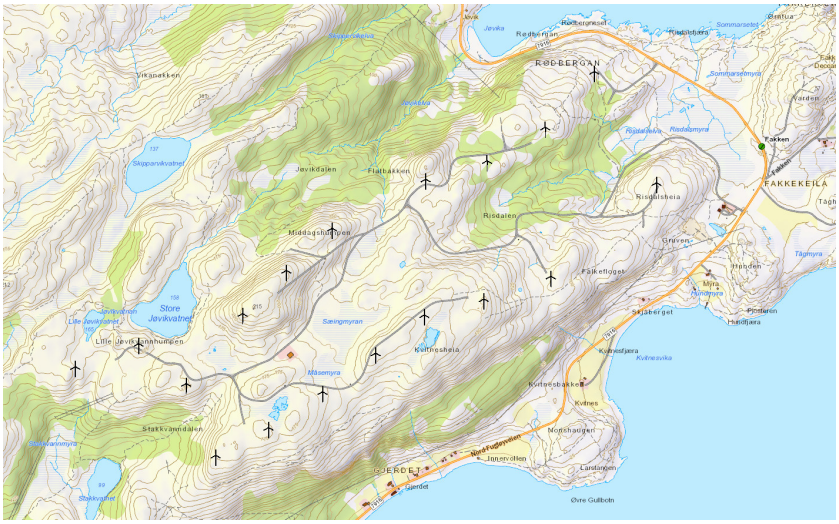


Figure 6.5: Fakken wind farm

6.1.5 Nygårdsfjellet

Nygårdsfjellet wind farm is the southernmost wind farm that is considered in this thesis, shown in figure 6.1. Located at an elevation of 400m with high mountains to the north this wind farm usually sees a high wind from the east during winter months (Birkelund, Alessandrini, Byrkjedal, & Monache, 2018). The wind farm consist of 14 operational wind turbines with an average rotor diameter of 93m. The average hub height of the turbines is 80m. The first three wind turbines at Nygårdsfjellet were installed in 2005, and another 11 were installed in 2011. Since then the yearly average power production from Nygårdsfjellet wind farm has been 104.0GWh.

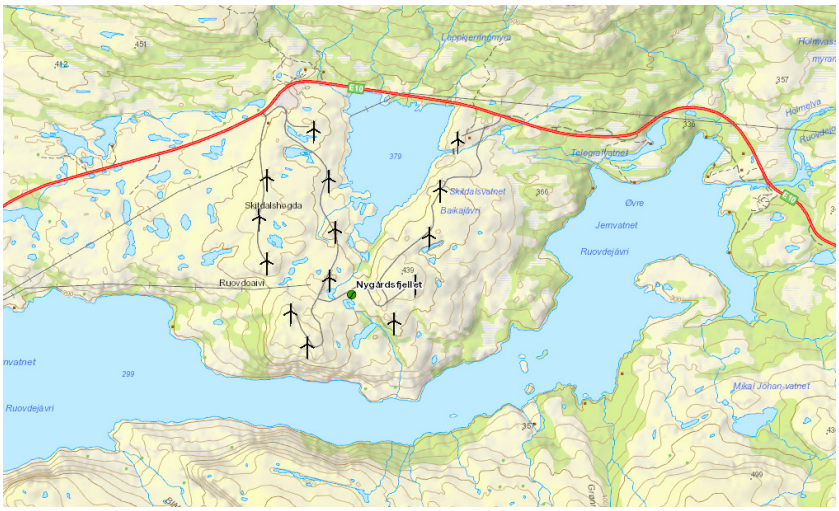


Figure 6.6: Nygårdsfjellet wind farm

6.2 Power Output Data

The historical power output of the five different wind parks are provided by NVE. The data is measured by Statnett at the point of entry to the power grid in MWh, and aggregated at hourly intervals to fit the resolution of the meteorological data. In this thesis the power output data is used both as input to a forecasting model, and for evaluation of the models' performance. As can be seen in figure 6.7, the power output across all five locations are not very different. In general, the power output is lower in summer than it is in winter, which is as expected considering that the wind speeds in summer time is usually lower than in winter. For several years of data this would be equivalent to a yearly seasonal pattern, but for this thesis, where only one year of data is available, no yearly seasonality is present. The dataset is also examined for daily patterns by looking at the hourly box-plot for the entire dataset, as can be seen in figure 6.10. The figure shows that there is no periodicity on a daily basis in the dataset. Considering the similarity of the datasets, the box plot is only provided for one of the locations, the rest of the locations were also checked for daily periodicity, but considering the similarities of the datasets the plots for the rest of the locations are not included here.

The autocorrelation and partial autocorrelation plots for all locations are shown in figure 6.8 and 6.9. From these plots it can be seen how many significant lags there are of the power output for each of the locations, which is helpful when determining the number of lags to give the models.

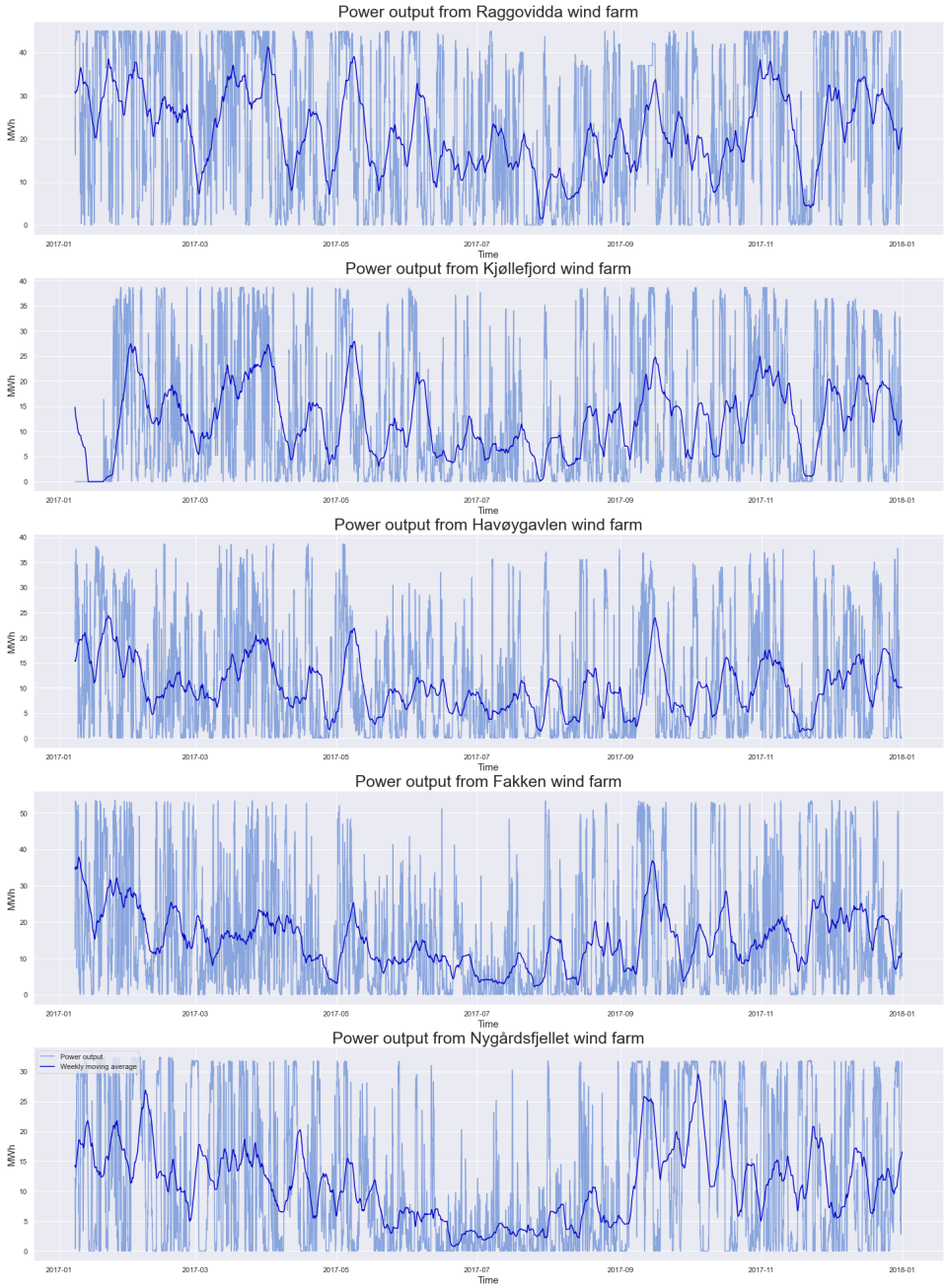


Figure 6.7: The power output as a function of time for all of the five different locations. A weekly rolling mean is also provided in dark blue to better see the pattern of the data.

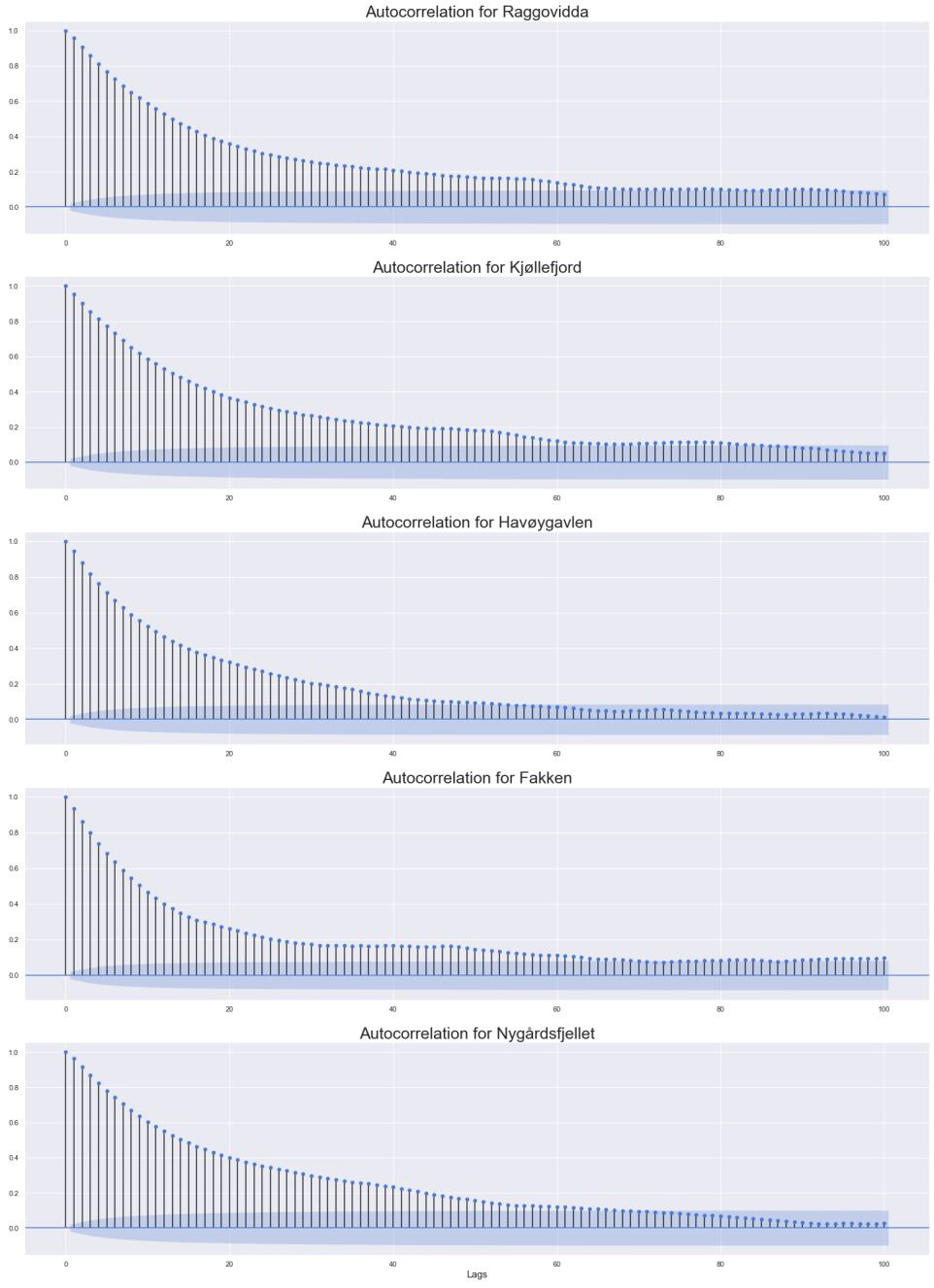


Figure 6.8: The autocorrelation plot for power output across all locations. The autocorrelation plot shows that there is significant correlation between the elements of the power output time series for all locations. This means that the future power output of the wind farms will be dependent on the lagged values of the time series which is helpful when determining the number of input lags to give the prediction models.

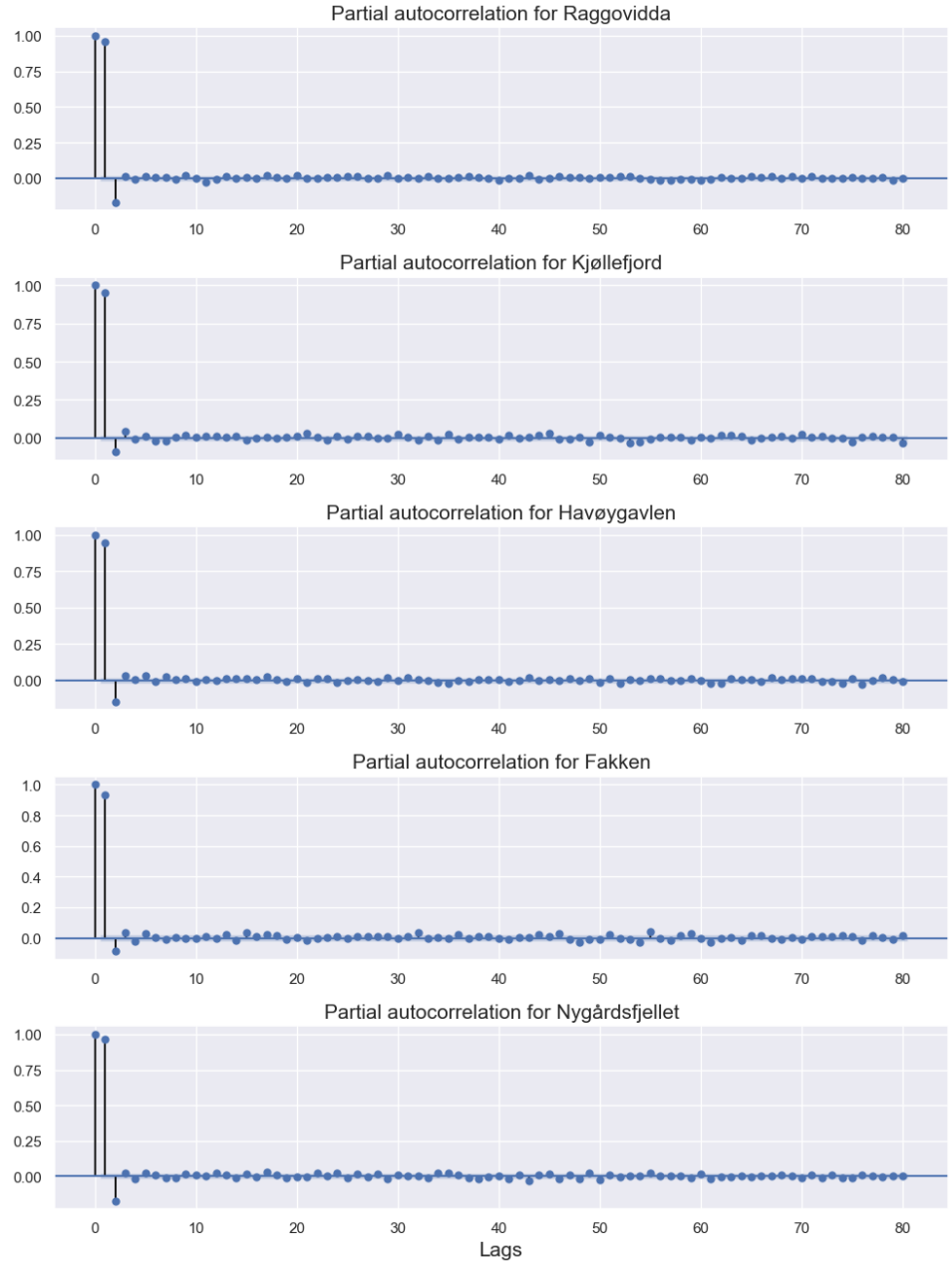


Figure 6.9: The partial autocorrelation plot for power output across all locations. The partial autocorrelation plot is a summary of the relationship between an observation in a time series with observations at prior time steps, but without the relationships of intervening observations. The plot illustrates that there is no correlations for lag values beyond two lags for the power output for all locations. In this thesis this plot is used to help determine the number of lagged timestep values to give the models.

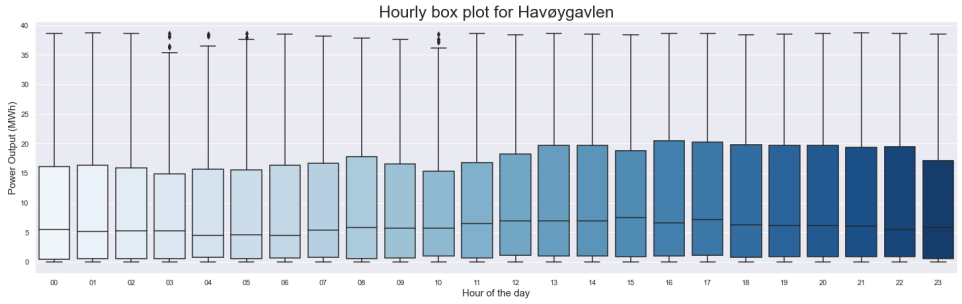


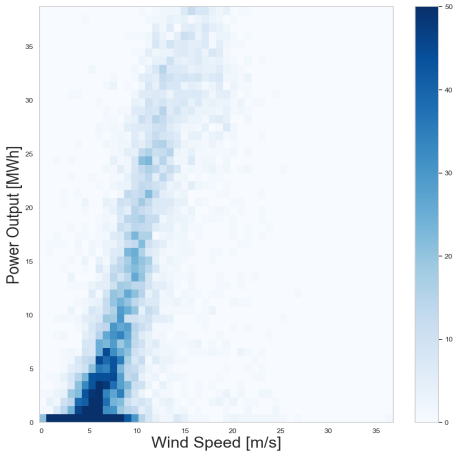
Figure 6.10: The hourly box-plot of power output at Havøygavlen wind farm for examination of daily periodicity in the dataset. It is seen that the power output during a day is fairly constant, and there is no periodicity on a daily level.

6.3 Meteorological Data

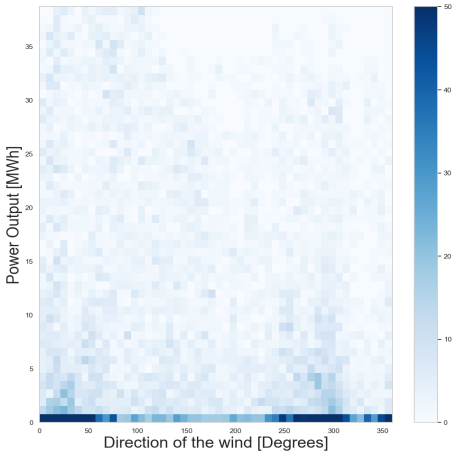
The meteorological data provided by the Norwegian Meteorological Institute is calculated by using the state-of-the-art numerical weather prediction system MetCoOp Ensemble Prediction System (MEPS). MEPS is a cooperation between the Norwegian Meteorological Institute and Swedish Meteorological and Hydrological Institute. The predictions cover both countries with a resolution of 2.5km, using a horizontal grid of 739x949 points centered at 63.5°N and 15°E. Forecasts are issued four times a day at 00.00 am, 06.00 am, 12.00 pm, and 6.00 pm UTC with a 66 hour lead time. The model was the main operational weather forecast model in Norway from 2014 to 2016, and the high-resolution model has been shown to improve the wind, temperature, and precipitation forecast in complex terrain (Birkelund et al., 2018). The data retrieved from the model for use in this thesis includes predictions of wind speed and wind direction in degrees at hub height, temperature two meters above the surface, and surface air pressure. Each sample in the dataset represents the average prediction of the last hour interpolated from a point located close to each wind farm.

It is expected that the wind speed is the variable that will have the strongest relationship with the power output data from the wind farms. This is confirmed in figure 6.11. It is seen in figure 6.11a, where the two-dimensional histogram of the power output at Havøygavlen wind farm and the recorded wind speed is shown, that as the wind speed increases, the power output also increases. The relationship between the power output and the rest of the exogenous variables is also explored. It is seen for the rest of the variables that there is no strong correlation between the power output and the variables, only small indications that the power output at Havøygavlen is higher for low temperatures, for low

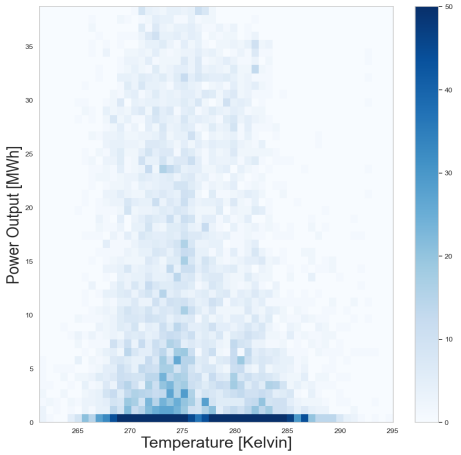
surface pressure, and wind coming from 0 and 360 degrees.



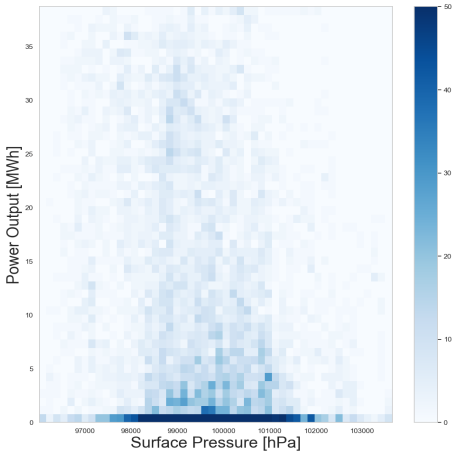
(a) 2D histogram of wind speed and power output



(b) 2D histogram of wind direction and power output



(c) 2D histogram of temperature and power output



(d) 2D histogram of surface pressure and power output

Figure 6.11: 2D histograms of power output vs. the exogenous variables from Havøygavlen wind farm. It is clearly seen that the correlation between wind speed and the power output is the strongest. Small tendencies can be seen in the rest of the plots where the power output is largest for cold temperatures, for wind directions around 0 and 360 degrees and for low surface pressure.

7

Data Preparation

Before the forecasting models are built, the dataset has to be prepared. Missing values are imputed, and feature vectors are processed in order to make it easier for the model to interpret them. This chapter also describes the splitting of data into training and test sets.

7.1 Missing values

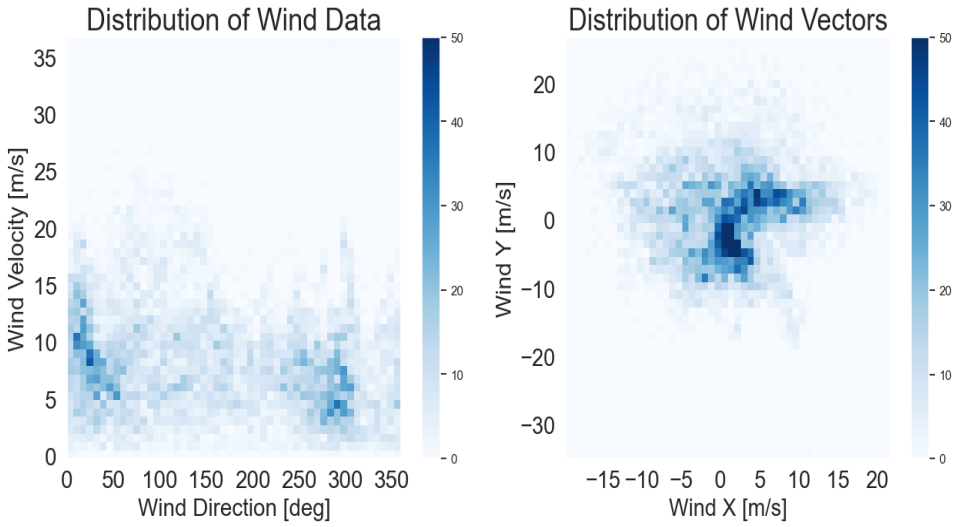
In the dataset there are 17 missing forecasts that are interpreted as missing values. Since the data is at an hourly resolution and the forecasts are made every 6 hours, this means that in total there are missing values at $17 * 6 = 102$ time steps in all the predictor variables. To avoid any loss of information or training data, the missing values are imputed by considering the slope of the time series and the values of the 67 hour ahead forecast that was made at the previous time step. This way the uncertainty of the forecast will be greater for those 102 time steps, but they will be somewhat accurate, and the data can be used for training. The formula used for calculation of the estimated forecast values can be described mathematically as

$$\hat{X}_t = X_{t-1} + \left(\frac{X_{t+n} - X_{t-1}}{y_{t+n} - y_{t-1}} \right) (y_t - y_{t-1}) \tag{7.1}$$

Here X refers to the actual time series where every 6th hour a new forecast is made, and y is the 67h forecast made at time step X_{t-1}.

7.2 Feature Engineering

The wind data in the dataset used in this thesis includes the direction of the wind in degrees and the wind speed in m/s. In this format the pattern between predictor variables and input variables may be difficult to interpret since 0 and 360 seems to be very different, but in fact they are very close to each other. As an example, the distribution of the wind data from Havøygavlen using the unprocessed dataset is shown in figure 7.1a. Since angles in degrees does not provide a strong signal for a learning model, the wind speed and the wind direction features are combined and converted from polar to Cartesian coordinates, so that a stronger signal can be interpreted from the wind data, this is done for all the wind farm locations. The distribution of wind data after feature engineering at Havøygavlen wind farm can be seen in figure 7.1b.



(a) Before feature engineering

(b) After feature engineering

Figure 7.1: Distribution of wind data for Havøygavlen before and after feature engineering

7.3 Training, Validation and Test Data

When creating training, validation and test sets from the time series data, the temporal components inherent in the data have to be considered. Traditional methods for splitting a dataset with independent observations into training, validation, and test data are not feasible for this forecasting task, as they assume that there is no relationship between observations and that each observation is independent of another. Typically, such methods rely on shuffling the dataset

so that a similar distribution of instances can be seen in both the training and test data. In time series the data must be split in such a way that it respects the temporal order of the observations. In a sequential dataset there might be different properties of the data across different periods of the year. Differences in the dataset might lead to some parts of the dataset that are easy to learn from and some parts of the dataset are difficult to learn from. If the model is trained on a dataset that is easy to learn from, but tested on a part of the dataset that is hard to understand, the performance of the model might yield suboptimal results that do not have anything to do with the overall performance of the model. For example, if the model was only trained on instances from the summer months, it might not perform very well when tested on the test dataset containing only winter months. To make sure the test results are representative of the overall model performance, the dataset is divided into 4 different splits for training, validation and test data as described in table 7.1.

	Training	**Validation**	**Test**
Split 1	January - June	July - September	October - December
Split 2	April - September	October - December	January - March
Split 3	July - December	January - March	April - June
Split 4	January - March + September - December	April - June	July - September

Table 7.1: Splitting of dataset into training, validation and test sets

Another aspect to consider is the randomness of each of the models indicating slightly different results on each run. In neural networks this randomness might be introduced by randomly initiation of weights, while in the RF forest model the trees are randomly generated in each run, hence the model results will rarely be identical even when provided with the same data. By splitting the dataset into four different splits and training the model a number of times, the average performance of the 4 runs can be recorded as the overall performance of the model. To have a closer look at the distribution of the training, test and validation data, a distribution plot of the data from Kjøllefjord wind farm is shown in figure 7.2. In the figure it can be seen that for some of the train-validation-test splits of the dataset there is an uneven distribution of zeros.

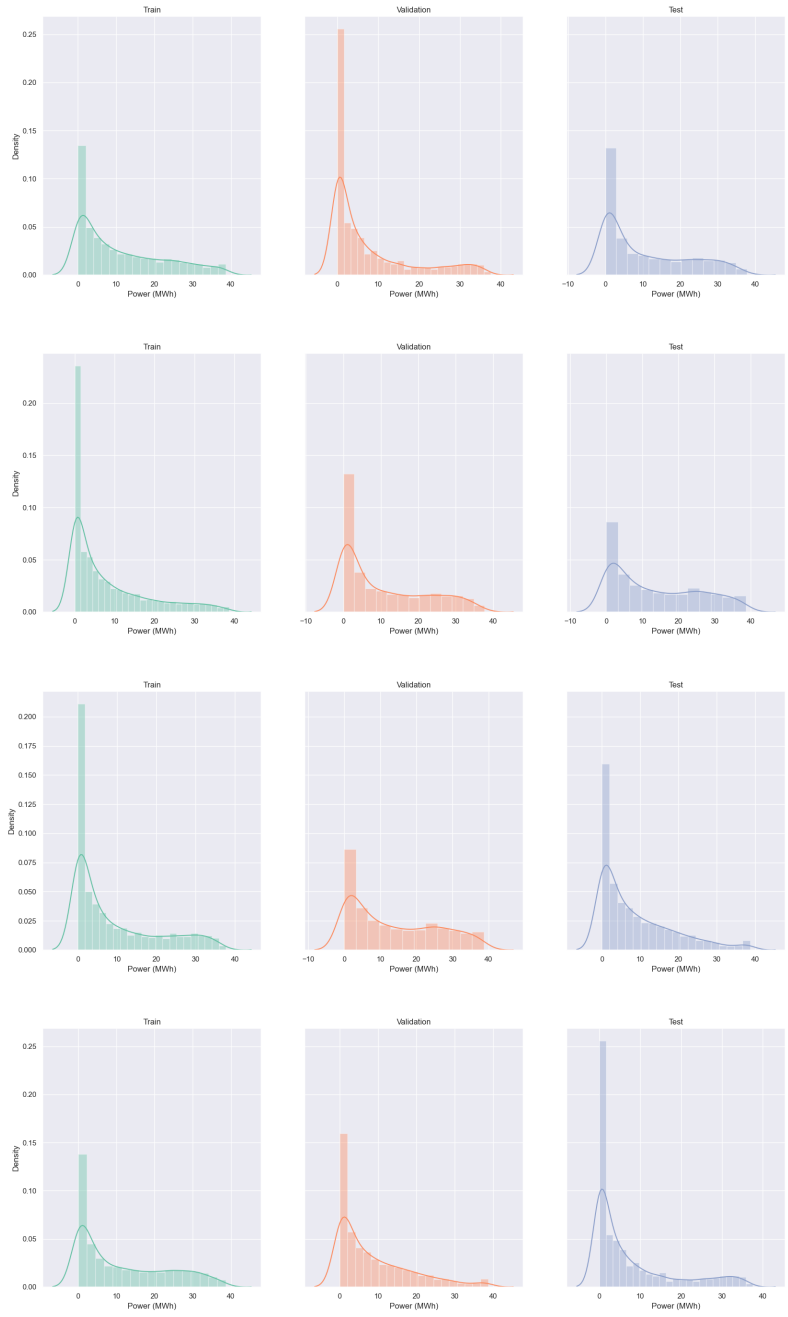


Figure 7.2: Distribution of train, validation and test data for four different dataset splits at Kjøllefjord wind farm

7.4 Sliding Window Representation

A time series dataset must be made fit for a supervised learning algorithm, which requires that the data is provided as a collection of samples where each sample has an input component and a label. The input components are often called X and the labels are referred to as y. The model will then learn how to translate the input components X into the output components y. A time series dataset can be transformed into a supervised learning problem by using the so called sliding window method. In this method the values at a prior time step are used as input to the model and the value at the current time step is used as a label. The values from the prior time step is often referred to as the lag observations. Using this method one ensures that the temporal dependence between observations are obtained, and will be preserved while training the model (Brownlee, 2019).

7.5 Normalization

Data normalization refers to the act of taking every sample in the dataset and transforming the values from their natural range to the models operating range (Palit & Popovic, 2006). Normalizing the dataset is helpful in reducing the training time of a neural network as well as avoiding the problem of exploding gradients. In some cases where the forecasting model uses the Euclidean distance, it may lead to better results if all the input features are scaled as opposed to providing the data with raw values. If the features of the dataset include attributes on different numeric ranges, the normalization of the dataset helps in avoiding the attributes in greater numeric ranges to dominate attributes in smaller numeric ranges (Brownlee, 2019). The normalization used for transforming the dataset in this thesis is calculated as

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{7.2}$$

where X' is the normalized dataset, X refers to the original dataset, X_{min} and X_{max} are the lowest and the largest value in the dataset, respectively. When normalizing the dataset, only the statistics from the training data is used, so that the model will have no access to the values in the validation or the test sets. The validation and the test data is then normalized according to the statistics of the training data.

8

Implementation

This chapter will describe the methodological decisions made while implementing the forecasting models that were described in chapter 5. An essential part of building models for time series forecasting is the tuning of hyperparameters for each model. In sections, 8.1-8.4 the choice of hyperparameters and other specific decisions made regarding the implementation of the methods are introduced. For each of the forecasting methods a model was fit for each of the splits of the dataset from each of the wind farm locations. The number of lagged values of the data the models received is determined by looking at the ACF and the PACF plots in chapter 6 in figures 6.8 and 6.9. It is seen in the plots that the future power output is highly correlated with previously observed values of the time series. To avoid giving the models too much information it is decided that the input lags that will be used in the models for each of the locations is set to two timesteps. The models were also tested with the use of as many significant lags that are shown in the autocorrelation plot for each location, and with 12 and 24 lags. However, the more input lags the models received the worse they performed which will be further discussed in chapter 10.

The ARIMAX model was implemented using the statsmodels library in Python. The random forest model and the support vector regression model were implemented using the scikit learn library, and the neural networks were implemented using TensorFlow and Keras.

8.1 ARIMAX

The implementation of the ARIMAX model for every wind farm location was done by choosing the appropriate values for the p, d, and q parameters. Selection of the parameters was done for every split of the dataset using only the training data to ensure that no knowledge of the test data was given to the model. The order of differencing d was determined by applying the ADF and the KPSS test to the data to check for stationarity. The tests were conducted on both the dependent variable and the exogenous variables. For all variables, the ADF test indicated stationary timeseries while the KPSS test implied that not all variables were stationary. Hence, differencing all variables was done once, which resulted in both the tests implying stationary data. However, the ADF test and the KPSS test only check the data for trends in the dataset. An autocorrelation plot of each of the variables in the datasets, as shown in figure 8.2, showed that some periodicity was still present in the temperature data, implying that the dataset was not fully stationary. However, removing the seasonality in the dataset might lead to exaggerated differencing of the dependent variable. Considering that the dependent variable was stationary after one differencing, the order of differencing d was set to $d = 1$. According to common practice (Brockwell & Davis, 2016), the p parameter was determined by looking at the partial autocorrelation function (PACF) plot, as shown in figure 8.1, where p is equal to the number of significant lags in the plot. This same logic was applied for selection of the q parameter of the model, only for this parameter the number of significant lags in the ACF plot was the determining factor.

In chapter 6 it was seen that the statistical properties of the datasets were fairly similar, which was confirmed when implementing the ARIMAX model. For every set of training data of each location, the same amount of preprocessing was needed to obtain stationary series. In this section only the plots for Havøygavlen wind farm is shown, but the statistical properties of the data is representative for all locations. The autoregressive parameter p and the moving average parameter q also became the same for all the training datasets. Eventually, the parameters that were chosen for the ARIMAX models were $p = 2$, $q = 2$, $d = 1$.

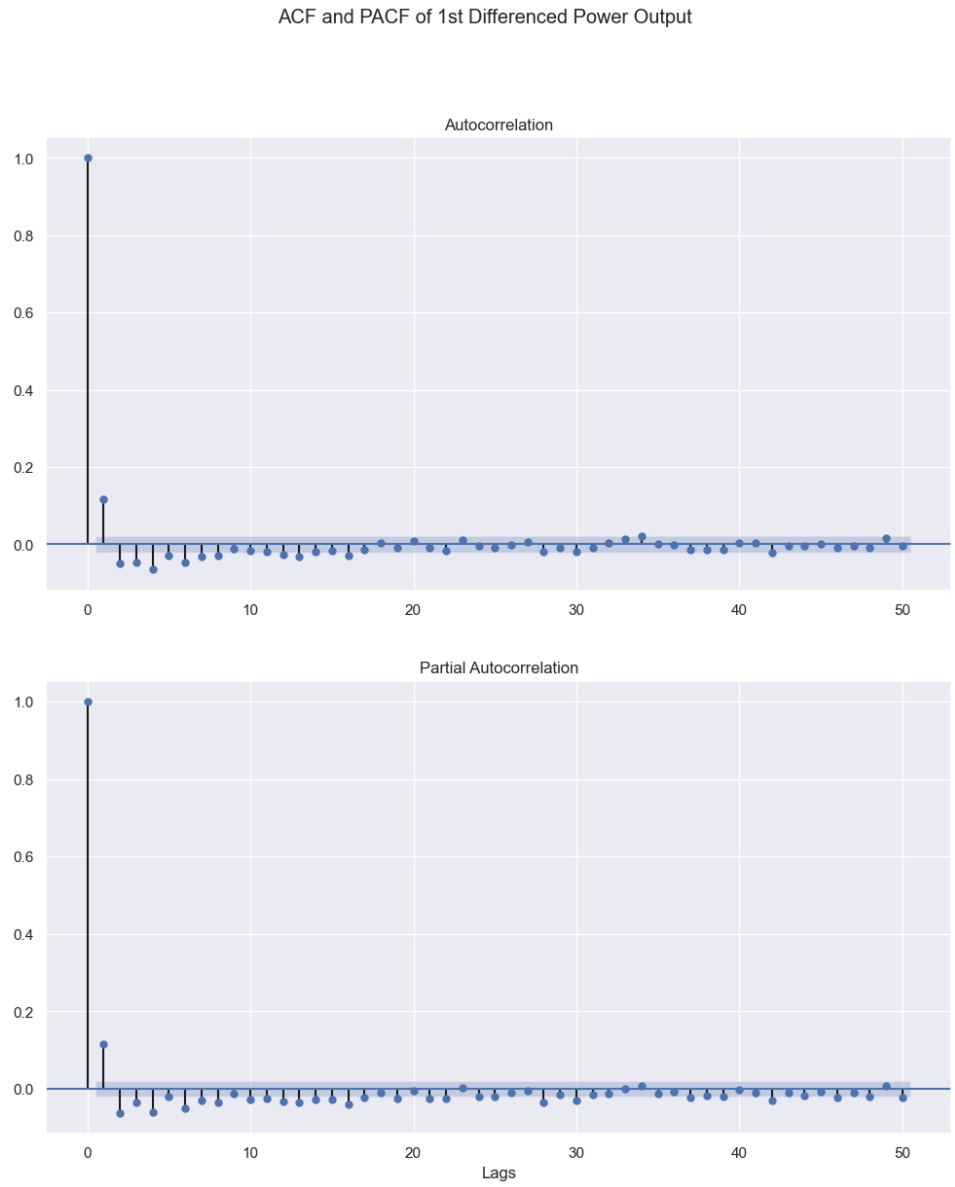


Figure 8.1: ACF and PACF of first difference of dependent variable that is used for determining the p and q parameters of the ARIMAX model.

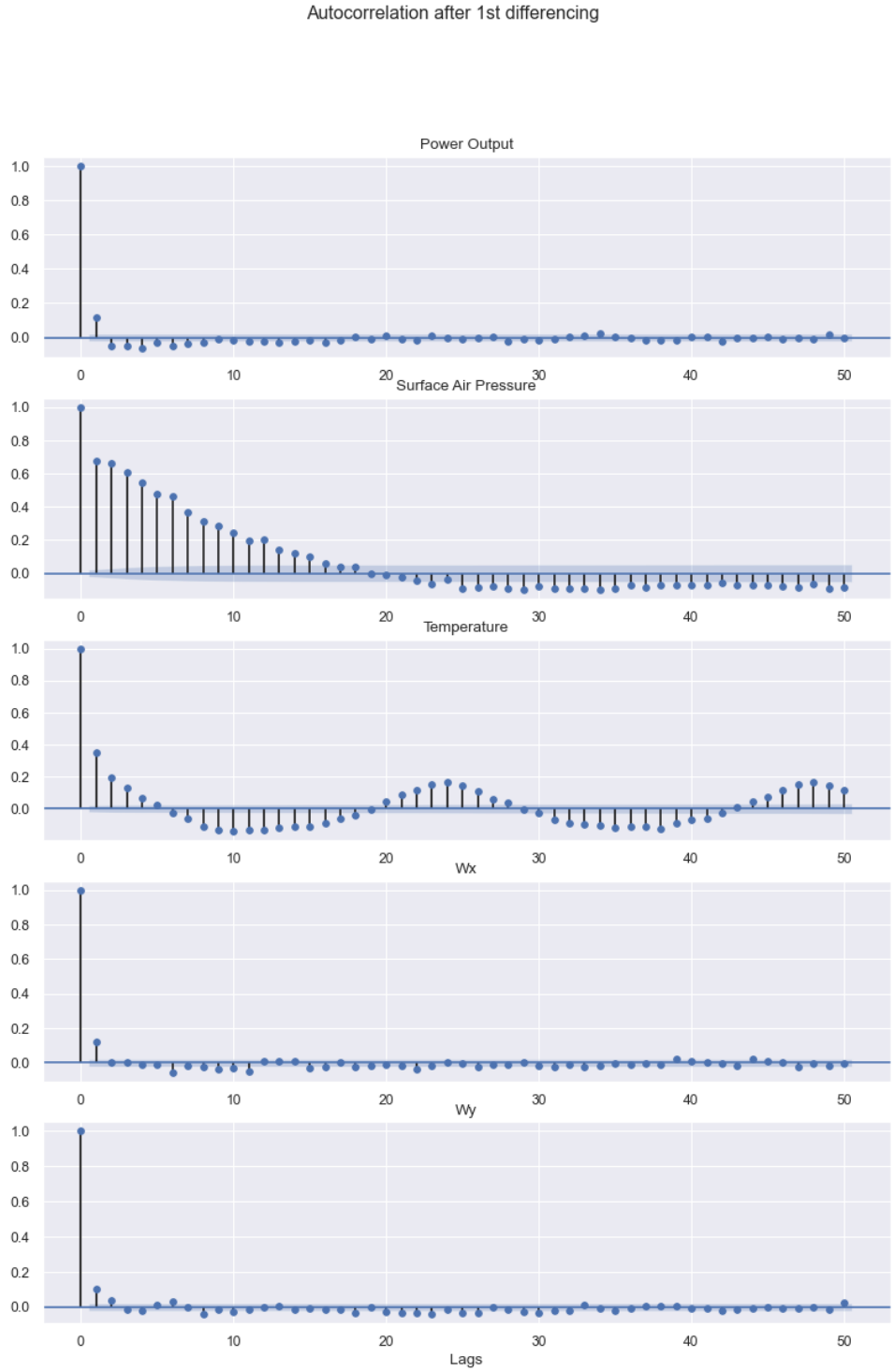


Figure 8.2: Autocorrelation plot of all variables from Havøygavlen wind farm after first differencing of the dataset.

8.2 Random Forest

As described in chapter 5, the performance of the random forest model can be optimized by tuning several hyperparameters. The main parameters being the number of features considered in each node (m), the number of trees in the forest (T), the elements in a node required to perform a split (M_s), the elements required to create a node (M_l) and the maximum depth up to which a tree can grow (L). The number of features to be considered in each node when looking for the best split was set to $m = \sqrt{P}$ according to the rule of thumb as suggested by (Breiman, 2001), where P is the total number of predictor variables. The maximum depth of the trees was not modified, so the nodes were all expanded until they contained the minimum number of datapoints required to create an end node. Instead, to avoid overfitting, the number of elements in a node required to perform a split (M_s), the number of elements required to create a leaf (M_l) and the number of trees in the forest (T) were determined by a grid search. Since M_l is more restrictive than M_s, it is reasonable to suggest values for $M_l < M_s$ (Babar et al., 2020). At last, the range of values in the grid search for M_l was set to $[1, 10]$, the range of M_s was set to $[2, 12]$ and the choice of T was set to $50, 100, 200, 300, 400, 500$. The combination of parameters that were chosen for each split of the dataset and each wind farm location is presented in table 8.1.

		Number of trees (T)	**Min samples split** (M_s)	**Min samples per leaf** (M_l)
Raggovidda	Split 1	300	2	2
	Split 2	300	2	2
	Split 3	200	4	4
	Split 4	400	8	2
Kjøllefjord	Split 1	300	2	2
	Split 2	400	2	2
	Split 3	100	4	1
	Split 4	100	2	2
Havøygavlen	Split 1	100	2	2
	Split 2	100	4	4
	Split 3	50	8	2
	Split 4	100	2	2
Fakken	Split 1	50	2	2
	Split 2	100	2	2
	Split 3	100	2	2
	Split 4	300	2	2
Nygårdsfjellet	Split 1	200	2	2
	Split 2	500	2	2
	Split 3	500	2	2
	Split 4	50	2	2

Table 8.1: Hyperparameters for random forest model

8.3 Support Vector Regression

As described in chapter 5, the SVR model is based on mapping of the input data into a higher-dimensional feature space by a nonlinear mapping function and linear regression. The mapping operation of the input data is done by choosing an appropriate kernel function. A commonly used kernel function for SVR is the Gaussian radial basis function (RBF), which will be used in this thesis as well. The RBF is given by

$$K(x, x_i) = \exp^{-\gamma(\|x-x_i\|^2)} \tag{8.1}$$

The kernel function can be understood as a similarity measure between two points that yields the dot product of those two points in the transformed feature space. The γ parameter in the RBF is a tuneable parameter in the SVR model. It determines the variance of the kernel function in (8.1). In the case of a large variance in the input data, when γ is low, two points may be considered similar even if they are far away from each other. If γ is large, the variance of the RBF kernel and in the associated transformed feature space will decrease, and two points will be considered similar only if they are very close to each other in input space. Other important hyperparameters to tune in the SVR model is the C parameter and the ϵ parameter. The C parameter is referred to as the regularization parameter. It is used in the SVR model to prevent the model from overfitting the training data. The penalty of the C parameter is the L2 penalty, as described in chapter 5. The ϵ parameter specifies the size of the epsilon tube, as illustrated in figure 5.3. To determine the values of γ, C and ϵ, a grid search was run for each location and each split for the SVR model. The grid search included the choice of values in the set $\{0.001, 0.01, 0.1, 'scale', 'auto'\}$ for the γ parameter. Here 'auto' is equal to $1/m$, where m is the number of features in the dataset, and 'scale' represents $1/(m * Var(X))$, where $Var(X)$ is the variance of the training data. The range of the C parameter was set to [1,6], and the range of the ϵ parameter was set to [0.001,0.03]. The combination of hyperparameters that was used for each location and each split in the SVR model is presented in table 8.2.

		Regularization C	Epsilon tube ϵ	Variance γ
Raggovidda	Split 1	3	0.02	'auto'
	Split 2	6	0.02	0.1
	Split 3	3	0.01	0.001
	Split 4	3	0.01	0.001
Kjøllefjord	Split 1	6	0.02	0.01
	Split 2	3	0.001	'auto'
	Split 3	5	0.02	0.001
	Split 4	6	0.02	0.1
Havøygavlen	Split 1	3	0.02	'auto'
	Split 2	6	0.01	0.1
	Split 3	6	0.02	0.01
	Split 4	6	0.001	0.1
Fakken	Split 1	3	0.02	'auto'
	Split 2	3	0.02	0.01
	Split 3	6	0.01	0.1
	Split 4	6	0.01	0.1
Nygårdsfjellet	Split 1	6	0.001	'auto'
	Split 2	4	0.02	0.01
	Split 3	6	0.02	0.1
	Split 4	3	0.02	0.01

Table 8.2: Hyperparameters for SVR model

8.4 Long-Short Term Memory Network

The main advantage of the LSTM model in sequence modelling is its ability to learn long term dependencies in the dataset without facing the problem of vanishing or exploding gradients. When implementing the LSTM model in TensorFlow, the model has to be specified as a 'stateful' model in order for it to preserve the internal state of the network throughout the training process. In doing so the batch size of the model has to be a common denominator of the size of the training data, the validation data and the test data. As a consequence, a few samples has to be dropped from the dataset for training of the network in order to operate with the desired batch size. The training data for the LSTM model will therefore not be exactly the same as the training data for the rest of the models implemented.

In chapter 5 it was stated that the performance of a neural network model heavily depends on the choice of optimizer, the loss function, the network architecture and the choice of regularization techniques. In this thesis the

Adam optimizer will be used in both the neural network models, and to avoid overfitting of the network a dropout layer is used in between every hidden layer in the network as the regularization technique. The loss function will be the MSE function, and the activation function of the network will be the ReLu activation function, which was shown in figure 6.11c. Experimental results have shown that the ReLu activation function tends to outperform the sigmoid and the tanh function when used in neural networks (Gonzalez & Woods, 2018). Hence ReLu activation function will be used in this thesis. The training of an LSTM network, as with any other deep learning method, is usually a time consuming task. Due to computational limitations and time constraints some adjustments had to be done when implementing the LSTM model in this thesis. It was noticed that the training time of the network became so large that it was decided that the LSTM model would only be tested for Havøygavlen wind farm, the reasons for this will be explained in later chapters. The methodological decisions that were made on the basis of speeding up the training time was to set the batch size to 128, which is often faster as opposed to using smaller batch sizes. The rest of the hyperparameters, including the number of hidden layers in the network, the number of neurons in each hidden layer, the dropout rate and the learning rate, was found by grid searching the model for each split of the dataset, the chosen hyperparameters are presented in table 8.3.

		Learning rate	Batch size	Dropout rate	Hidden layers	Neurons per layer
Havøygavlen	Split 1	0.00001	128	0.1	2	64
	Split 2	0.0001	128	0.1	2	64
	Split 3	0.00001	128	0.2	3	64
	Split 4	0.0001	128	0.5	3	64

Table 8.3: Hyperparameters for LSTM model for Havøygavlen Wind Farm

Part IV

Results and Discussion

9

Experiments and Results

This chapter will present the experiments done for assessing the performance of the different forecasting models and their performances. The RF model and the SVR model were both used with the direct and the recursive forecasting methods as described in chapter 4. Due to limitations in computational capacity and time constraints related to the submission deadline, the LSTM model was used only for one of the wind farm locations and for three different time horizons. According to the performance of the persistence model for 1-hour ahead predictions on all wind farm locations it was shown that the model had the worst performance on Havøygavlen. The persistence model can in some cases be used as measure of the complexity of the dataset. Because Havøygavlen seemed to be the wind farm location that was the most difficult to predict, the LSTM model was tested on this location.

For every location and every split of the dataset, predictions were made for all the forecasting horizons. The MAE, RMSE, and the NRMSE for every forecasting horizon presented in tables 9.1-9.5 are the average performance for each split of the datasets. This way, some of the uncertainty in using non-deterministic models is eliminated, and the models are evaluated on different seasons of the year rather than for one particular time of the year.

The experiment results are first presented for each of the wind farm locations in sections 9.1 - 9.5, and in section 9.6 the overall results of the different models are shown.

9.1 Raggovidda Wind Farm

The results from Raggovidda wind farm show that the best performing model across all time horizons is the recursive SVR model, while the worst performing model is the direct RF model, especially at large forecasting horizons. All of the implemented models, except the ARIMAX and the direct RF model, outperform the persistence model at forecasting horizons $h > 12$. This can be seen in figure 9.2 and figure 9.3. The average results across all the dataset splits, from all models and all forecasting horizons are shown in table 9.1.

Results From Raggovidda Wind Farm											
Method	Error Metric	Forecasting Horizon									
		h=1	h=2	h=3	h=4	h=5	h=6	h=9	h=12	h=18	h=24
Persistence	MAE	0.64	0.85	1.66	2.53	3.19	2.97	2.40	2.94	6.70	7.62
	RMSE	1.54	3.00	4.18	4.93	5.53	5.85	4.98	4.68	7.41	7.13
	NRMSE	0.03	0.05	0.10	0.15	0.19	0.18	0.16	0.22	0.46	0.50
RF(Recursive)	MAE	6.44	4.99	3.80	3.17	3.75	4.10	3.93	4.23	4.10	4.37
	RMSE	6.44	5.22	4.39	3.89	4.78	5.40	4.97	5.16	5.14	5.56
	NRMSE	0.30	0.24	0.20	0.18	0.22	0.25	0.22	0.24	0.24	0.26
RF(Direct)	MAE	4.07	3.96	3.11	3.39	4.69	5.64	6.73	8.04	10.36	12.06
	RMSE	4.07	3.97	3.38	3.64	5.51	6.82	8.08	9.47	12.18	13.84
	NRMSE	0.19	0.18	0.15	0.17	0.25	0.31	0.37	0.43	0.56	0.64
SVR(recursive)	MAE	2.60	3.37	2.92	3.00	3.73	4.18	3.65	3.56	3.69	3.85
	RMSE	2.60	3.54	3.31	3.30	4.95	5.68	5.05	4.76	5.05	5.48
	NRMSE	0.12	0.16	0.15	0.15	0.23	0.26	0.23	0.22	0.23	0.25
SVR(Direct)	MAE	2.69	1.86	2.43	2.99	4.02	4.21	3.86	4.07	5.80	6.79
	RMSE	2.69	2.08	2.85	3.80	5.10	5.38	5.17	5.27	7.75	9.04
	NRMSE	0.12	0.10	0.13	0.17	0.23	0.25	0.24	0.24	0.36	0.42
ARIMAX	MAE	3.63	3.43	3.45	3.62	4.30	5.04	6.40	7.62	7.78	9.54
	RMSE	3.63	3.78	4.17	4.37	5.02	5.85	7.54	8.71	9.11	11.03
	NRMSE	0.17	0.17	0.19	0.20	0.23	0.27	0.35	0.40	0.42	0.51

Table 9.1: Results from Raggovidda wind farm. The results show that the persistence model is the best performing model for short forecasting horizons, but when $h > 12$ the persistence model is outperformed by the recursive RF and SVR and the direct SVR.

From the table it can be seen that the direct approaches of the RF and the SVR models have a low error across all error measures for short forecasting horizons, and the error increases as the forecasting horizon gets bigger. Figure 9.1 shows the 24 hour predictions for one of the test sets. It can be seen that both the recursive forecasting approaches do a better job in approximating the actual power output of the wind farm, whereas the direct approaches seem to estimate some trend or overall average of the power output. This indicates why the recursive forecasting method outperforms the direct method for large forecasting horizons. The reason might lie in the different way the models are built when using the recursive vs. the direct forecasting approach, which will be further discussed in chapter 10. It is also noticeable that the ARIMAX model

follows the pattern of the power output for short time horizons, but when the forecasting horizon increases the model accuracy decreases.

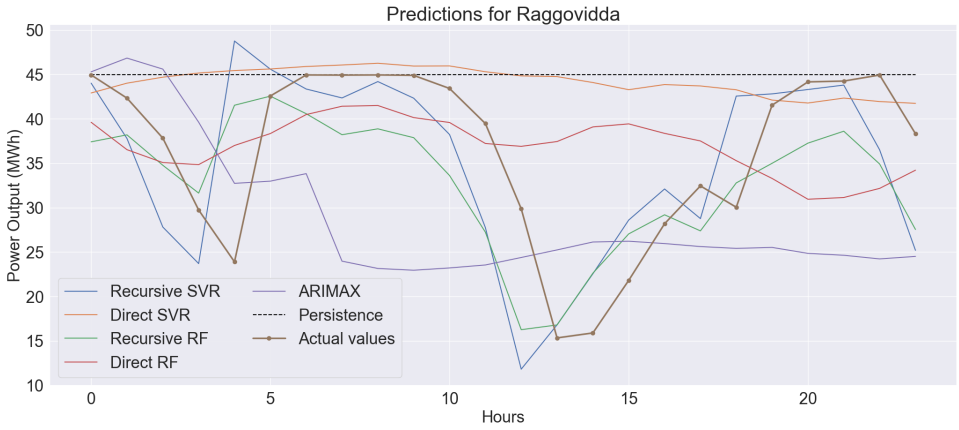


Figure 9.1: 24 hours predictions for Raggovidda wind farm from the different forecasting models on one of the test datasets.

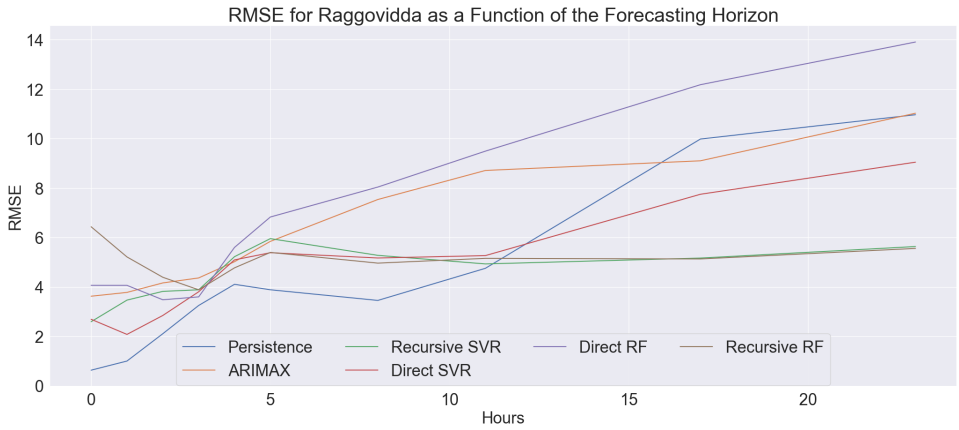


Figure 9.2: RMSE as a function of the forecasting horizon for Raggovidda

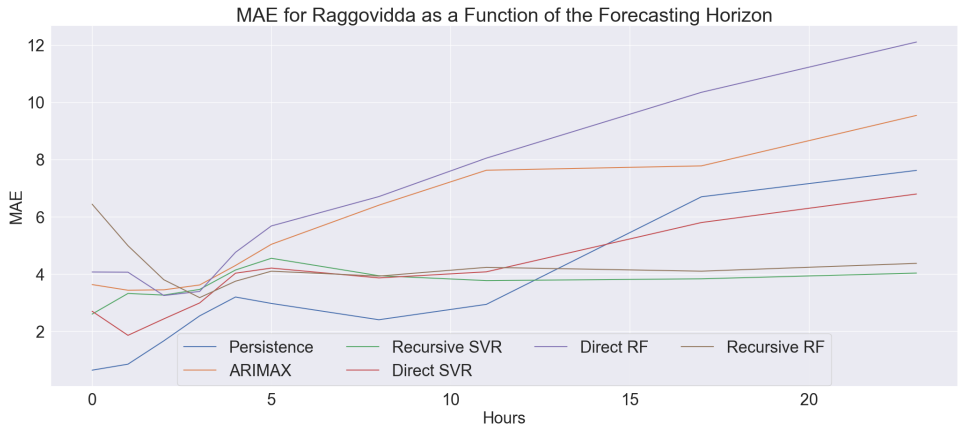


Figure 9.3: MAE as a function of the forecasting horizon for Raggovidda

9.2 Kjøllefjord Wind Farm

The results from Kjøllefjord wind farm show that the recursive SVR model yields the best results for time horizons $h > 3$ in terms of all the error measures, which can be seen in figures 9.5 and 9.6. The results for all forecasting horizons and all the different forecasting models are shown in table 9.2.

Results From Kjøllefjord Wind Farm											
Method	Error Metric	Forecasting Horizon									
		h=1	h=2	h=3	h=4	h=5	h=6	h=9	h=12	h=18	h=24
Persistence(Recursive)	MAE	1.21	4.17	6.41	7.16	6.73	6.04	5.56	6.38	8.47	8.38
	RMSE	1.98	6.38	8.63	9.22	9.18	8.53	8.19	9.20	11.59	11.34
	NRMSE	0.10	0.45	0.63	0.67	0.63	0.58	0.55	0.65	0.82	0.79
RF(Recursive)	MAE	5.82	4.62	5.01	5.21	4.64	4.47	4.74	4.49	3.99	4.09
	RMSE	5.82	4.82	5.23	5.38	4.99	4.87	5.25	5.11	4.73	4.99
	NRMSE	0.47	0.39	0.42	0.42	0.44	0.39	0.43	0.41	0.38	0.40
RF(Direct)	MAE	6.21	5.12	6.05	6.31	5.88	6.02	7.09	7.23	7.46	7.85
	RMSE	6.21	5.32	6.45	6.68	6.66	6.72	8.35	8.54	8.67	9.01
	NRMSE	0.50	0.43	0.52	0.54	0.54	0.54	0.68	0.69	0.70	0.73
SVR(recursive)	MAE	6.78	5.31	4.09	4.06	3.79	3.88	3.91	4.01	3.62	3.69
	RMSE	6.78	5.73	4.77	4.62	4.42	4.60	4.58	4.75	4.41	4.48
	NRMSE	0.55	0.46	0.38	0.37	0.36	0.37	0.37	0.38	0.36	0.36
SVR(Direct)	MAE	7.22	4.37	4.76	4.69	4.41	4.72	5.61	5.79	5.56	6.56
	RMSE	7.22	5.22	5.55	5.38	5.12	5.31	6.43	6.68	6.63	7.86
	NRMSE	0.58	0.42	0.45	0.44	0.41	0.43	0.52	0.54	0.54	0.64
ARIMAX	MAE	9.78	12.05	13.77	14.23	13.50	12.73	12.39	11.36	10.67	10.02
	RMSE	9.78	13.17	14.96	15.19	14.60	14.00	13.72	12.88	12.34	11.75
	NRMSE	0.79	1.07	1.21	1.23	1.18	1.13	1.11	1.04	0.99	0.95

Table 9.2: Results from Kjøllefjord wind farm. The results show that the persistence model is the best performing model for forecasting horizons $h < 3$. When $h > 3$ the persistence model is outperformed by both the recursive and direct implementations of the RF and SVR model.

For one-hour-ahead predictions the persistence model is the best performing model, but when $h = 2$, the persistence model is outperformed by the direct SVR and both the RF models in terms of the NRMSE. The worst performing model for Kjøllefjord wind farm is the ARIMAX model that is not competitive with the persistence model for any time horizons. In figure 9.4 it can be seen that the ARIMAX model indeed does a bad job in approximating the power output at Kjøllefjord wind farm for any time horizons. However, the persistence model is acting very poorly on this particular test dataset as well. The results in table 9.2 is the average results across all splits of the dataset, and possibly the persistence model performs better on the rest of the year. The recursive SVR model is the best at estimating the pattern of the power output, closely followed by the recursive RF model. However, it seems as if the RF underestimates the power output in the peak values, and over estimates in the minimum values.

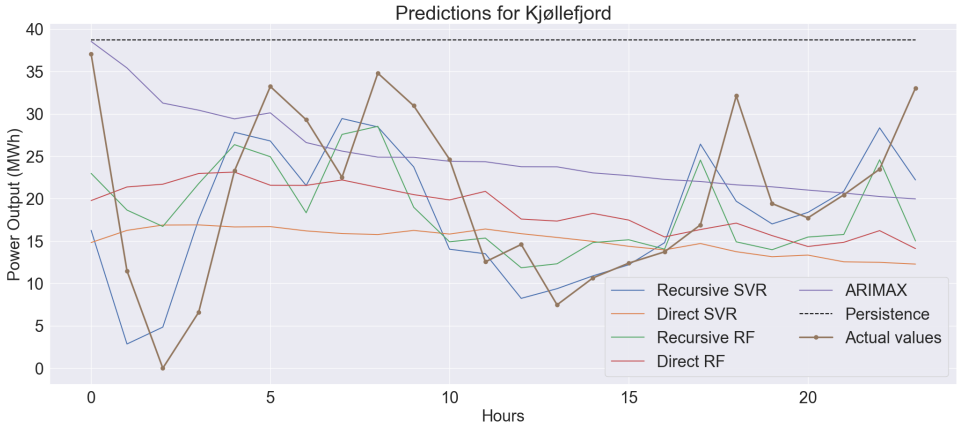


Figure 9.4: 24 hours predictions for Kjøllefjord wind farm from all the different forecasting models on one of the test datasets.

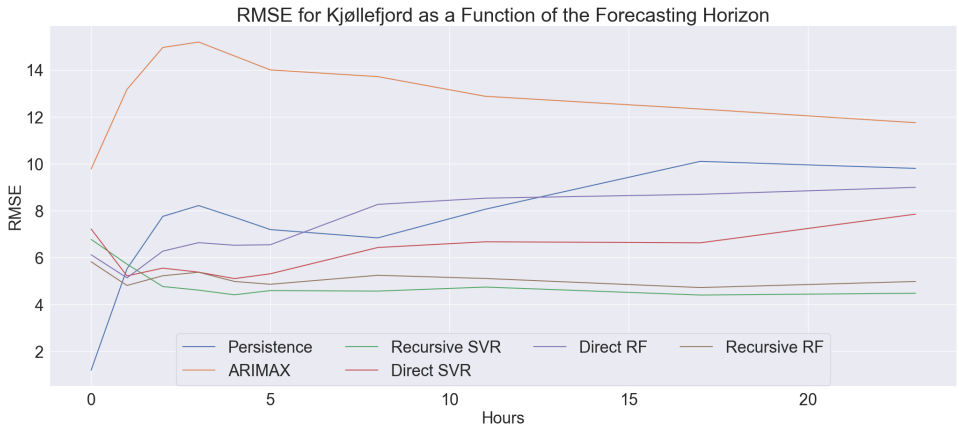


Figure 9.5: RMSE as a function of the forecasting horizon for Kjøllefjord

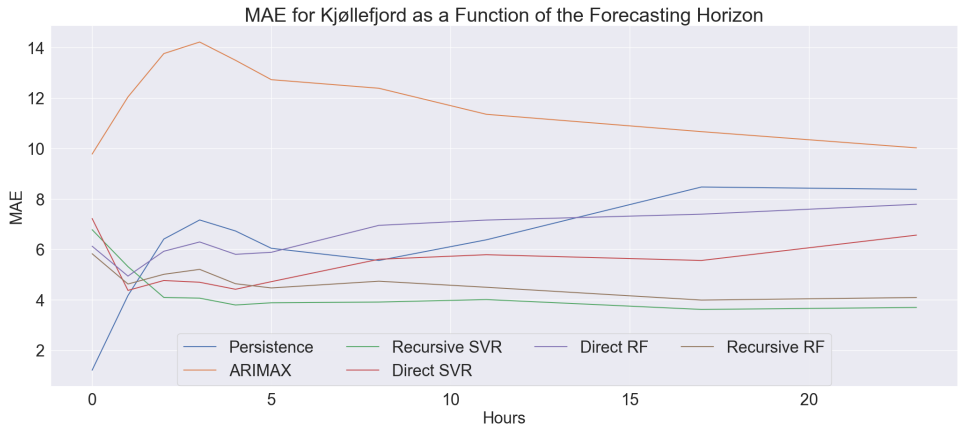


Figure 9.6: MAE as a function of the forecasting horizon for Kjøllefjord

9.3 Havøygavlen Wind Farm

At Havøygavlen wind farm the LSTM model was also tested for three of the time horizons. In this thesis, it seems as if the LSTM model is not competitive with the simple machine learning algorithms, namely the RF and the SVR models. The reason for this is most likely that the hyperparameters that were found for the LSTM model were not optimal. It was very time consuming to tune the hyperparameters of the LSTM model and several assumptions had to be made in order to limit the computational power necessary for reaching the submission deadline of the thesis.

The overall best model for Havøygavlen wind farm is the recursive SVR model, which outperforms the persistence model for all forecasting horizons as shown in figures 9.8 and 9.9. At $h > 1$ the recursive RF model also shows a good performance. The worst performing model is the ARIMAX model, closely followed by the LSTM model, which was only tested for three forecasting horizons. In figure 9.7 the 24 hours ahead predictions from the different forecasting models are shown. Similarly to the rest of the locations it is seen in figure 9.7 that the recursive implementation of the SVR and the RF best follows the pattern of the power output. The RF model struggles to predict the maximum and minimum values, and the direct implementation of the RF and SVR models seems to be estimating the overall trend of the power output.

Results From Havøygavlen Wind Farm											
Method	Error Metric	Forecasting Horizon									
		h=1	h=2	h=3	h=4	h=5	h=6	h=9	h=12	h=18	h=24
Persistence	MAE	4.78	4.45	3.56	3.27	2.84	2.64	3.36	4.64	6.46	7.46
	RMSE	4.58	4.07	3.41	4.26	4.85	5.20	6.09	8.42	11.42	13.38
	NRMSE	0.46	0.44	0.39	0.38	0.36	0.35	0.44	0.59	0.77	0.86
RF(Recursive)	MAE	5.13	3.92	4.03	3.66	3.36	3.56	3.55	3.49	3.32	3.47
	RMSE	5.13	4.16	4.23	4.06	3.83	4.08	4.04	4.02	4.15	4.40
	NRMSE	0.50	0.40	0.41	0.39	0.37	0.40	0.39	0.39	0.40	0.43
RF(Direct)	MAE	3.95	3.97	4.33	4.81	4.89	4.59	4.62	4.97	4.47	6.06
	RMSE	3.95	4.14	4.59	5.18	5.30	5.07	5.45	5.99	6.73	7.18
	NRMSE	0.38	0.40	0.45	0.50	0.51	0.49	0.53	0.58	0.65	0.70
SVR(recursive)	MAE	2.08	2.51	2.30	2.78	2.71	2.96	3.23	3.59	3.48	3.44
	RMSE	2.08	2.65	2.54	3.34	3.22	3.65	3.77	4.14	4.20	4.44
	NRMSE	0.20	0.26	0.25	0.32	0.31	0.35	0.37	0.40	0.41	0.43
SVR(Direct)	MAE	1.94	1.45	2.35	2.79	3.35	3.93	4.57	5.06	5.54	6.18
	RMSE	1.94	1.67	2.78	3.23	3.86	4.54	5.38	6.19	6.92	6.18
	NRMSE	0.19	0.16	0.27	0.31	0.37	0.44	0.52	0.60	0.67	0.73
ARIMAX	MAE	11.43	10.89	10.62	10.35	10.08	9.80	9.41	8.96	8.82	8.44
	RMSE	11.43	11.07	10.83	10.55	10.40	10.31	10.15	10.04	10.24	9.87
	NRMSE	1.11	1.07	1.05	1.02	1.01	1.00	0.98	0.97	1.00	0.96
LSTM	MAE	5.15							8.56		8.95
	RMSE	7.56							10.61		11.14
	NRMSE	0.73							1.03		1.08

Table 9.3: Results from Havøygavlen wind farm. The results show that the persistence model was outperformed by the recursive SVR model for all forecasting horizons closely followed by the recursive RF model that is better than the persistence for $h > 2$.

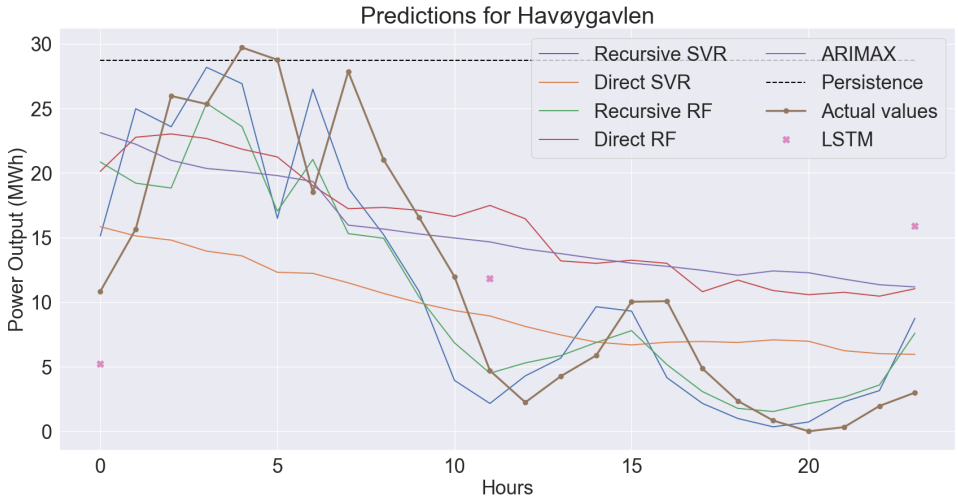


Figure 9.7: 24 hours predictions for Havøygavlen wind farm from all the different forecasting models on one of the test datasets.

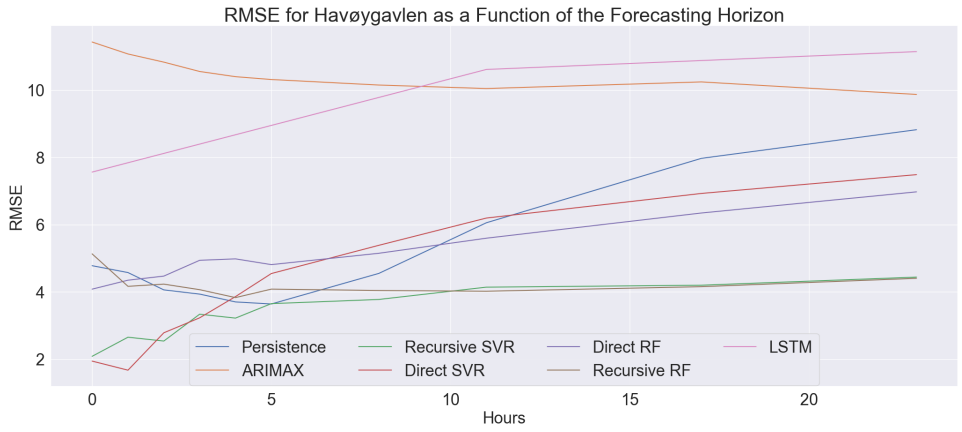


Figure 9.8: RMSE as a function of the forecasting horizon for Havøygavlen

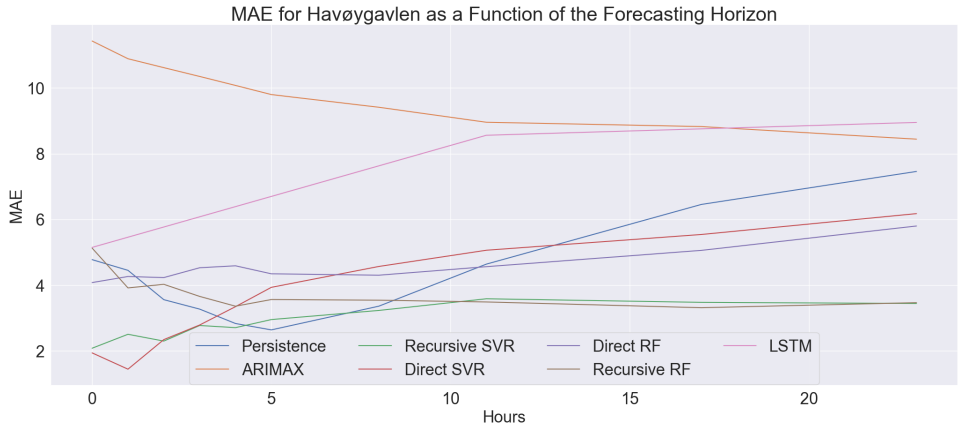


Figure 9.9: MAE as a function of the forecasting horizon for Havøygavlen

9.4 Fakken Wind Farm

At Fakken wind farm the best performing model is again the recursive SVR model. For short forecasting horizons the direct SVR also outperforms the persistence model, but the recursive SVR is better for large forecasting horizons. The recursive RF model is outperformed by the persistence model for $h < 3$, but it is better than the persistence model for large forecasting horizons. The direct RF model is better than the persistence model for large forecasting horizons, but it is worse than the recursive implementations of both the RF and SVR model. The ARIMAX model shows poor results across all timesteps. Figures 9.11 and 9.12 gives an overview of the results. The average results from all splits of the dataset for all the forecasting horizons and all the models are presented in

table 9.4.

Results From Fakken Wind Farm											
Method	Error Metric	Forecasting Horizon									
		h=1	h=2	h=3	h=4	h=5	h=6	h=9	h=12	h=18	h=24
Persistence	MAE	4.35	6.05	7.02	7.08	6.97	7.61	10.56	12.51	14.00	16.36
	RMSE	4.58	5.82	6.31	6.42	6.44	7.07	8.35	10.42	14.21	17.26
	NRMSE	0.29	0.42	0.51	0.51	0.51	0.56	0.81	0.94	1.09	1.24
RF(Recursive)	MAE	8.35	7.67	7.12	7.55	7.41	8.18	7.15	6.55	6.70	6.79
	RMSE	8.35	7.97	7.50	7.91	7.78	8.76	7.81	7.31	8.02	8.37
	NRMSE	0.55	0.52	0.49	0.52	0.51	0.57	0.51	0.48	0.53	0.55
RF(direct)	MAE	5.84	5.98	5.96	7.28	8.08	8.73	9.86	11.22	12.45	14.90
	RMSE	5.84	6.44	6.32	7.97	8.79	9.44	10.82	12.32	14.49	17.48
	NRMSE	0.38	0.42	0.41	0.52	0.58	0.62	0.71	0.81	0.95	1.15
SVR(recursive)	MAE	3.94	3.33	3.16	3.30	3.17	4.22	3.84	3.76	4.80	5.23
	RMSE	3.94	3.92	3.83	3.88	3.75	5.31	4.85	4.97	6.39	6.97
	NRMSE	0.26	0.26	0.25	0.26	0.25	0.35	0.32	0.33	0.42	0.46
SVR(Direct).	MAE	3.45	2.65	2.85	4.42	5.38	6.19	6.44	7.34	9.92	11.32
	RMSE	3.45	3.10	3.32	5.48	6.45	7.22	7.62	8.42	12.20	13.73
	NRMSE	0.23	0.20	0.22	0.22	0.42	0.47	0.50	0.55	0.80	0.90
ARIMAX	MAE	14.69	15.98	15.93	15.86	15.92	15.79	16.24	16.67	16.84	16.75
	RMSE	14.69	16.21	16.36	16.50	16.54	16.44	17.10	17.64	18.89	19.22
	NRMSE	0.96	1.06	1.07	1.08	1.09	1.08	1.12	1.16	1.24	1.26

Table 9.4: Results from Fakken. The results show that the best performing model is the recursive SVR model. The direct SVR model also performs well for short forecasting horizons.

Figure 9.10 shows the 24 hours ahead predictions for Fakken wind farm. It is seen that the recursive implementations of the RF and the SVR model capture large variations in the timeseries quite well, while the ARIMAX and the direct SVR models are almost unaffected by the peak in power output at 15 hours. The direct RF model shows some adjustments to this event, but the overall performance of the model for large forecasting horizons is not comparable to the recursive implementation.

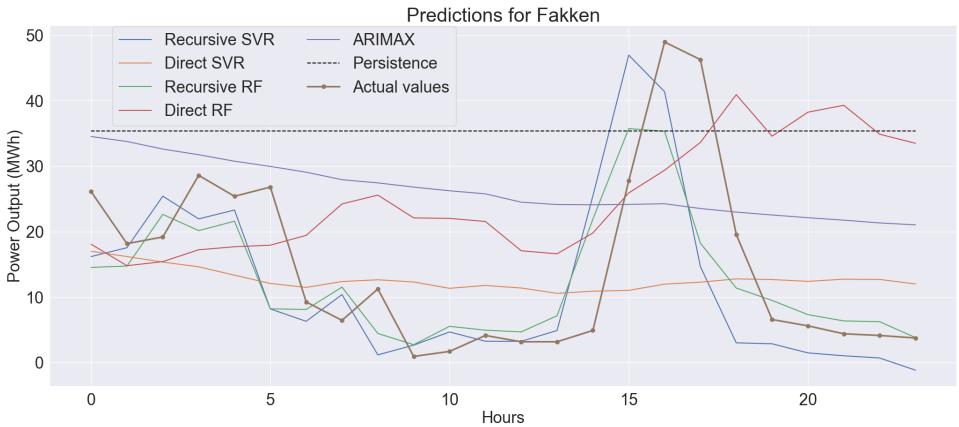


Figure 9.10: 24 hours predictions for Fakken wind farm for all forecasting models.

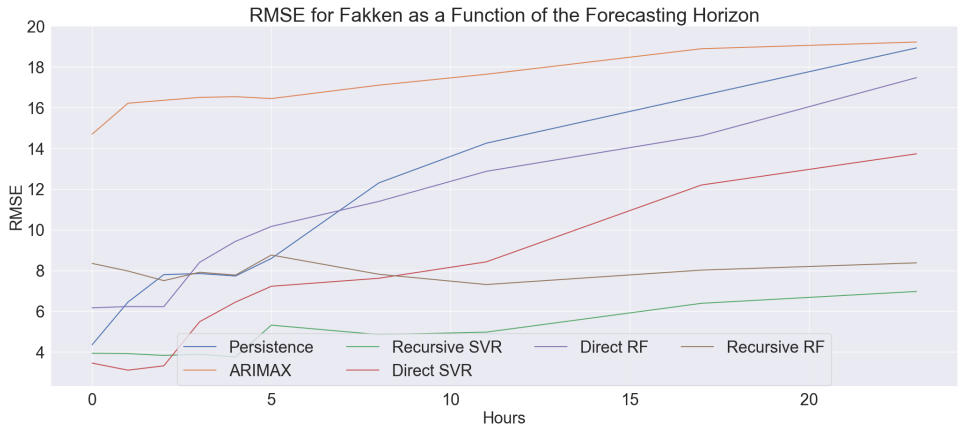


Figure 9.11: RMSE as a function of the forecasting horizon for Fakken

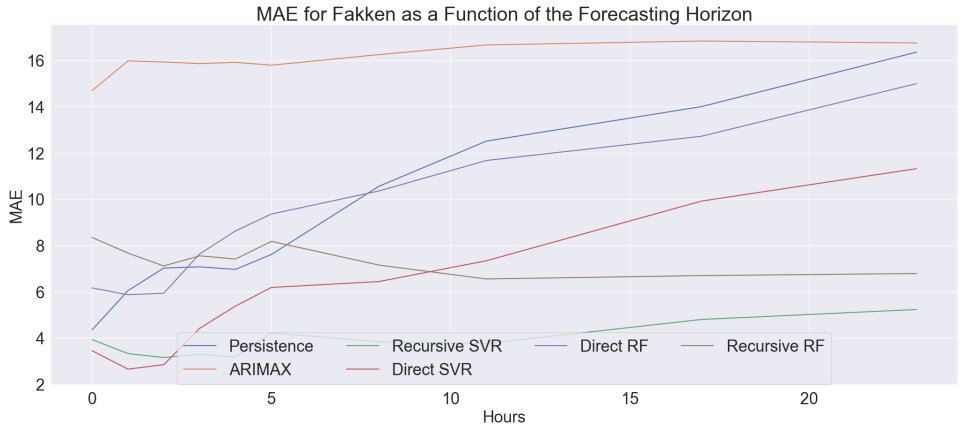


Figure 9.12: MAE as a function of the forecasting horizon for Fakken

9.5 Nygårdsfjellet Wind Farm

At Nygårdsfjellet wind farm the results show that both the recursive RF and the recursive SVR model outperform the persistence model model across all forecasting horizons in terms of all error measures. The direct SVR model also does a good job on this dataset for all forecasting horizons, but it is seen that in terms of the RMSE the persistence model is better for $h > 6$. The persistence model also outperforms the ARIMAX model for all timesteps. The observations are illustrated in figures 9.14 and 9.15. The average results from all splits of the dataset for all forecasting horizons and all models are presented in table 9.5.

The predictions from each of the models for one of the test dataset for Nygårdsfjellet wind farm are shown in figure 9.13. It can be seen that the recursive forecasting strategy is, in addition to detecting the pattern of the power output, also better at prediction the zero level in the time series. In the wind power dataset the zero values are often consecutively repeated values, because the wind turbines are shut down for a period of time, most likely due to high winds or maintenance work. Since the high winds is a determining weather factor for a turbine shut down, but the maintenance events are unaccounted for in the dataset it may be harder for some models to interpret the zero level. For example the recursive models will know from the recursive predictions that the last value is low, but the direct methods may be unaffected by the events that led to zero wind power output. This will be further discussed in chapter 10.

Results From Nygårdsfjellet Wind Farm											
Method	Error Metric	Forecasting Horizon									
		h=1	h=2	h=3	h=4	h=5	h=6	h=9	h=12	h=18	h=24
Persistence	MAE	2.35	3.51	4.44	5.01	5.63	5.67	6.81	7.01	7.02	7.69
	RMSE	1.10	1.50	2.78	4.23	5.16	5.26	5.14	4.62	4.00	4.94
	NRMSE	0.21	0.33	0.43	0.48	0.54	0.55	0.65	0.67	0.68	0.81
RF(Recursive)	MAE	2.18	3.16	3.38	3.05	2.96	3.10	2.83	2.58	2.59	2.62
	RMSE	2.18	3.37	3.60	3.31	3.22	3.51	3.23	2.99	3.07	3.31
	NRMSE	0.20	0.30	0.32	0.30	0.29	0.32	0.29	0.27	0.28	0.30
RF(Direct)	MAE	1.38	1.89	1.71	1.95	2.61	3.11	3.97	6.15	9.30	10.46
	RMSE	1.38	2.00	1.88	2.27	3.34	3.91	4.98	7.85	11.16	11.99
	NRMSE	0.12	0.18	0.17	0.21	0.30	0.35	0.45	0.70	1.00	1.08
SVR(recursive)	MAE	1.18	1.66	2.11	2.56	2.30	2.53	2.40	2.03	1.74	1.91
	RMSE	1.18	1.75	2.52	3.04	2.82	3.13	2.97	2.66	2.41	2.80
	NRMSE	0.11	0.16	0.23	0.27	0.25	0.28	0.27	0.24	0.22	0.25
SVR(Direct)	MAE	0.89	0.84	1.11	1.62	2.78	3.67	4.27.	4.62	5.89	6.82
	RMSE	0.89	0.99	1.30	2.14	3.87	4.84	5.63	6.07	7.39	8.46
	NRMSE	0.08	0.09	0.12	0.19	0.35	0.43	0.51	0.55	0.66	0.76
ARIMAX	MAE	6.81	7.25	7.39	6.94	6.46	6.14	7.27	8.08	8.88	9.14
	RMSE	6.81	7.33	7.55	7.22	7.24	7.22	8.28	9.07	9.89	10.08
	NRMSE	0.61	0.66	0.68	0.65	0.65	0.65	0.74	0.81	0.89	0.911

Table 9.5: Results from Nygårdsfjellet. The results show that the recursive RF and SVR outperforms the persistence model for all forecasting horizons in terms of all error measures. The worst performing model is the ARIMAX model.

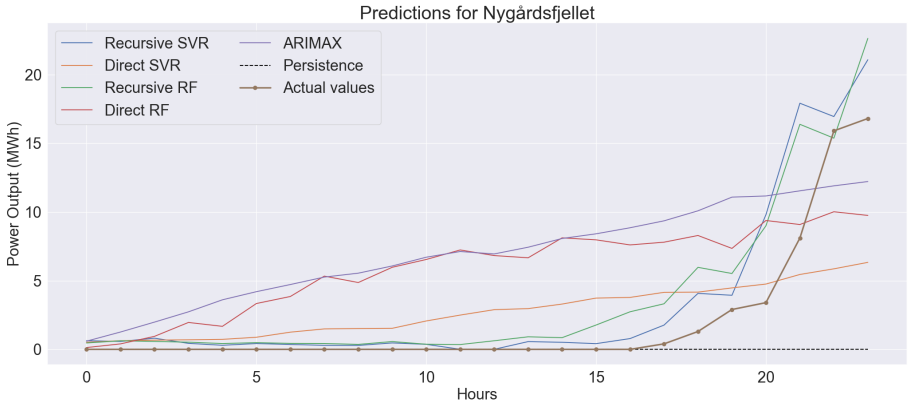


Figure 9.13: 24 hours predictions for Nygårdsfjellet wind farm from all the different forecasting models.

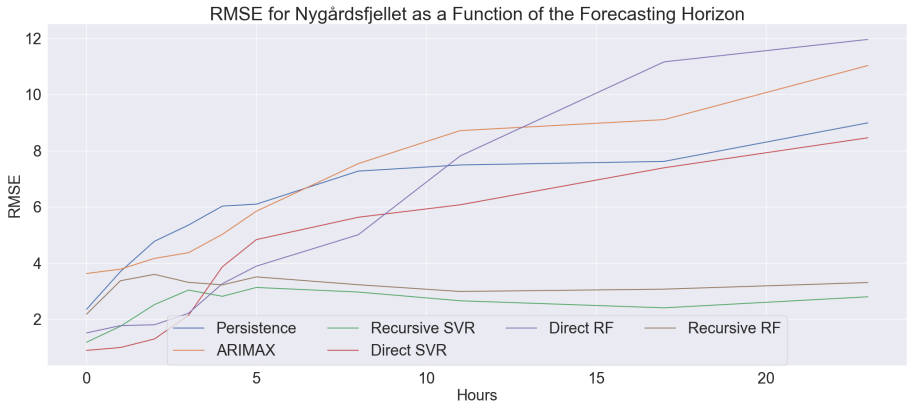


Figure 9.14: RMSE as a function of the forecasting horizon for Nygårdsfjellet

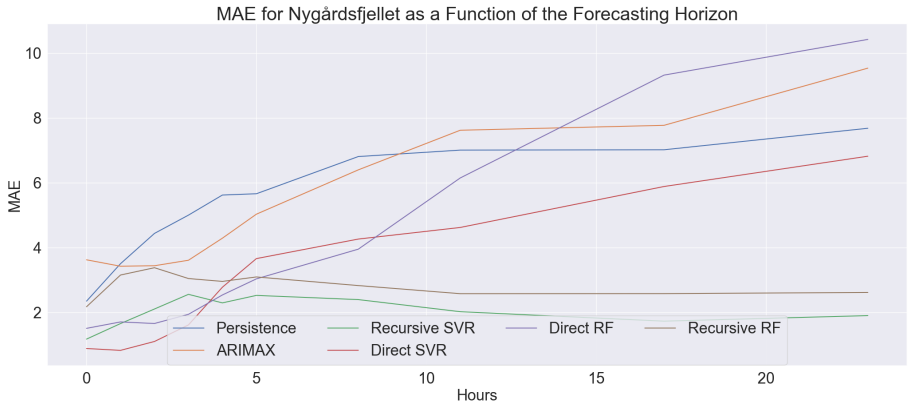


Figure 9.15: MAE as a function of the forecasting horizon for Nygårdsfjellet

9.6 Overall Results

When considering the overall performance of the models across all the five different wind farm locations the NRMSE is used for comparison. The comparison of the modes are done according to the overall NRMSE across all forecasting horizons, and the performance for different forecasting horizons. In figure 9.16 the average NRMSE across all the different wind farm locations as a function of time is shown.

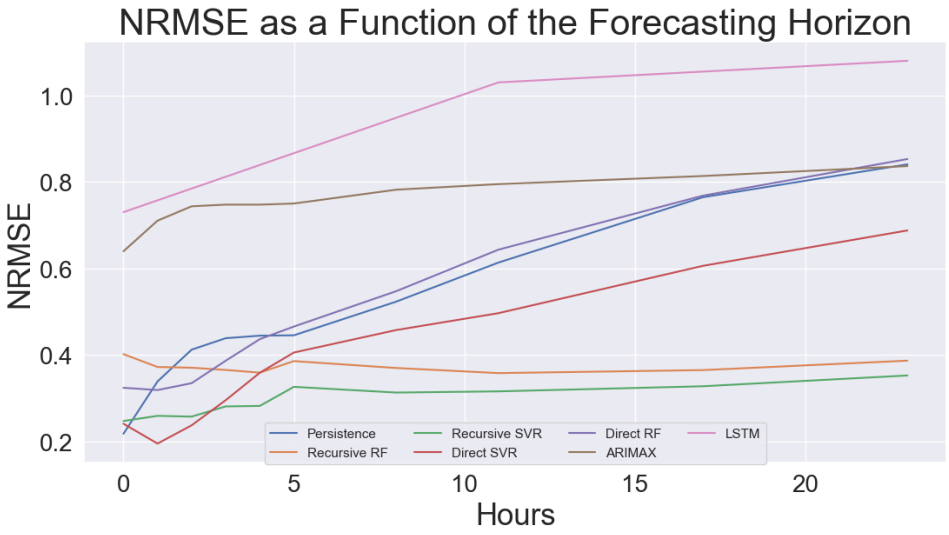


Figure 9.16: Average NRMSE as a function of forecasting horizons

As seen in sections 9.1-9.5 in figures 9.1-9.13 it is evident that the recursive implementations of the models does a better job in detecting the pattern of the power output from all wind farms while the direct approaches usually has a good performance on short timescales, but their accuracy decreases as the forecasting horizon increases, as expected. This is confirmed by looking at figure 9.16. It is seen that the recursive forecasting approaches, quite unexpectedly, have a relatively constant error as the forecasting horizon increases, whereas the error of the direct approaches increases as the forecasting horizon increases. The ARIMAX model has a more constant error for long forecasting horizons, but it is seen that the error is slightly lower for shorter timescales than for longer timescales. For one hour predictions the persistence model is the best performing model, but the error of the persistence model rapidly increases along with the forecasting horizon. The NRMSE of the LSTM model is also included in the plot, but the error is only recorded for Havøygavlen, so the overall error is not comparable with the other averaged errors, particularly since Havøygavlen is thought to be the most difficult location to forecast based on the results of the persistence model. However, it is seen that also for the

LSTM model the error increases with the forecasting horizon, but it seems as if he growth rate of the error is lower than for the persistence model.

In figure 9.17 it is seen that it is the recursive SVR model that has the overall best performance across all datasets, closely followed by the recursive RF model. The worst performing model is the ARIMAX model, and the second worst is the persistence model.

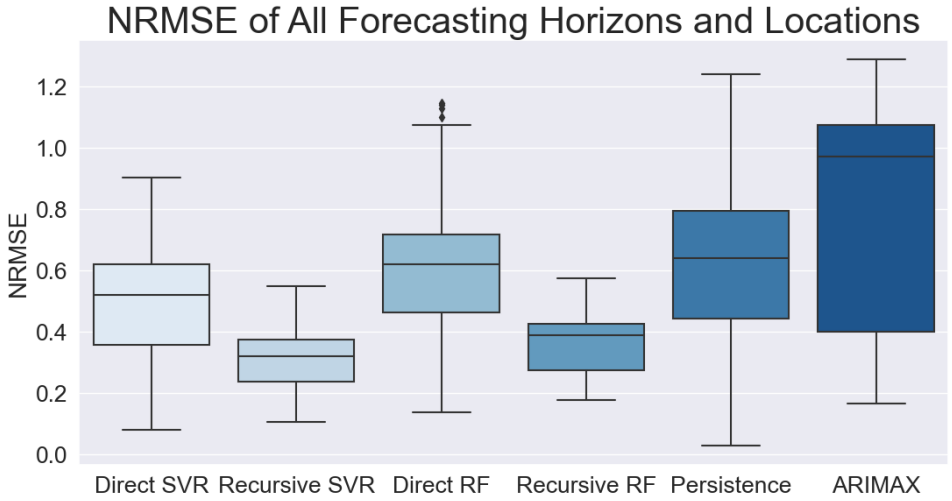
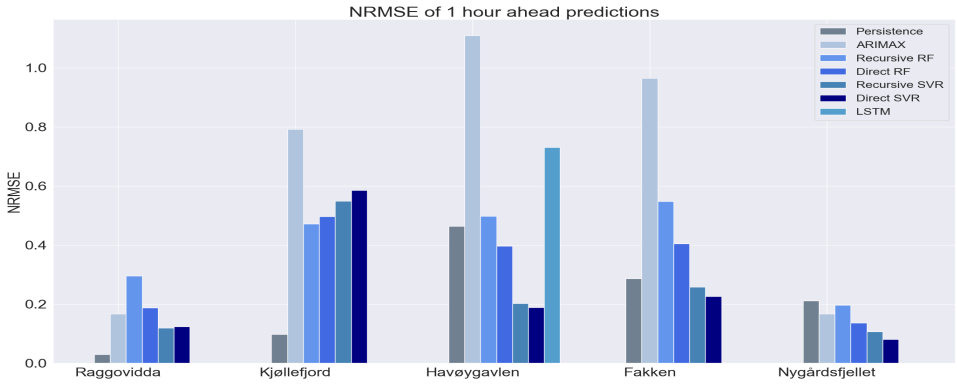
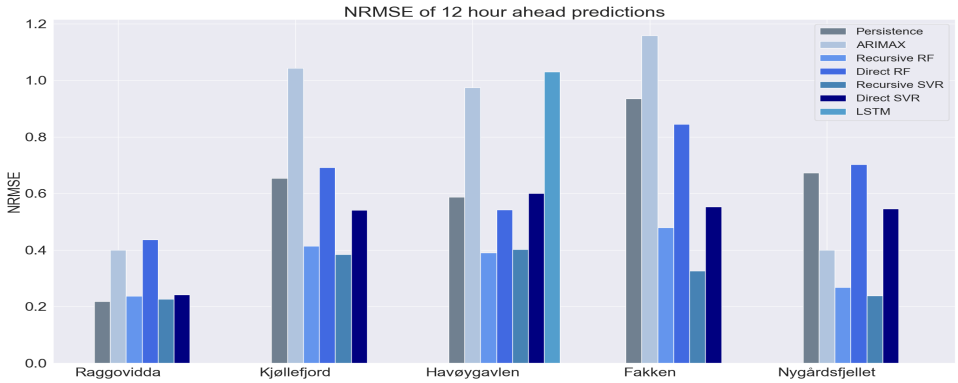


Figure 9.17: NRMSE of all models across all forecasting horizons and locations shown as a box plot. The plot shows that the ARIMAX model is the worst performing model across all datasets with the largest spread of the NRMSE and the highest median NRMSE. The persistence model is the second worst. It has a much lower median NRMSE than the ARIMAX, but with larger spread than the rest of the models. The best performing model is the recursive SVR model that has the lowest spread and median NRMSE.

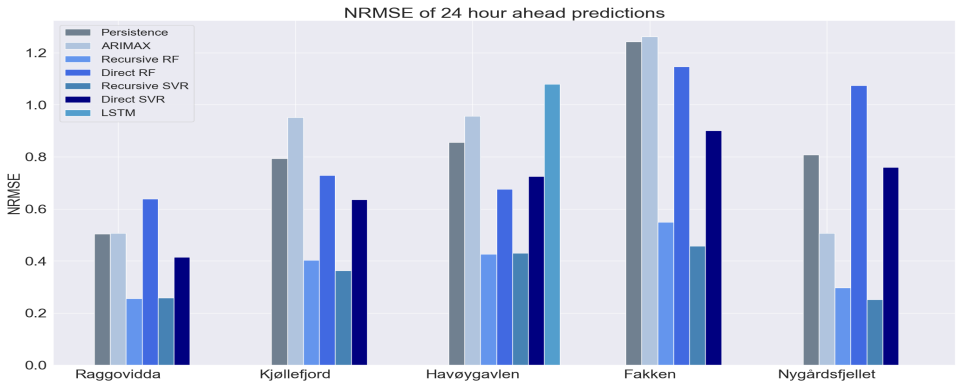
Another observation that is made through section 9.1 - 9.5 is the there is a big difference across locations in which models that performs well on different datasets. This is illustrated in figure 9.18.



(a) The NRMSE of different locations for one-hour-ahead predictions



(b) The NRMSE of different locations for 12 hours ahead predictions



(c) The NRMSE of different locations for 24 hours ahead predictions

Figure 9.18: The NRMSE for all wind farm locations and different forecasting horizons.

It is seen in figure 9.18 that the ARIMAX model, for example, shows a good performance for one-hour-ahead predictions at Raggovidda and Nygårdsfjellet, but at Havøygavlen it is by far the worst performing model. For one-hour-ahead

predictions at Nygårdsfjellet wind farm the direct SVR is the best performing model, while for Kjøllefjord wind farm the direct SVR model is the second worst performing model.

To summarize, it is noted in this chapter that the best model for wind power predictions, as implemented in this thesis, is the recursive SVR model. The worst model was the ARIMAX model. The persistence model often outperformed the more complicated models for short forecasting horizons, but when the forecasting horizon increased the error rapidly increased. The recursive RF model also showed promising results for large forecasting horizons, and the direct SVR and RF models showed good results for short timescales, but were outperformed by the recursive implementations for longer forecasting horizons. It was also seen that the models' performance is highly dependent on the forecasting horizons they are evaluated by and the locations the predictions are made for.

10 Discussion

In chapter 9 it was found that the best performing model implemented in this thesis is the recursive SVR model closely followed by the recursive RF model. The results varied greatly across all of the locations. As seen in literature, wind power forecasting is a difficult task, and in many cases complicated forecasting methods do not show any improvement compared to the performance of the persistence model. There are several factors that could have influenced the results obtained in this thesis, and in this chapter a discussion of the results and the limitations of the projects will be provided.

10.1 Project Limitations

The initial aim of this thesis was to implement two neural network models, namely the LSTM and the temporal convolutional network (TCN), for prediction of future power output at five different wind farms in northern Norway, and compare these to simpler machine learning models, such as the RF and SVR and the statistical ARIMAX model. Both the LSTM and the TCN models were implemented, but the computational resources they needed for hyperparameter tuning were too large for the time scope of this thesis. Issues with computational capacity, along with time constraints made it challenging to include properly optimized models for each of the wind farms and each of the forecasting horizons that was determined for the project. Unfortunately, the neural network models had to be greatly reduced, and the final predictions from one neural

network model only included one location and three different forecasting horizons. The amount of time that were devoted to the neural network models also had a negative effect on the results that are presented in the final results of this thesis. In chapter 9 it is seen that the recursive implementations of the RF and the SVR models outperform the direct approaches and the ARIMAX model as well as the LSTM model. However, these results are somewhat suspicious, as the recursive approach is generally expected to produce a higher error than the direct approach because of the accumulation of errors, and this should be particularly evident for long forecasting horizons. This point will be further discussed in section 10.2. In figure 9.16 it is observed that the recursive methods produce an error that is quite stable over the range of forecasting horizons, although an increase would be expected both due to the mentioned accumulation of error in the recursive approach and the general increase in uncertainty with larger forecasting horizon. From the prediction plots shown for each of the wind farms in chapter 9 in sections 9.1 - 9.5 it is also observed that the predictions looks like a shifted series of the power output itself, where the models predict the outcome before the actual event occurs. This may indicate that the models have access to information that should be kept from the model in the forecasting process, and that there is an error in the implementation of the recursive forecasting approach. Another observation that was made in the presentation of the results was that the direct forecasting approach seems to find an overall trend of the wind power output, and use this as the prediction results of the model. It is hard to determine whether this is the models best approach for a low error among the predictions, or if this is a result of models that are not properly optimized. It is noted, however, that the direct approach still outperforms the persistence model for large forecasting horizons, but are worse than the persistence model for short time horizons. This is consistent with what is seen in the literature for wind power forecasting, where complicated forecasting models often are outperformed by the persistence model for short forecasting horizons.

10.2 Recursive vs. Direct Forecasting Method

In the presentation of results for both the RF and the SVR model it was noted that the recursive forecasting method almost always outperforms the direct forecasting method method, particularly for long forecasting horizons. According to (Taieb, Hyndman, et al., 2012), a recursive forecasting method is biased when the underlying model is nonlinear, but the direct forecasting method has higher variance because it uses fewer observations when estimating the model, especially for longer forecasting horizons. In many cases the recursive forecasting strategy is expected to have a worse performance than the direct strategy because of the accumulation of errors that builds up as the forecasting

horizon increases when using the recursive strategy. However, because the direct forecasting method generates a new model for each horizon, it is possible that consecutive forecasts are based on different conditioning information and model statistics, which may lead to inconsistency in the forecasting model resulting in a problem when generating forecasts from potentially very different models at various forecasting horizons (Taieb et al., 2012). When using this approach on volatile data such as the wind power dataset, this may have an extenuated effect. This problem is further worsened when each of the generated models are nonlinear and nonparametrically estimated (Chen, Yang, & Hafner, 2004). Contrarily, the recursive forecasting strategy ensures that the fitted model matches the data generating process as closely as possible, and compared to the direct forecasting strategy the recursive strategy requires a significantly lower amount of computational power especially when there is a large amount of data involved. Considering the performance of the recursive RF and SVR models on this dataset, an idea would be to adopt the DirRec strategy for multi-step forecasting that was described in chapter 4 when using these models for forecasting of wind power.

10.3 Tuning of Hyperparameters in LSTM Model

Considering the many different models, and the many different time horizons that this thesis set out to predict, it is possible that not all models were optimized to their full potential. For instance, the LSTM model could have been optimized for each of the forecasting horizons, which might have generated better results for the model. Additionally, a better basis for comparison of the LSTM model with the rest of the models would have been established if results for every forecasting horizon were found. In this thesis the optimal hyperparameters of the LSTM model were found by running a relatively small grid search for each of the dataset splits for Havøygavlen wind farm, and forecasting was done directly for 1 hour, 12 hours and 24 hours ahead. Simple steps such as grid searching for a greater range of hyperparameters could have led to improvements in the LSTM model. On the other hand, the improvements in performance achieved by the exhaustive process of hyperparameter tuning may not in all real world applications be preferred over a simpler model with slightly worse results. For research purposes it would, of course, be interesting to see the results of an optimal LSTM model on this dataset.

An advantage with neural network models for time series forecasting is that it can directly forecast the entire forecasting horizon by using the multiple input multiple output (MIMO) strategy that was described in chapter 4. Hence, experimenting with the LSTM model and different forecasting strategies could be worth exploring.

In chapter 2 it was learned that (Xiaoyun et al., 2016) has obtained successful results when using the LSTM with the principal components of the exogenous variables from a NWP model. Considering the dependencies that was observed between the exogenous variables and the dependent variable in figure 6.11 in section 6, it might be that the LSTM model has received too much input data and that the model would have better interpreted the dependencies between the weather data and the power output data if only the most important information was given to the model by means of some dimensionality reduction techniques. This is also the case for the rest of the models, especially the SVR model, as the RF model already performs some feature selection when generating its ensemble of decision trees.

10.4 ARIMAX model

It was observed in chapter 9 that the ARIMAX model is the worst performing model implemented in this thesis. In the ARIMAX model the data has to be rendered stationary by the d parameter in the model. It is shown in chapter 8 that after differencing the power output data once, all the variables in the dataset are stationary except the temperature data, which still has some traces of a daily pattern present. To avoid exaggerated differencing of the dependent variable in the dataset it was decided that the data would only be differenced once in the model. Since the seasonality in the temperature data is still present, this might have lead to spurious regression of the dataset (Hyndman, 2010). It is noted in chapter 6 that the correlation between the power output data and the temperature data is weak, and a solution to the problem could be to predict the future power output from the dataset by dropping the temperature variable. This way all the variables in the dataset would be rendered stationary after one differencing, and the model might be able to achieve better results. A quick experiment is run to check the models performance when the temperature data is dropped from the dataset, and the results are shown in table 10.1 and figures 10.1-10.3, which show the error as a function of the forecasting horizon. It is seen in the figures that the difference between the ARIMAX model with temperature data, and the ARIMAX model without temperature data is small for short forecasting horizons, but when the forecasting horizon increases, the model that does not include temperature data has a better performance in terms of all error measures. However, looking at the errors of the persistence model in the figures, it is seen that the ARIMAX model is still not competitive with the simple persistence model on this dataset.

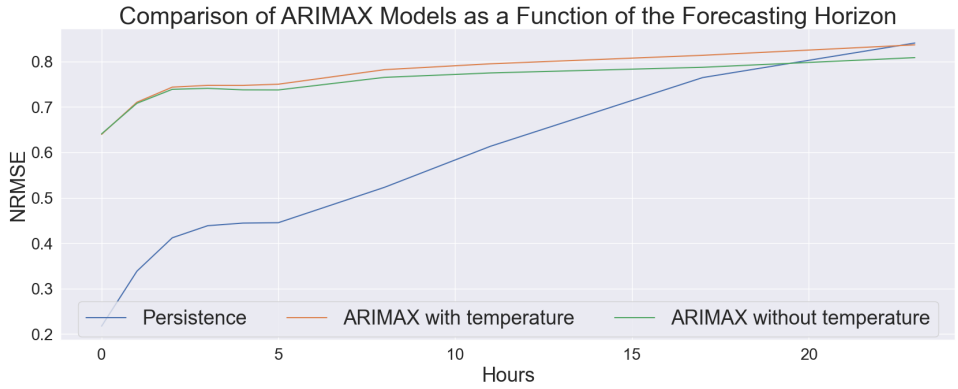


Figure 10.1: Comparison of the ARIMAX model in terms of the average NRMSE across all locations with and without temperature data

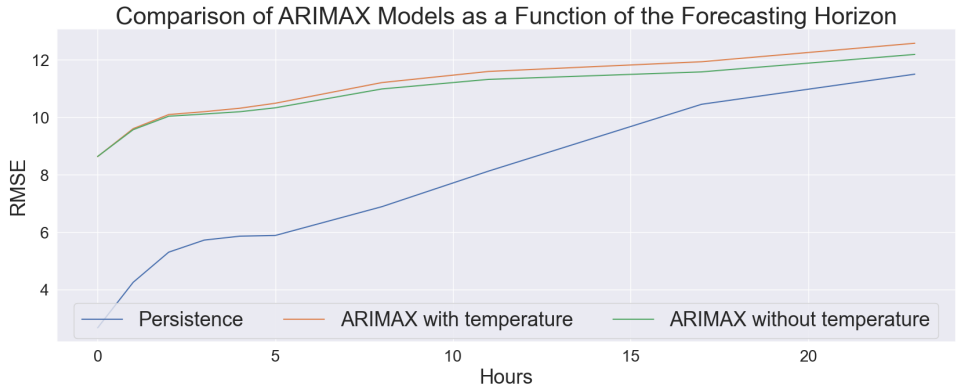


Figure 10.2: Comparison of the ARIMAX model in terms of the average RMSE across all locations with and without temperature data

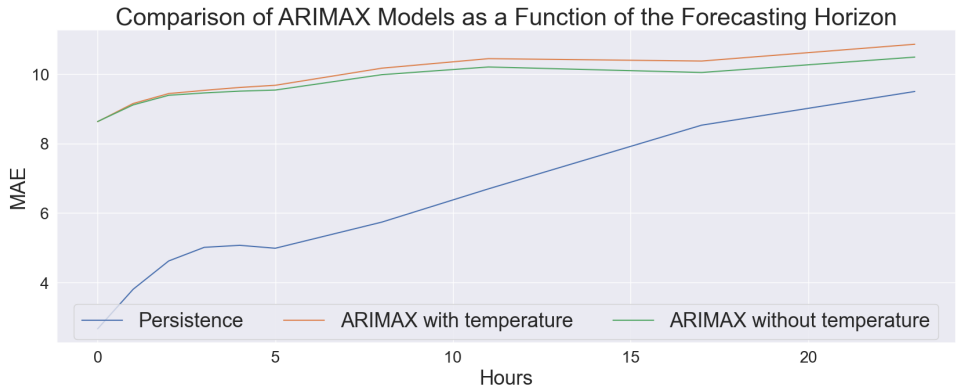


Figure 10.3: Comparison of the ARIMAX model in terms of the average MAE across all locations with and without temperature data

ARIMAX model without temperature data											
Location	Error Metric	h=1	h=2	h=3	h=4	h=5	h=6	h=9	h=12	h=18	h=24
Raggovidda	MAE	3.55	3.36	3.42	3.58	4.25	4.97	6.31	7.48	7.60	9.31
	RMSE	3.55	3.72	4.14	4.34	4.98	5.77	7.42	8.54	8.93	10.79
	NRMSE	0.16	0.17	0.19	0.20	0.23	0.26	0.34	0.39	0.41	0.49
Kjøllefjord	MAE	9.79	11.99	13.67	14.10	13.36	12.57	12.14	10.98	10.10	9.42
	RMSE	9.79	13.15	14.86	15.05	14.44	13.83	13.43	12.47	11.70	11.09
	NRMSE	0.79	1.07	1.20	1.22	1.17	1.12	1.09	1.01	0.95	0.90
Havøygavlen	MAE	11.47	10.85	10.49	10.19	9.84	9.49	9.07	8.64	8.42	8.01
	RMSE	11.47	10.99	10.69	10.38	10.13	9.96	9.76	9.70	9.83	9.48
	NRMSE	1.12	1.07	1.04	1.01	0.98	0.97	0.95	0.94	0.95	0.92
Fakken	MAE	14.81	16.03	15.95	15.85	15.86	15.71	16.10	16.44	16.52	16.41
	RMSE	14.81	16.26	16.36	16.45	16.44	16.33	16.91	17.35	18.51	18.81
	NRMSE	0.97	1.07	1.07	1.08	1.08	1.07	1.11	1.14	1.21	1.23
Nygårdsfjellet	MAE	6.71	7.15	7.29	6.82	6.33	6.04	7.11	7.88	8.64	8.88
	RMSE	6.71	7.23	7.45	7.12	7.13	7.12	8.12	8.85	9.63	9.81
	NRMSE	0.60	0.65	0.67	0.64	0.64	0.64	0.73	0.79	0.87	0.88

Table 10.1: Results of ARIMAX model without temperature data

10.5 Input data

In chapter 6 the number of lags included in the input data fed to the models was determined by looking at the ACF and PACF plots in figure 6.8 and 6.9. It was seen in figure 6.8 that there is a strong correlation between lagged values in the historical power output data, and in figure 6.9 it was seen that without the relationships between intervening observations, there is no significant correlation for lag values beyond two lags in the historical power output data. The models were first tested and validated for the number of significant lags shown for each of the locations in figure 6.8. This means that the minimum number of lags given to the models were 50 lags for Havøygavlen and the maximum number of lags given to the models were 80 lags for Kjøllefjord. Considering the four different exogenous variables that are used in this thesis, this means that for Kjøllefjord the number of features given to the models were 399, and for Havøygavlen the number of features given to the models were 249. As a consequence the models performance was very bad, possibly because the models received redundant information and were curtailed by the curse of dimensionality. The curse of dimensionality states that as the number of dimensions in the data increases the input space may become so large that the available data samples become sparse in the input space, which may yield sub-optimal results from a machine learning model (Theodoridis & Koutroumbas, 2009). The number of lags were then reduced to two lags according to the PACF plot in figure 6.9. This showed a significant improvement in the models performance, and therefore this number of lags was chosen for the experiments in this thesis. However, too few lags in the model might lead to unexploited information given to the models, and by using more input lags the models might better be able to connect the information in the historical

values to the current observations. Ideally, the number of lags used in the models should be validated as a hyperparameter on the same basis as the rest of the hyperparameters given to each of the models, but because of previously mentioned limitations this was not feasible for all models.

10.6 Non-deterministic Models

The limited amount of available data may also have affected the results obtained in the experiments because of the non-deterministic nature of the models that were tested. A non-deterministic model is influenced by randomness in initialization and adjustments. Hence, the models are never identical, even when presented with the exact same data. In this thesis the dataset was divided into four splits, representing a k-fold cross validation technique for time series forecasting with k=4, and the performance of a model was presented as the average error across all splits for one location. However, because of the many models that were tested and limitations in computational capacity, the models were only tested once for each split. A better approach might have been to run the k-fold cross-validation several times and report the average performance of more than one round of experiments. This way the variance in the average performance between runs could have been minimized.

11

Conclusion

In this thesis four different machine learning models for multi-step time series forecasting of the power output from five different wind farms in northern Norway have been implemented and compared. Forecasting the power output from wind farms is an important task for a more effective integration of wind power to the power grid, further development of wind power technology, and increased wind power production. However, the volatile nature of wind makes wind power forecasting a difficult task. The data collected from five different wind farms in Northern Norway and the forecast weather data from the Norwegian Meteorological Institute was processed, explored, and structured for the purpose of multi-step and multivariate prediction of the future wind power output from all wind farms. The implemented models were evaluated by comparing their performance, first for each of the wind farm locations and the forecasting horizons, and later across all wind farm locations. It was found that the persistence model performs well on short forecasting horizons, compared to the more complex machine learning algorithms. However, when the prediction horizon increased, the more advanced models such as the RF and the SVR models showed better results than the simple persistence model and the ARIMAX model. The ARIMAX model did not show improvements with regards to the persistence model, nor any of the other implemented models for any of the forecasting horizons. Obstacles along the way in the project resulted in a reduction of the experiments with the neural network models that were implemented, and the LSTM model was only tested for one of the locations and three different forecasting horizons. For one-hour-ahead predictions the LSTM model showed promising results for Havøygavlen wind farm, but for

large forecasting horizons the neural network model was outperformed by the persistence model. A better basis for comparison would have been established for the LSTM model if it was tested for all of the locations and all the forecasting horizons. Additionally, more computational capacity could have been helpful in better tuning of the hyperparameters of the model, which might have improved the results of the neural network model.

Two of the machine learning models, namely the RF and the SVR models, were also evaluated with respect to their performance when implemented recursively or when the predictions were made directly for the targeted prediction horizon. It was seen from the results that the recursive approach was better at predicting the future power output than the direct approach. However, the predictions from the recursive approach looked like a shifted series of the power output itself, where the events in the power output were predicted before they actually occurred, which was suspicious. According to theory, the recursive approach should have an accumulation of errors for large forecasting horizons that is not seen in the direct forecasting approach. This is not the case for the models implemented in this thesis, which indicates that the recursive approach implemented in this thesis is flawed.

Altogether it was found in this thesis that the persistence model is preferred for very short forecasting horizons as opposed to the proposed models in this project. Excluding the results of the recursive approach, it was seen that for large forecasting horizons the SVR model outperformed the persistence model when using the direct approach for all of the wind farm sites. For a few locations the direct RF model also showed improvements to the persistence model for large forecasting horizons. Hence, it can be concluded that the machine learning models implemented in this thesis are worth experimenting with for the purpose of forecasting the future wind power output at Raggovidda, Kjøllefjord, Havøygavlen, Fakken and Nygårdsfjellet wind farms. There are, however, a few unanswered questions that are left when finishing this project: Are shallow machine learning models preferred over deep learning models for wind farm locations in Northern Norway? Will the recursive, direct or a combination of the two multi-step forecasting techniques perform better on the given dataset? And what is the optimal usage of exogenous variables when predicting the wind power output at these locations?

Although some questions are left unanswered, the work presented in this thesis will, hopefully, contribute to further exploration and development of prediction techniques suitable for this dataset. The author certainly has learned more than thought possible over the course of five months. About time series forecasting, machine learning in general, and how to go about when trying to solve a scientific research problem.

11.1 Future Works

Future work on this dataset that may contribute to better results for forecasting the wind power output of wind farms in Northern Norway includes looking at several of the notions made in chapter 10 and by trying to answer some of the questions stated in the previous section. By focusing the attention of a project to the training and validation of the LSTM neural network, comparable results to the results obtained in this thesis would be achieved. The performance of a deep learning model on this dataset could then be compared to the performance of the simpler machine learning models already implemented in this thesis, and an important basis for the way forward would then be established. Future works should also consider evaluating the optimal number of lags that should be given to a model for this dataset, as this is a parameter that greatly influences the performance of different models. The use of dimensionality reduction techniques should also be considered according to the low correlations that were observed between some of the exogeneous variables and the power output in this project.

Bibliography

Asdrubali, F., Baldinelli, G., D'Alessandro, F., & Scrucca, F. (2015). Life cycle assessment of electricity production from renewable energies: Review and results harmonization. *Renewable and Sustainable Energy Reviews*, *42*, 1113–1122.

Babar, B., Luppino, L. T., Boström, T., & Anfinsen, S. N. (2020). Random forest regression for improved mapping of solar irradiance at high latitudes. *Solar Energy*, *198*, 81–92.

Bianchi, F. M., Maiorino, E., Kampffmeyer, M. C., Rizzi, A., & Jenssen, R. (2017). An overview and comparative analysis of recurrent neural networks for short term load forecasting. *arXiv preprint arXiv:1705.04378*.

Bilal, B., Ndongo, M., Adjallah, K. H., Sava, A., Kébé, C. M., Ndiaye, P. A., & Sambou, V. (2018). Wind turbine power output prediction model design based on artificial neural networks and climatic spatiotemporal data. In *2018 ieee international conference on industrial technology (icit)* (pp. 1085–1092).

Birkelund, Y., Alessandrini, S., Byrkjedal, Ø., & Monache, L. D. (2018). Wind power predictions in complex terrain using analog ensembles.

Bontempi, G., Taieb, S. B., & Le Borgne, Y.-A. (2012). Machine learning strategies for time series forecasting. In *European business intelligence summer school* (pp. 62–77).

Botterud, A., Wang, J., Miranda, V., & Bessa, R. J. (2010). Wind power forecasting in u.s. electricity markets. *The Electricity Journal*, *23*(3), 71-82. Retrieved from `https://www.sciencedirect.com/science/article/pii/S104061901000062X`

BP p.l.c. (2020). Bp statistical review of world energy 2020. *BP Statistical Review, London, UK*.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Brockwell, P. J., & Davis, R. A. (2016). *Introduction to time series and forecasting*. USA: Springer International Publishing Switzerland.

Brownlee, J. (2017). A gentle introduction to backpropagation through time. *Machine Learning Mastery*, *23*(06). Retrieved from `https://machinelearningmastery.com/gentle-introduction-backpropagation-time/` ([Online; accessed 31 May 2021])

Brownlee, J. (2019). *Deep learning for time series forecasting*. Author.

Byrkjedal, Ø., & Åkervik, E. (2009). Vindkart for norge. *NVE Oppdragsrapport A*, *9*, 1–38.

Chatfield, C. (2000). *Time-series forecasting*. CRC press.

Chen, R., Yang, L., & Hafner, C. (2004). Nonparametric multistep-ahead prediction in time series analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *66*(3), 669–686.

Costa, A., Crespo, A., Navarro, J., Lizcano, G., Madsen, H., & Feitosa, E. (2008). A review on the young history of the wind power short-term prediction. *Renewable and Sustainable Energy Reviews*, *12*(6), 1725–1744.

Ding, Y. (2019). *Data science for wind energy*. CRC Press.

Duran, M. J., Cros, D., & Riquelme, J. (2007). Short-term wind power forecast based on arx models. *Journal of Energy Engineering*, *133*(3), 172–180.

Encyclopædia Britannica. (2011). *Wind Turbine.* `https://www.britannica.com/technology/wind-turbine`. ([Online; accessed 07 May 2021])

Foley, A. M., Leahy, P. G., Marvuglia, A., & McKeogh, E. J. (2012). Current methods and advances in forecasting of wind power generation. *Renewable Energy*, *37*(1), 1–8.

Fossem, A. A. (2019). *Short-term wind power prediction models in complex terrain based on statistical time series analysis.*

Friedman, J., Hastie, T., Tibshirani, R., et al. (2001). *The elements of statistical learning* (Vol. 1) (No. 10). Springer series in statistics New York.

Gers, F. A., Schmidhuber, J., & Cummins, F. (1999). Learning to forget: Continual prediction with lstm.

Giebel, G., & Kariniotakis, G. (2017). Wind power forecasting—a review of the state of the art. *Renewable energy forecasting*, 59–109.

Gonzalez, R. C., & Woods, R. E. (2018). *Digital image processing*. Pearson Global Edition.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. (`http://www.deeplearningbook.org`)

Guezuraga, B., Zauner, R., & Pölz, W. (2012). Life cycle assessment of two different 2 mw class wind turbines. *Renewable Energy*, *37*(1), 37–44.

Hanifi, S., Liu, X., Lin, Z., & Lotfian, S. (2020). A critical review of wind power forecasting methods—past, present and future. *Energies*, *13*(15), 3764.

Heinermann, J., & Kramer, O. (2016). Machine learning ensembles for wind power prediction. *Renewable Energy*, *89*, 671–679.

Hobijn, B., Franses, P. H., & Ooms, M. (2004). Generalizations of the kpss-test for stationarity. *Statistica Neerlandica*, *58*(4), 483–502.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hong, Y.-Y., & Rioflorido, C. L. P. P. (2019). A hybrid deep learning-based neural network for 24-h ahead wind power forecasting. *Applied Energy*, *250*, 530–539.

Hyndman, R. J. (2010). *The arimax model muddle.* `https://robjhyndman.com/hyndsight/arimax/`. (Accessed: 2021-06-10)

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting principles and practice*. Melbourne, Australia: OTexts.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International journal of forecasting*, *22*(4), 679–688.

Infield, D., & Freris, L. (2020). *Renewable energy in power systems*. John Wiley & Sons.

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260.

Kong, W., Dong, Z. Y., Jia, Y., Hill, D. J., Xu, Y., & Zhang, Y. (2017). Short-term residential load forecasting based on lstm recurrent neural network. *IEEE Transactions on Smart Grid*, *10*(1), 841–851.

Kwiatkowski, D., Phillips, P. C., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of econometrics*, *54*(1-3), 159–178.

Lahouar, A., & Slama, J. B. H. (2017). Hour-ahead wind power forecast based on random forests. *Renewable energy*, *109*, 529–541.

Lange, M., & Focken, U. (2006). *Physical approach to short-term wind power prediction* (Vol. 208). Springer.

Lee, J., & Zhao, F. (2020). Gwec global wind report 2019. *Wind Global Energy Council, Tech. Rep.*.

Le Quéré, C., Jackson, R. B., Jones, M. W., Smith, A. J., Abernethy, S., Andrew, R. M., ... others (2020). Temporary reduction in daily global co 2 emissions during the covid-19 forced confinement. *Nature Climate Change*, 1–7.

Letcher, T. M. (2017). *Wind energy engineering: A handbook for onshore and offshore wind turbines*. Academic Press.

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*.

Liu, D., Chen, Q., & Mori, K. (2015). Time series forecasting method of building energy consumption using support vector regression. In *2015 ieee international conference on information and automation* (pp. 1628–1632).

Luppino, L. T., Bianchi, F. M., Moser, G., & Anfinsen, S. N. (2018). Remote sensing image regression for heterogeneous change detection. In *2018 ieee 28th international workshop on machine learning for signal processing (mlsp)* (pp. 1–6).

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and machine learning forecasting methods: Concerns and ways forward. *PloS one*, *13*(3), e0194889.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115–133.

Milligan, M., Schwartz, M. N., & Wan, Y.-h. (2003). *Statistical wind power forecasting for us wind farms* (Tech. Rep.). National Renewable Energy

Lab., Golden, CO (US).

Milligan, M. R., Miller, A. H., & Chapman, F. (1995). *Estimating the economic value of wind forecasting to utilities* (Tech. Rep.). National Renewable Energy Lab., Golden, CO (United States).

Müller, K.-R., Smola, A. J., Rätsch, G., Schölkopf, B., Kohlmorgen, J., & Vapnik, V. (1997). Predicting time series with support vector machines. In *International conference on artificial neural networks* (pp. 999–1004).

Nord Pool. (2021). *The Power Market.* `https://www.nordpoolgroup.com/the-power-market/`. ([Online; accessed 08 May 2021])

NVE, N. (2021). *Vindkraftdata-NVE.* `https://www.nve.no/energiforsyning/kraftproduksjon/vindkraft/vindkraftdata/`. (Accessed: 2021-05-22)

Palit, A. K., & Popovic, D. (2006). *Computational intelligence in time series forecasting: theory and engineering applications*. Springer Science & Business Media.

Ritchie, H., & Roser, M. (2020). Energy. *Our World in Data*. (https://ourworldindata.org/energy)

Rohrig, K., & Lange, B. (2006). Application of wind power prediction tools for power system operations. In *2006 ieee power engineering society general meeting* (pp. 5–pp).

Schölkopf, B. (2000). The kernel trick for distances. *Advances in neural information processing systems*, *13*, 301–307.

Sharda, R., & Patil, R. B. (1992). Connectionist approach to time series prediction: an empirical test. *Journal of Intelligent Manufacturing*, *3*(5), 317–323.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929–1958.

Statnett. (2016). *Dagens losninger i systemdriften.* `https://www.statnett.no/for-aktorer-i-kraftbransjen/systemansvaret/om-systemansvaret/`.

Statnett. (2018). *Slik fungerer kraftsystemet.* `https://www.statnett.no/om-statnett/bli-bedre-kjent-med-statnett/slik-fungerer-kraftsystemet/`. ([Online; accessed 08 May 2021])

Taieb, S. B., Hyndman, R. J., et al. (2012). *Recursive and direct multi-step forecasting: the best of both worlds* (Vol. 19). Citeseer.

Taieb, S. B., Sorjamaa, A., & Bontempi, G. (2010). Multiple-output modeling for multi-step-ahead time series forecasting. *Neurocomputing*, *73*(10-12), 1950–1957.

Theodoridis, S., & Koutroumbas, K. (2009). *Pattern recognition, edition four.* Academic Press, fourth edition Edition.

Torres, J. L., Garcia, A., De Blas, M., & De Francisco, A. (2005). Forecast of hourly average wind speed with arma models in navarre (spain). *Solar energy*, *79*(1), 65–77.

Tsay, R. S. (2014). *Multivariate time series analysis: With r and financial applications*. Chicago: Wiley.

Twidell, J., & Weir, T. (2015). *Renewable energy resources*. Routledge.

United Nations Environment Programme. (2019). *Emissions gap report 2019* (Tech. Rep.). Nairobi, Kenya: United Nations Environment Programme.

US Energy Information Administration. (2019). International energy outlook 2019: With projections to 2050.

Welling, M. (2004). Support vector regression. *Department of Computer Science, University of Toronto, Toronto (Kanada)*.

Wendell, L. L., Wegley, H. L., & Verholek, M. G. (1978). *Report from a working group meeting on wind forecasts for wecs operation* (Tech. Rep.). Battelle Pacific Northwest Labs., Richland, WA (USA).

Weron, R. (2014). Electricity price forecasting: A review of the state-of-the-art with a look into the future. *International journal of forecasting*, *30*(4), 1030–1081.

Wu, Y.-K., & Hong, J.-S. (2007). A literature review of wind forecasting technology in the world. In *2007 ieee lausanne power tech* (pp. 504–509).

Xiaoyun, Q., Xiaoning, K., Chao, Z., Shuai, J., & Xiuda, M. (2016). Short-term prediction of wind power based on deep long short-term memory. In *2016 ieee pes asia-pacific power and energy engineering conference (appeec)* (pp. 1148–1152).

Yoder, M., Hering, A. S., Navidi, W. C., & Larson, K. (2014). Short-term forecasting of categorical changes in wind power with markov chain models. *Wind energy*, *17*(9), 1425–1439.

Zhang, G. P., & Qi, M. (2005). Neural network forecasting for seasonal and trend time series. *European journal of operational research*, *160*(2), 501–514.

Zhang, J., Yan, J., Infield, D., Liu, Y., & Lien, F.-s. (2019). Short-term forecasting and uncertainty analysis of wind turbine power based on long short-term memory network and gaussian mixture model. *Applied Energy*, *241*, 229–244.