Norwegian University
of Life Sciences

Master's Thesis 2022 30 ECTS
Faculty of Science and Technology

Stacked LSTM for Wind Turbine Yaw Fault Forecasting Based on SCADA Data Analysis

Liubou Khorava
MSc. Environmental Physics and Renewable Energy

Acknowledgements

To my grandfather Sasha that used to tell a story about his friend studying for nine years and chuckle it at. It is a pity he can not see I have beaten that record. To my granny Nelly ad my father Sergey that unfortunately can not see the end of this studies journey.

To my husband, Rasmus, that has heavily promoted NMBU as a perfect study place and it did was a love at first sight indeed with too many specialisations to choose between. To my son who has been part of my studies for 4 of 6 years and had to study subject books with me. You are an apple of my eye.

To my mom for simply being there whenever I needed it.

To my dear norwegian frieds, when I thought, I had enough friends already: to Kristin, Kajsa, Andrea and Daniella. And special shout out to Martin and Kaia - I do not know how I would have managed so many exams without your help.

Ås, February 15th, 2022

Liubou Khorava

All models are wrong, but some are useful.

George Box

Summary

With the final change of our society from the fossil resources reliant only to the one with higher use of renewable resources, the necessity of improving the efficiency and profitability of renewable resources is of primary importance.

The performance of a wind turbine depends on the wind conditions as well as on the optimal extraction of kinetic energy and its transformation in the electricity. A wind turbine is a complex system and the coordination of all subsystems should be carefully orchestrated. The focus of this thesis is improvement of predict yaw faults relying on data collected via SCADA system. The data used in the experiment is kindly provided by Statkraft. A possibility to forecast the on-start of a fault alarm some time in advance gives possibility to implement required measures for eliminating the fault. Remote and automatic forecasting of such faults is utter importance for offshore wind parks that are emerging now all around the world.

The goal of this thesis is to improve the algorithms implemented by fellow student Tallaksrud (2021). The employed strategy focused on expanding the amount of information in the dataset. The study of influence of such parameters as rolling window length, a method used for handling the missing data, the number of features in the dataset and the number of time sequences in probability of a yaw alarm on-start. The preferential choice is a longer period in order to give time to fix the fault without risking long downtime and failure of other subsystems.

The results for 12 models are presented together with a single-layer LSTM model for comparison of stacked models predicting better than a simple one. The best results was produced with a single-layer model with the with MAE score of 0.001. The model presents varying forecasting behaviour as a result of randomness and instability. The conclusion is that the stacked LSTM models do not cope with solving the problem in the thesis

Sammendrag

Med den endelige skiften av våres samfunn fra å være bare fossile ressursene basert til en med høyere bruk av fornybare ressurser, behovet for forbedring av effektivitet og lønsomhet av fornybære ressurser er av primær betydning.

Ytelsen til en vindturbin avhenger av vindforholdene samtidig som optimal utvinning av kinetisk energi og dens transformasjon i elektrisiteten. En vindturbin er et komplekst system og koordineringen av alle delsystemer bør være nøye organisert. Fokuset i denne oppgaven er forbedring av yaw feil prediksjon basert på data samlet med SCADA-systemet. Data som ble brukt i forsøket er levert av Statkraft. En mulighet til å forutsi start av en feilalarm i forveien gir mulighet til å iverksette nødvendige tiltak for å eliminere feilen. Automatisk og mennesker-uavhengig varsling av slike feil er ekstremt viktig for offshore vindparker som dukker opp nå over hele verden.

Målet med denne oppgaven er å forbedre algoritmene implementert av medstudent Tallaksrud (2021). Den anvendte strategien fokuserte på å utvide mengden informasjon i datasettet. Undersøkelse av påvirkning av parametere som rullende vindulengde, en metode som brukes for å håndtere de manglende dataene, antall parametrene i datasettet og antall tidssekvenser med sannsynlighet for en yaw ralarm ved start. Det foretrukne valget er en lengre periode for å gi tid til å fikse feilen uten å risikere lang nedetid og svikt i andre delsystemer.

Resultatene for 12 modeller presenteres sammen med en enkeltlags LSTM-modell for sammenligning om stablede modeller klarer å predikere bedre enn en enkel model. De beste resultatene ble produsert med en enkeltlagsmodell med MAE-score på 0,001. Modellen presenterer varierende prognoseatferd som følge av tilfeldighet og ustabilitet. Derfor konklusjonen er at stablede LSTM modeller passer ikke til denne problmestillingen.

Contents

List of Figures

List of Tables

Part I

Background

Chapter 1

Introduction

1.1 Motivation

The demand for renewable energy is growing and will continue to grow in the near future as the result of the transformation of the global society from dependency on fossil renewable to a greater use of renewable resources. Wind energy is one of the promising resources and also the one that has experienced increase of total number installed wind parks and the increase in the efficiency. The interest for offshore wind parks requires improvements in the operation and management strategies of wind parks. Wind energy needs not only to be green solution but also a competitive solution.

With the better and more reliable extraction of wind resources Norway will be able not only to have an alternative source of renewable energy, but also goods to substitute partially the export of oil and gas as a reliable source of national income.

The possible improvement options for wind parks lies within the extraction and generation process of wind energy. The current systems become more and more data-driven and require less interventions from human resources. The digitalisation of such systems requires implementation of data-driven solutions able to cope with big amounts of data such as machine learning models.

1.2 The aim of the thesis

The aim of the thesis is to improve the strategy proposed by Tallaksrud (2021). The improvement strategy relies on the idea of increasing the amount of data put in the data set in order to study if machine learning models are able to learn patterns from this data and able to forecast the new instances of yaw faults.

As the yaw faults were of major interest, the choice was made to concentrate on these in this work. The wider choice of parameters will be taken, their relevance for the fault prediction will be studied and the narrower scope of the parameters will be used as the training data for the model.

The following aspects are tested under the experiment:

- testing of 2h output vs 10 hours output probability;
- testing of various windows length as input data;
- testing of dropping and imputation influence on the performance;
- testing of various number of features in the dataset;

Part II

Theoretical Background

Chapter 2

Wind Energy Theory

The chapter opens with the theory behind wind energy, its harnessing by wind turbines and wind parks. In continuation the overview of yaw faults influence on a wind turbine performance is presented to be followed the theory for time series data, its analysis applying machine learning. Such issues as non-stationary data, missing values and various methods for handling it are described in details as well as recurrent neural networks. The aim of this section is to provide the fundamental understanding for the theory relevant to the aim of the thesis.

2.1 Wind resources

The absorption of solar radiation by Earth's surface is uneven which leads to differences in the energy transferred to the air masses in the lowest parts of the atmosphere. These differences in the energy manifested through temperature and pressure gradients cause the circulation of the air masses around the globe. The more energetic and warmer ones around the equator rise to the higher latitudes, while the colder ones by the Poles descend to the lower latitudes. This air masses movement is known to us as wind and the kinetic energy of the atmospheric air masses is the wind energy which is of the major interest of this thesis. The direction of the circulation is influenced by the Coriolis effect on the global scale that results in the patterns illustrated in Figure 2.1. (Landberg, 2015)

The variations in the wind characteristics can be considered from spatial and temporal aspects. Temporal variations can be considered on the longer time scale as a decade, medium scale as a year and shorter scales as season, month, day, hours, minutes and seconds. The longer time scale variations are not so well understood and are hard to predict, while seasonal and even shorter time-scale variations are well understood, but still can be predicted only short time ahead. (Tony L. Burton, 2021).Turbulence is an example of wind speed variations on time-scale of minutes or even seconds. Its importance will be discussed further in 2.2.3.

From the spatial perspective the large scale variations in wind behaviour define the climate for a particular region. On a smaller scale the topography is of importance. The variations important for wind turbines happen in the lowest part of the atmosphere, the Surface Layer. The height

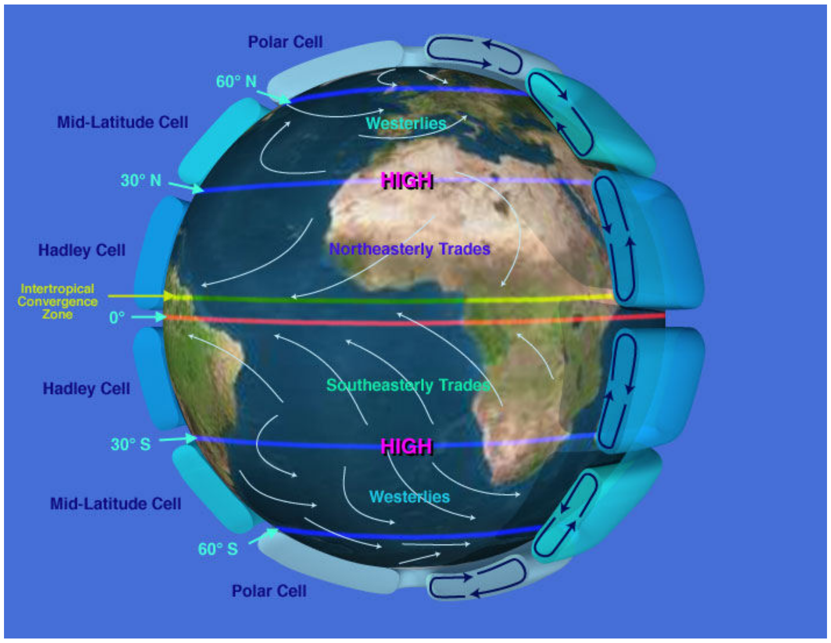


Figure 2.1: A simplified illustration of the circulation of the air masses around Earth (NASA, n.d.)

of this layer is about varies between 50 and 200 meters throughout the day and is influenced by the stability of the atmosphere. The Surface Layer is also known as well mixed layer and the logarithmic wind profile is applied to characterise the change in wind speed with height within this layer. An example of such a plot is presented in Figure 2.2. It can be observed that the wind has different profiles depending on the surface it follows. The wind behaviour on small spatial scale is more complicated due to the impact of obstacles, height, surface's roughness, vertical displacement of the profile due obstacles and orography. (Landberg, 2015).

The total combination of various variations makes wind an highly intermittent energy source that is challenging to forecast.

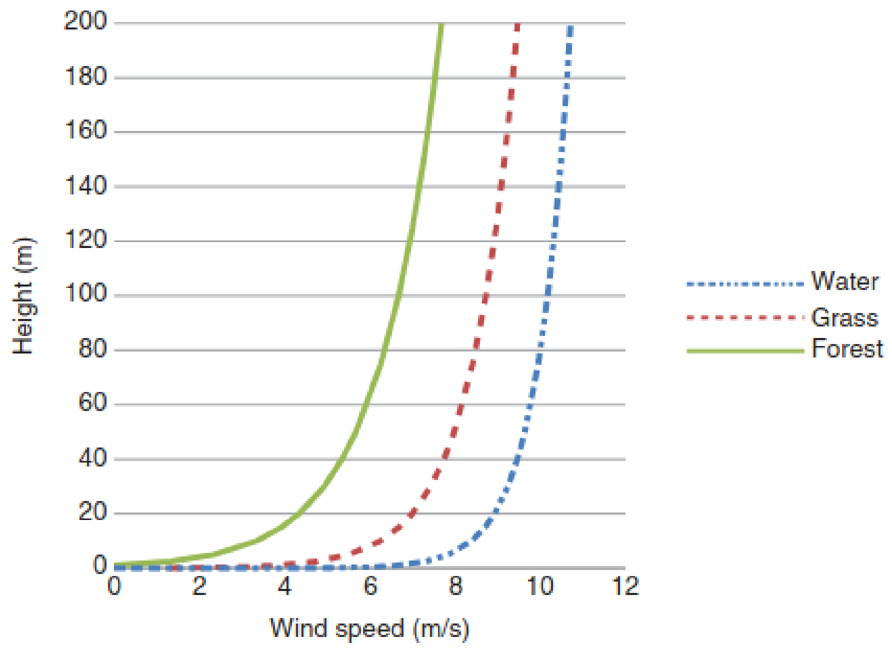


Figure 2.2: The logarithmic profiles for wind over three type of surface: water, grass and forest (Landberg, 2015).

2.1.1 Wind resources in Norway

The wind resources for Norway has been presented in great details in the reports from Kjeller Vindteknikk, where individual ones cover the average wind at the heights of 50 m, 80 m and 120 m, energy production, terrain complexity and icing. In the mainland part of Norway the average wind speed varies from 3 - 4 m/s up to 8 - 9 m/s in some areas high on the mountain range. The coastline has predominately wind speed at 8-9 m/s. These average wind speed measurements are considered for the height of 120 meters as it is more relevant for modern wind turbines. While the offshore wind speed has the average value of 9 - 10 m/s in the north and within approximately 100 km from the coast in the south to 10 - 11 further into the sea (NVE, 2008).

In accordance with the European Energy Agency the mean wind speed is higher in the northern coastal area of Europe than in Southern Europe and inland (EEA, 2021). But the available coastal area is limited due to the mountainous landscape. The mainland part of Norway scores between 10 and 50 RIX (Ruggedness Index) points. RIX is a measure for quantify the complexity of the terrain. Another important characteristic of the local wind climate is hourly wind energy generation. According to the wind resources maps from NVE the coastline has approximately between 3 000 and 3 500 hours of wind energy generation annually.

2.1.2 Wind energy harnessing

The simplified process of solar energy transformation into electricity via wind turbines is illustrated in Figure 2.3.

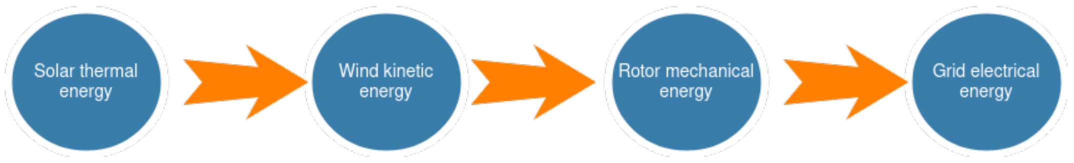


Figure 2.3: A simplified block diagram for the energy transformation process

The first step of this process was partially described above as the solar radiation absorption by the atmosphere which happens unevenly and creates the differences in the energy concentration. These gradients cause the air masses circulation with the kinetic energy. The wind energy is then captured by a wind turbine's blades, transferred to a rotor and is converted into the rotational mechanical energy. The rotation of the rotor is transferred further via accelerating gears to the generator and finally converted into the electricity. To be able to be transferred to the grid, the energy must pass a transformer for its voltage to be increased.

Betz Law or Betz's Elementary Momentum Theory

The amount of energy any wind turbine can extract from the airflow is bounded by the theoretical limit formulated by Albert Betz known as Betz Law. His theory assumes a range of simplifications:

- a turbine is to operate with a disc covering the area with the same radius as the length of the blades,
- a turbine is to operate without losses,
- the airflow is frictionless, steady, homogeneous and incompressible.

The starting point is to define the limit for the extracted power is to consider power P_1 before extraction, the mechanical power P_{mech} extracted by the disc and the power P_2 after the encounter as shown in Figure 2.4:

$$P_{mech} = P_1 - P_2. \tag{2.1}$$

Expression for each individual power can be derived from the relation of power as energy per unit of time

$$P = \frac{E}{t}(W), \tag{2.2}$$

where E is the kinetic energy of the airflow of mass m propagating with the velocity v

$$E_{kinetic} = \frac{1}{2} * m * v^2 (Nm). \tag{2.3}$$

Combining equations 2.2 and 2.3 leads to

$$P = \frac{\frac{1}{2} * m * v^2}{t} = \frac{1}{2} * \dot{\mathrm{m}} * v^2 (W), \tag{2.4}$$

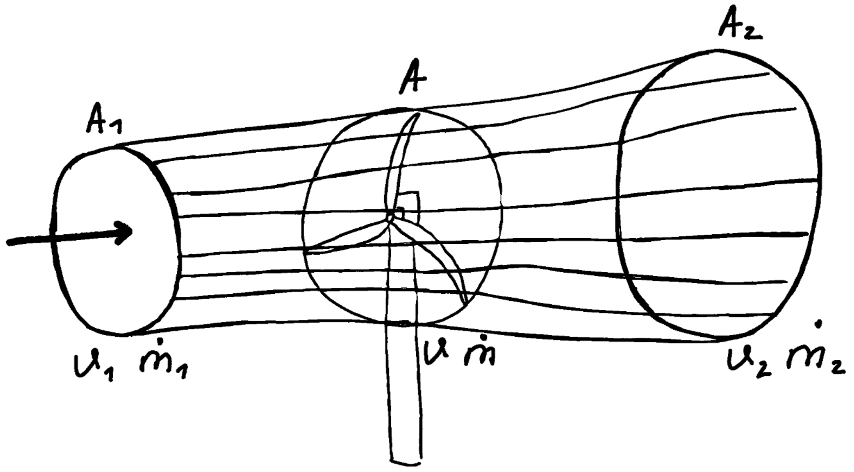


Figure 2.4: Diagram for the airflow before,at and behind the rotor inspired by (Hau, 2006)

where $\dot{\mathrm{m}}$ is the mass flow.

The mass flow in its turn derived by relating a volume of the airflow passing with the velocity v through a cross-section A with the thickness d = 1 and :

$$\dot{\mathrm{m}} = \dot{\mathrm{V}} * \varrho = A * \varrho * v(kg/s) \tag{2.5}$$

Combining equation 2.1, 2.4 and 2.5 gives P_{mech} as

$$P_{mech} = \frac{1}{2} * \varrho * A_1 * v_1^3 - \frac{1}{2} * \varrho * A_2 * v_2^3 = \frac{1}{2} * \varrho * (A_1 * v_1^3 - A_2 * v_2^3)(W). \tag{2.6}$$

The system is considered a closed one and then the requirement of the mass flow results expressed as

$$\varrho * v_1 * A_1 = \varrho * v_2 * A_2 \tag{2.7}$$

gives us the following equation for the power extracted

$$P_{mech} = \frac{1}{2} * \varrho * v_1 * A_1 * (v_1^2 - v_2^2) = \frac{1}{2} * \dot{m} * (v_1^2 - v_2^2)(W) \tag{2.8}$$

From equation 2.8 it is easy to conclude that the maximum power can be extracted if the airflow's velocity behind the rotor is zero, i.e the air stands still. This result is not obtainable physically, consequently the value of maximum extracted power depends on the optimal ration between the velocities before and after the disc.

The force F_{air} with which the airflow acts on the disc can be expressed

$$F_{air} = m * \overrightarrow{a} = \dot{m} * \delta v = \dot{m}(v_1 - v_2)(N), \tag{2.9}$$

by Newton's second law and is equal to the force exerted by the disc on the air by Newton's third law. Connecting the power required for it via work one obtains the expression for P_mech

$$P = Fair * v = \dot{m} * (v_1 - v_2) * v^{'}(W),, \tag{2.10}$$

Equating two expressions for P_mech from equations 2.8 and 2.10 yields that the velocity v of the airflow at the disc is

$$v^{'} = \frac{v_1 + v_2}{2}(m/s). \tag{2.11}$$

Substituting this expression for the velocity through the converter into equation 2.5 and combining it with the equation 2.8 results in the final expression for the mechanical power

$$P = \frac{1}{4} * \varrho * A * (v_1^2 - v_2^2) * (v_1 + v_2)(W) \tag{2.12}$$

The expression for the mechanical power of a free airflow P_0 can be acquired from the equation 2.12 with setting v_2 = 0 for zero power extracted. P_0 is required as the reference point to find the power coefficient c_p of a wind turbine.

$$C_p = \frac{P}{P_0} = \frac{\frac{1}{4} * \varrho * A * (v_1^2 - v_2^2) * (v_1 + v_2)}{\frac{1}{2} * \varrho * A * v_1^3} \tag{2.13}$$

The optimal relation between v_2/v_1 has been determined to be $1/3$, what leads to the maximum ideal power coefficient c_p equal to

$$C_p = \frac{16}{27} = 0.593 \tag{2.14}$$

C_p is the theoretical limit for the maximum power that a wind turbine can extract from the airflow. This value is lower for actual wind turbines which are presented in Section 2.2.

$$P = \frac{1}{2}\rho S C_p v^3 cos^3\alpha \tag{2.15}$$

where P is the power of the wind turbine, ρ is the air density, S is the swept area of the wind turbine, C_p is the wind utilisation coefficient, and v is the wind speed (Zhang, Yang, Li, Yan, & Li, 2021).

2.2 Wind turbines

A wind turbine, also known as a wind energy converter or wind power plant, is a complex machine that is used to harness wind energy. Wind turbines come in different shapes and sizes. The major types are horizontal axis wind turbines (HAWT) and vertical axis wind turbines (VAWT), drag-type or lift-type, downwind or upwind type(Hau, 2006). The modern wind turbines are predominantly downwind lift-type horizontal axis wind turbines (HAWTs) with three blades. It means that the rotor blades are places in front of the tower, the axis of rotation is in the same plane as ground. A simplified illustration of a HAWT is presented in Figure 2.5. The major components of a wind turbine are a nacelle and the rotor attached to the front of it. Both of them are resting on the tower.

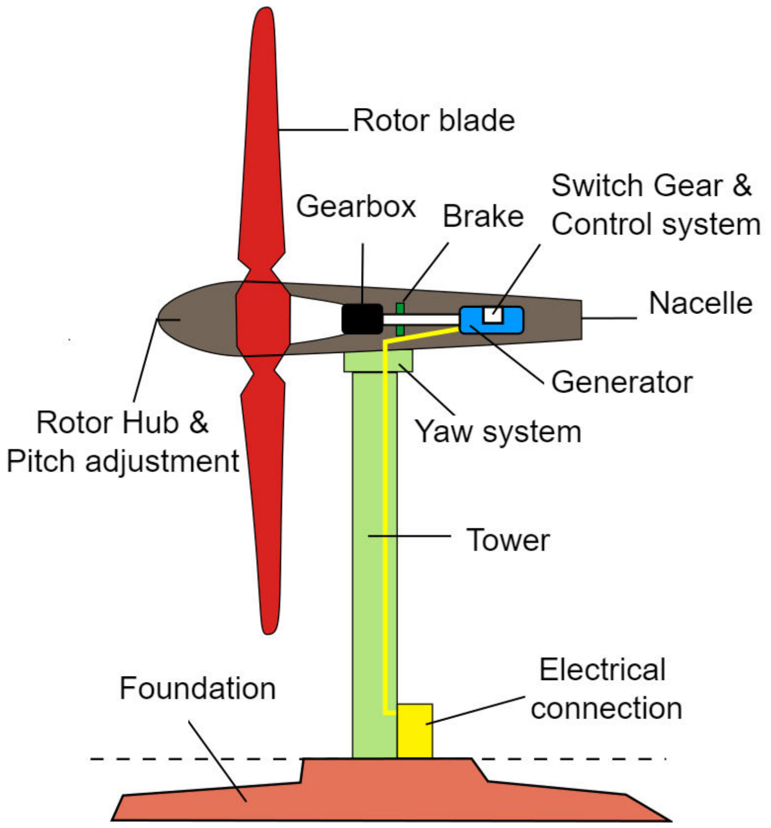


Figure 2.5: Major components of a horizontal axis wind turbine (German Wikipedia, 2011)

The majority of modern turbines utilise lift force for power extraction as it yields a higher power coefficient c_p discussed in the previous section 2.1.2. The rotor blades control can be performed either via pitch or stall and the alignment with the wind direction can achieved either via free or active yaw(Manwell, McGowan, & Rogers, 2010). The size of wind turbines blades and the hub height correspondingly has grown approximately twice as large within last two decades (Office of Energy Efficiency and Renewable Energy, 2021).The tendency is caused by the fact that the power extracted is linearly proportional to the area swept by the blades.

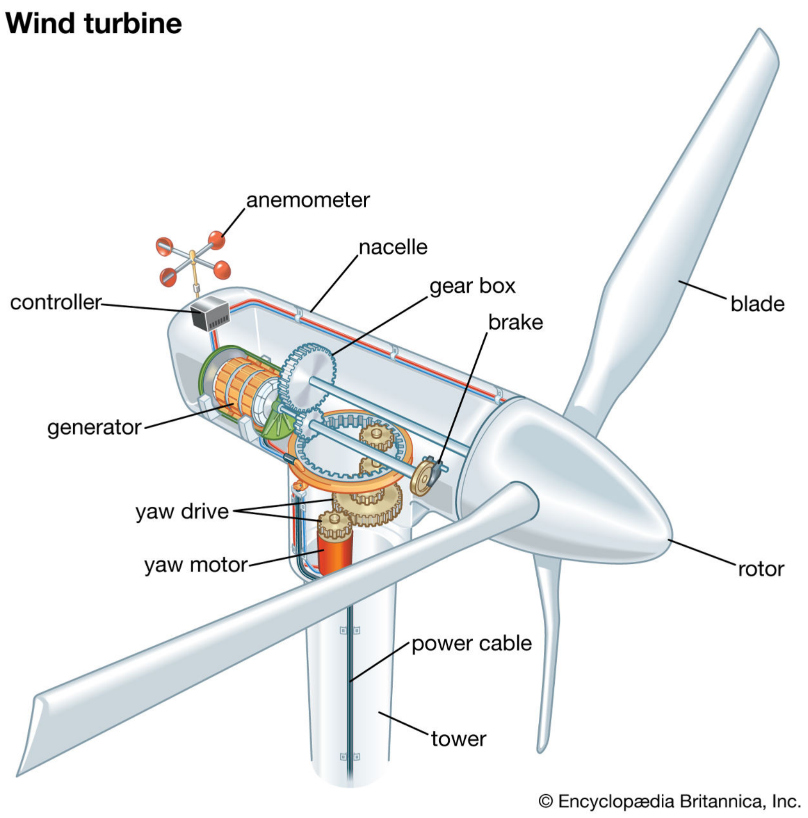


Figure 2.6: A detailed view of the nacelle with yaw and rotor subsystems. Image retrieved with permission from (Badurek, 2015)

A more detailed construction of a nacelle and a rotor is shown in Figure 2.6. The rotor is the part that converts the kinetic energy into electricity. The massive rotor blades are kept in place by the rotor hub and main bearing. Their position and thus the angle of attack by the incoming airflow is controlled by pitch adjustments. With sufficient force the blades come to movement and their low-speed rotation is transferred to the main shaft. These rotations are not frequent enough to create enough power and they are sped up with a gearbox. These high frequent rotations are passed further to the generator via the high speed shaft. A brake is attached to it to prevail too high rotations that would lead to the destruction of the turbine. The generator is either an induction or synchronous that requires constant rotations.

The whole nacelle-rotor unit is rotated into the wind by the yaw system. The maximum of wind energy extraction is obtained when the wind direction is orthogonal to the blades' plane. At the back of the nacelle sits a control unit that is responsible for managing the whole operation. The control relies on the measurements of sensors and is exercised with help of various controllers, power amplifiers and actuators (Manwell et al., 2010).

2.2.1 Yaw System

Yaw system is responsible for turning the rotor into the wind which allows it to extract maximum possible power from the air flow. Yawing can be also applied as a preventive measure against the structural deterioration by turning the rotor out of the wind direction in case of extreme weather conditions (Hau, 2006). On a wind park scale the yaw system allows to minimise the wake effect of the neighbouring turbines.

Modern industrial wind turbines are equipped with active yaw system, an example of which is shown in Figure 2.7. The major components are yaw drives, yaw breaks and a yaw bearing (Kim & Dalhoff, 2014). Additional components are locking system and control system (Hau, 2006). The other two types of yaw systems which are not used in large turbines are wind-vane yawing and free yawing.

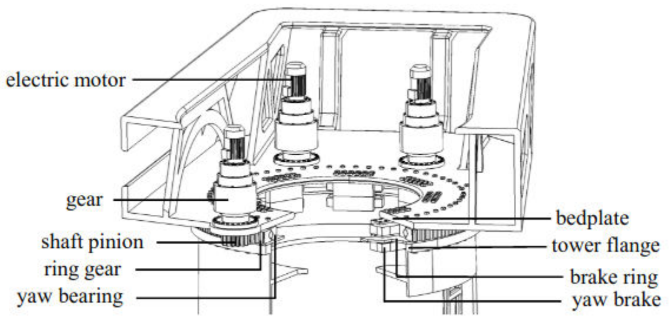


Figure 2.7: Yaw system. Image retrieved from (Kim & Dalhoff, 2014)

During the operation of the yaw system two objectives need to be satisfied. On the one hand, the yaw system should be sensitive and rapid to wind direction changes, on the other hand, not too rapid to avoid wearing the bearings. Additionally the nacelle and the rotor are such massive parts, that the regular speed of rotation around yawing axis is 360°per 5 min (Gasch & Twele, 2011).

Under the operation the yaw system experiences various types of faults that are discussed in detail in Subsection 2.2.7.

2.2.2 Turbine operation modes

The relation between the extracted power and the wind speed is demonstrated in Figure 2.8 with an empirical power curve. The curve is plotted based on the reference values provided by the turbine producer. The minimum speed required for a wind turbine to start operation is V_{cut-in} which is around 4 m/s in the reference power curve. The turbine's optimal mode for electricity generation is at V_{rated} which is approximately equal to 17 m/s. The turbine is shut off at the wind speed that exceeds 25 m/s and is denoted as $V_{cut-off}$. A power curve based on real values is demonstrated and discussed in section **??**.

Figure 2.8 demonstrates also operational modes of a wind turbine which strongly depend on the wind speed and can be used for analysis of a turbine's operation status. Thus if a power curve

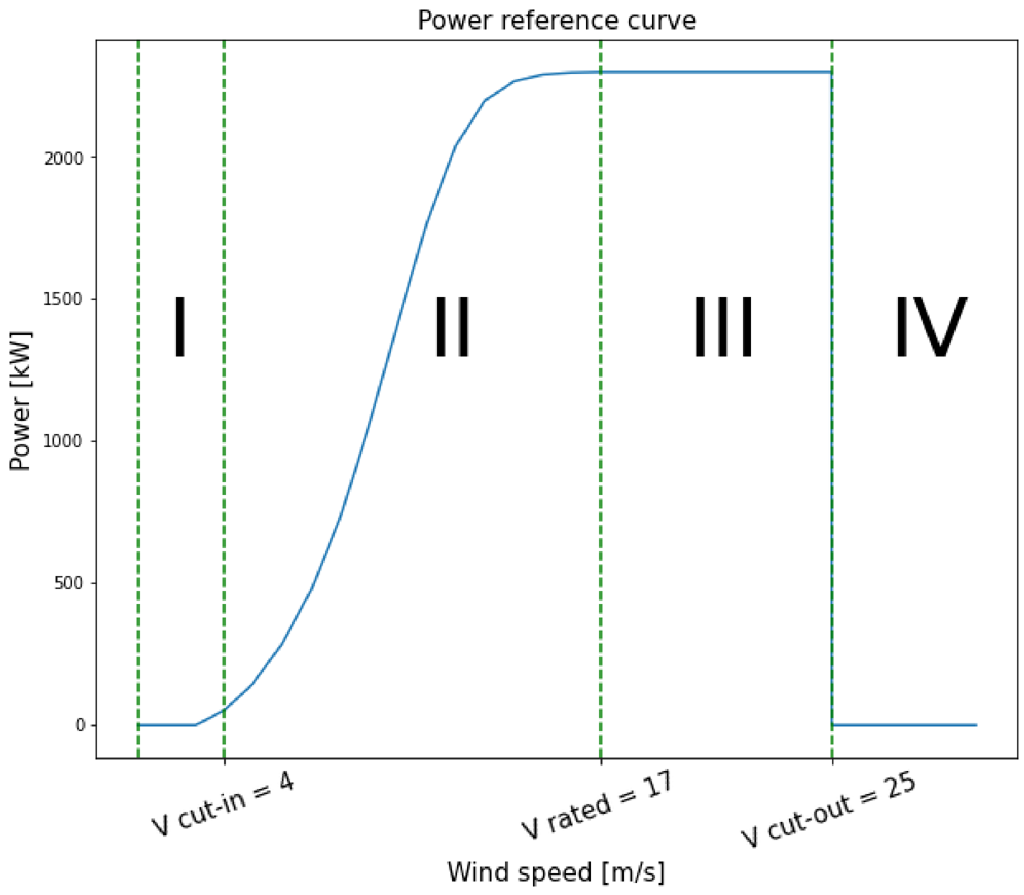


Figure 2.8: An example of reference power curve example with V_{cut-in}, V_{rated} and $V_{cut-off}$ specified as well as operational mode regions.

based on measurements does not follow the pattern that indicates faults in the operation.

In the notes to DTU course 46100 identifies Berg et al. (Berg, Mann, & Nielsen, 2013) the following modes for a wind turbine operation:

- normal power production which corresponds to region II in Figure 2.8;
- turbine start-up and shut down which occur at wind speeds V_{cut-in} and $V_{cut-off}$ correspondingly;
- failures, be it either turbine or grid related;
- operation under large yaw error;
- turbine in parked state when the blades are locked.

2.2.3 Wind park

A group of wind turbines installed in one location form a wind park. The size of a wind parks varies from small with several wind turbines to large ones with hundreds of them. Smøla wind park, Møre og Romsdal, was the largest wind park in Norway with total 68 turbines and maximum production capacity of 150 MW until Fosen Vind wind parks complex. The total number of wind turbines in all six wind farms in Fosen Vind is 278 and the total rated capacity of 1 GW. The installation of wind parks offshore is increasing as the wind conditions are more stable and the wind speed is higher as it was discussed on the example of Norway in section 2.1.1.

The operation of wind turbines in a park is further complicated by the wake effect due to the proximity of neighbouring turbines. Wake effect can be described as a shade cast by a turbine on other turbines downwind. The consequences of a wake are reduced power available in the airflow and increased turbulence of the air stream. The general strategy to minimise the effect of the wake is to put turbines at least three rotor diameters apart (Danish Wind Industry Association, 2003). The wind flow in the wake of a turbine at a distance of 4 diameters is still 30 % slower than the unaffected wind flow around.

2.2.4 Control systems

To maximise power output every wind turbine in a wind park should be able to adjust to intermittent weather conditions rapidly and run as smoothly as possible. The complexity of a single wind turbine alone requires remote and automatic systems in order to achieve these goals. Such systems are of crucial importance in the context of rising installation of offshore wind farms where the maintenance is costly and constitutes 25 - 30 % of the total installation costs (Röckmann, Lagerveld, & Stavenuiter, 2017).

Manwell et al. (Manwell et al., 2010) notes three factors important for a wind park operation: operation monitoring systems, analysis of power reductions causes and ample ways to improve the productivity. For monitoring data acquisition at wind parks SCADA (Supervisory Control and Data Acquisition) system is commonly employed. SCADA monitors and controls the operation of sensors installed on a wind turbine. The data is being gathered on an individual turbine level and on the wind park level. SCADA monitoring module gathers sensor signals for various measurements: temperature, pressure, lubrication level, vibration, position, wind speed and direction, current, voltage and power among many others. Measurements are conducted continuously on a second base and are typically averaged over 10 minutes periods. The gathered data is presented then as time series of measurements with both the parameter code, its value and the timestamp specified. The data is stored at a central computer for monitoring and controlling (Wang, Sharma, & Zhang, 2014).

The process of observing the components operation in order to identify any deviation from the normal operation is defined as condition monitoring and the corresponding system CMS. Condition monitoring systems allow to shift from time-based maintenance to conditional maintenance. There exists many types of wind turbine control monitoring systems, but the major goal of any of them is to predict faults before they occur. This allows to reduce operation and maintenance costs which are rising for the wind parks that has been installed a while ago and the warranties for them are expiring.

Previously CMS relied on observing a particular parameter for a component monitoring, such as vibration, temperature, pressure. With the advances in sensor and signal processing technology, big data processing and machine learning algorithms allows to employ analysis of a wider scope of data (Stetco et al., 2019).

The SWT-2.3-82 turbines at Smøla 2 have WebWPS SCADA and Web-based Turbine Condition Monitoring (TCM) system installed (Siemens AG, 2011). The WebWPS SCADA system provides information regarding electrical and mechanical systems, operation and fault status, meteorological data and grid related data (Siemens AG, 2011). The CM system monitors the wind turbine in real time and derives conclusion of a fault possibility relying on deviations from the reference vibration spectra (Siemens AG, 2011).

2.2.5 Wind measurements

The operation of a wind turbine and a wind park relies heavily on measurements of various wind characteristics: wind speed, wind direction, wind intensity. The choice of where to conduct these measurements is important as the power varies with the cube of the average wind speed. There are two choices for location of wind measurements and none of them is optimal. To place a measuring instruments too close to a wind turbine will influence the wind flow on the way to the turbines or will be influenced by their turning rotors. The only option to measure the exact wind speed at the turbine is when it is not operating. To put a measuring instruments far enough for them not to be influenced by wind turbines will not guarantee reliable measurement. The solution is to take the impact of the rotating rotors into the consideration and to adjust the measurements conducted in the vicinity of wind turbines. IEC standard IEC 61400-12-2 (ISO Central Secretary, 2013) provides the procedure for verifying the nacelle anemometry measurements with a transfer function.This function approximates the behaviour of a free airflow at the location of a wind turbine. The wind measurements are always impacted by the wind turbine and are not being collected live - reported live as it is too frequent. That is why it is better with automatic system for quick response.

2.2.6 Alarm

When a CMS registers a deviation between the actual measurements and the reference value an alarm is triggered.Every alarm is characterised with a duration time and is associated with predefined response. Possible reaction measures are the total stop of the turbine or restarting of a counter which are responsible for shutting down a particular system (Siemens, n.d.). The other type of triggered warnings is an event. The events do not cause the turbine to stop and is defined as an incident due to predefined conditions (Siemens, n.d.). The alarm and even logs are used for system status monitoring.

2.2.7 Yaw faults

The yaw system is associated with a range of yaw faults. The first type is connected to the inaccurate positioning of the nacelle in relation to the wind direction. In case when the yaw system does not manage to rotate the rotor in the wind an yaw error arises. Yaw error is defined as a misalignment of the nacelle position relative to the wind direction and can be calculated as

$$\theta_M = \theta_{WD} - \theta_N P, \tag{2.16}$$

where θ_M is the yaw misalignment error, θ_{WD} is the wind direction and $\theta_N P$ is the nacelle position. At yaw angle equal to 0 °the wind rotor is perpendicular to the wind.

The yaw angle noticeably reduces the power as it is demonstrated in Figure 2.9. The power generated by a turbine is directly proportional to the cosine of a yaw error as illustrated by equation 2.17.At yaw angle equal to 0 °and tip-speed ration λ equal to 8, the power coefficient c_p is at its maximum value of 0.59, at yaw angle = 20 °it is already 0.4 and as low as 0.1 at yaw angle of 60°.

$$\delta P \propto cos(\epsilon), \tag{2.17}$$

where ϵ is the yaw error. The yaw misalignment error is triggered when the yaw error measured by wind vanes exceeds some thresholds.

Besides the inaccurate positioning, the other often occurring types of faults associated with the yaw system are yaw gear tooth wear, abnormal noise, yaw limit switch fault, cable twisting and lubricant leakage (Yang, Chengbing, & Xinxin, 2012).

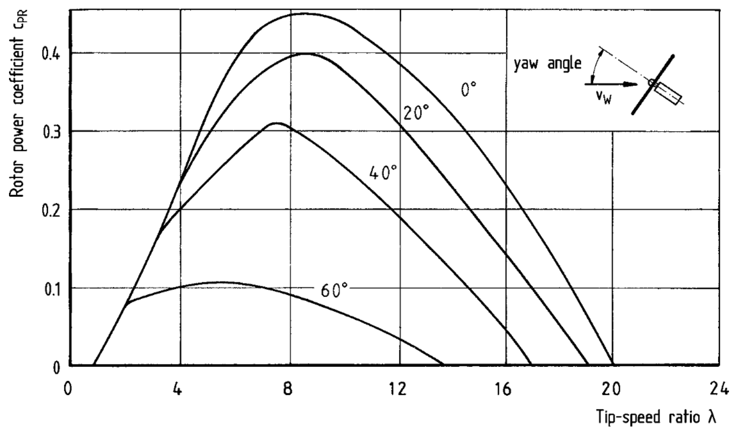


Figure 2.9: Decrease in the power coefficient of the rotor with increasing yaw angle. Image reprinted with permission from Springer-Verlag Berlin Heidelberg 2006 (Hau, 2006)

The parameters that can point to the presence of the yaw fault include measurements of temperature and pressure for the drives, stats for the break and bearing, as well as the yaw position and the nacelle position.According to the Hau (Hau, 2006) the vibration measurements can also be informative due to possible resonance development cyclically. Though according to the case paper (Liu & Hilmisson, 2014) yaw system fault can manifest itself through deviation of sensor measurements which are not directly connected with it. The paper provides two cases with fault arising in unexpected manner. In the first example the signals from the main bearing and tower accelerometers triggered alarm. After close inspection no faults were found in either of these systems. The change of the yaw damp assemblies helped to resolve the issue, for a short period though. In the second case, the measurements of the tower's accelometers deviated greatly from the reference value. After inspection of several systems, the blade adjustment and the yaw system loose screw tightening resolved the issued.

The instances of yaw faults are identified by SCADA as an fault or warning and are noted in the collected data. In accordance with reliability report (Sheng, 2013) the yaw system comes third among the top contributing subsystems to the overall failure rate by the number of failures per year. In regards of overall downtime is the yaw system also in the top three contributors after frequency converter and the pitch system.

2.2.8 Faults detection and forecasting

As it was mentioned earlier, the yaw system faults can manifest themselves in the sensor signals which are not directly connected with the yaw system. Models for wind turbine faults detection have been mostly relying on the physical model of a wind turbine (?, ?), which is a complicated process. The alternative solution is to employ a data-driven model. A data-driven approach implies the use of collected measurements from the system, such as SCADA operational data (Kandukuri, Robbersmyr, & Karimi, 2018). A third option is a hybrid of both. None of the approaches is known to be the panacea to the problem/go-to solution.

The use of the diverse SCADA data can be useful. To be able to forecast the yaw fault, the information gathered from SCADA needs to be analysed and an appropriate model should be fit to this data.

It is important to notice that a fault is a process, not an abrupt event. It means that the sings of a fault can be possibly detected in data well in advance it shall occur. Ahead of time fault forecasting permits the wind park operator to plan and resolve the issue before the fault arises (Encalada-Dávila, Puruncajas, Tutivén, & Vidal, 2021).

The detection of WT faults based on the SCADA data can help to reduce the cost as it relies on the data from the monitoring system already integrated in the wind turbine and does not require extra investments. In case of the success of fault forecasting, this approach will help to decrease operation and maintenance costs.

As the data gathered by SCADA system belongs to time series class, the theory related to that area is covered in the next part which is followed by the machine learning theory as a data-driven approach.

Chapter 3

Time Series Forecasting Theory

This section is dedicated to the theory for time series data exploring various aspects of how this data should be treated in comparison with cross-sectional data.

3.0.1 Time series data

Primarily data can be classified into three main groups: cross-sectional, time series and panel data, as illustrated in Figure 3.1. Cross-sectional data contains multiple observations of some parameters for different units for a single point in time. An example will be total power generated and wind speed for different turbines for a single day. Time series data consists of observations of a single unit collected over many points in time in ordered manner. Time series data can be recorded either continuously over a period of time or discretely at particular times or with a fixed interval. In regards of number of parameters in the data time series data can be either univariate or multivariate. Collection of various sensor measurements for a single turbine for a period of a year is an example for a time series data. Finally, panel data is combination of both as it is recorded as cross-sectional data over a period of time (B V Vishwas, 2020).

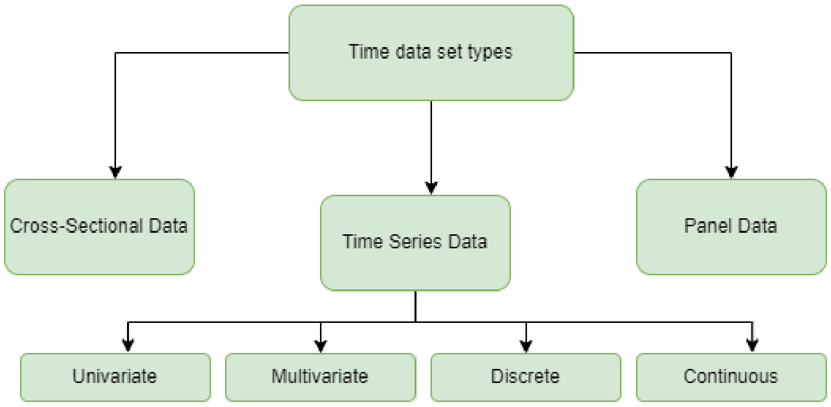
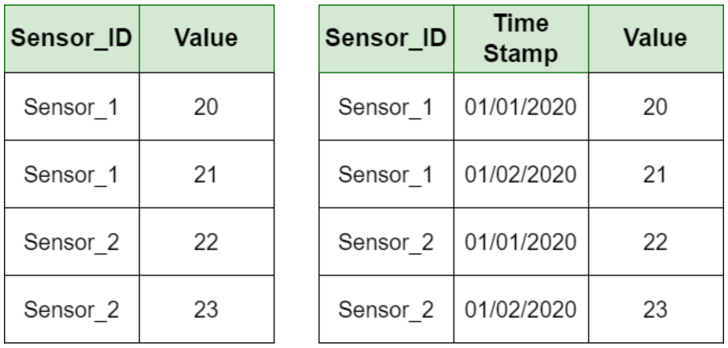


Figure 3.1: Types of time datasets

Figure 3.2 illustrates two datasets as they are usually presented for machine learning problems. The first one contains measurements' values and the identification label for a unit associated with this measurement. Entries in this dataset are to be treated equally in regards to time horizon for making a prediction. The second dataset though includes also the time stamp information for chronology of the measurements.

The abundance of time series data is rising as continuous monitoring and data collection is becoming more common in various spheres. This increase is caused by the digitalisation of various systems.

The sequential nature of the data plays a vital role in choosing specific data preprocessing techniques and specific models that can capture the patterns in the data. Traditionally was time series data studied based on statistical learning, recently the advances in machine and deep learning have widened the time series analysis and modelling toolkit. Currently, the best results can be achieved combining statistical methods with machine learning techniques (Nielsen, 2019).

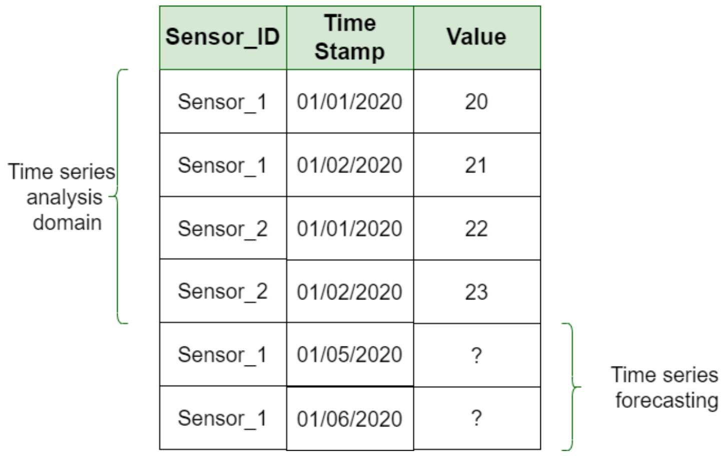


Sensor_ID	Value
Sensor_1	20
Sensor_1	21
Sensor_2	22
Sensor_2	23

Sensor_ID	Time Stamp	Value
Sensor_1	01/01/2020	20
Sensor_1	01/02/2020	21
Sensor_2	01/01/2020	22
Sensor_2	01/02/2020	23

Figure 3.2: A dataset without time data to the left and a time series dataset to the right. Inspired by (Lazzeri, 2020).

The difference in the approaches of these two fields is that statistical learning aims at finding the internal structure and patterns of the data based on mathematical and statistical models. Various methods of time series analysis help to build better datasets. While machine learning relies on the some of the statistical techniques to let a machine learn patterns from the data and forecast based on it. The difference between these two branches is demonstrated in Figure 3.3. While analysis is concentrated on building assumptions about the existing data, forecasting aims at predicting future values based on the historical records (Lazzeri, 2020).



Sensor_ID	Time Stamp	Value
Sensor_1	01/01/2020	20
Sensor_1	01/02/2020	21
Sensor_2	01/01/2020	22
Sensor_2	01/02/2020	23
Sensor_1	01/05/2020	?
Sensor_1	01/06/2020	?

Time series analysis domain
Time series forecasting

Figure 3.3: An illustration what is the difference between time series analysis and time series forecasting. Inspired by (Lazzeri, 2020)

3.0.2 Time series components

Real-world data are often impacted by deterministic components in data which are not so easy to spot at first glance by visual analysis. These components include trend, cyclical or seasonal and irregular. The mathematical representation of time series data as combination of these components is given by the following equation:

$$Y_t = f(T_t, S_t, E_t), \tag{3.1}$$

where

- Y_t is the time series values at time stamp t;
- T_t is a trend-cycle or long-term variation;
- S_t is the seasonal component and cyclic component;
- E_t is the irregular or residual component.

Figure 3.4 illustrates the impact and the pattern of all three components or signals due to the deterministic nature. Trend defines a long-term increase or decrease in data which can change direction. Seasonality and cyclicity are closely related, but differ in the length of the intervals for repetition. Seasons tend to have relatively fixed duration, while a cycle can repeat itself within various time period. Irregularities represent the component that can not be explained from the data. This component can also be denoted residual meaning something that is left after all the other components. The effect of these components can be either additive or multiplicative.

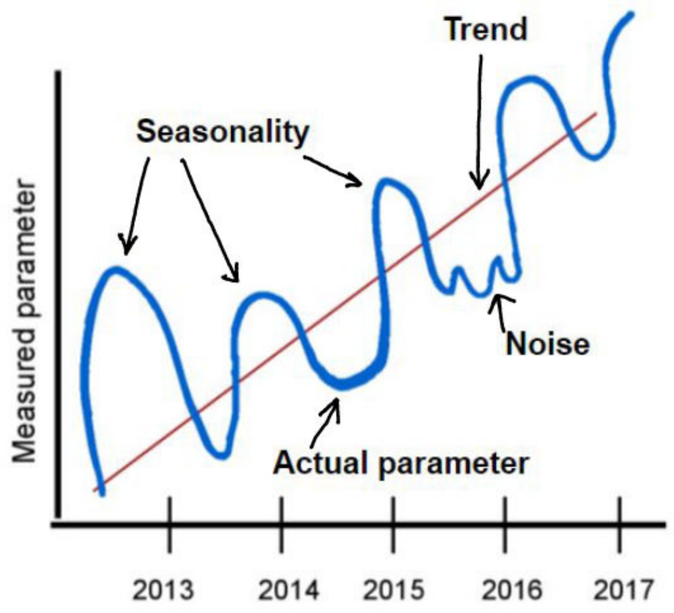


Figure 3.4: Illustration of seasonal, trend and noise components of time series in an actual plot of a parameter measured over time. Inspired by (Lazzeri, 2020).

Identification of these components and their impact on the whole dataset is crucial for creating a reliable statistical model and not the least for the machine learning model. Detrending and deseasoning are one of the methods applied to data in order to extract seasonal and trend components. The removed components can be used to engineer new features to build a better statistical forecasting model.

3.0.3 Stationarity

The notion of time series components removal is tightly connected with the concept of stationarity. The major criteria for data to be stationary are constant mean variance and covariance. Variance measures spread of a dataset around its mean value, while covariance measures of relationship between two random variables (Bjørnstad, 2017) .(Malt, 2020). That implies that stationary data fluctuates around the mean values during the time of observation (Pal & Prakash, 2017).

The invariance of data distribution over time simplifies the process of pattern identification. The major assumption is that the parameters are independent of time and each next period can be forecasted based on the previous ones. The stationarity can be subdivided into strong and weak. The strong one requires the invariance of all statistical parameters, while weak one requires the invariance of mean and auto-covariance functions (Lazzeri, 2020).

The non-stationarity manifests itself in data in the form of seasonal and cyclic variations or as trend. Such behaviour is expected of the weather data merely due to the existence of seasons and the changes in the weather in accordance.

It is preferential to make data stationary for more consistent results and easier data analysis for classical statistical methods, i.e. autoregressive (AR), moving average (MA), autoregressive integrated moving average (ARIMA) methods and seasonal autoregressive integrated moving average methods (SARIMA) (Lazzeri, 2020). Due to being a subset of linear regression models, these methods fail to recognise non-linear patterns in data in some cases. Deep learning models that are of focus for this paper are known to be able to cope with the data non-stationarity.

Chapter 4

Machine learning theory

This section covers the theoretical grounds for machine learning (ML) relevant for handling the yaw faults forecasting based on a wind turbine's operational data. The theory overview goes from the basic principals of machine learning and its types to deep learning as sub-field of machine learning. The final topic is deep learning models suiting time series data forecasting.

4.0.1 General definition and classification

The data flow of operational data for a wind turbine contains data points from hundreds of sensors with measurements conducted for over decades. The accumulated volume of data is out of scope for humans to process and analyse. Machines are more effective in that regard. The digitalisation of of all spheres of human activities requires also more rapid and ample tools for data collection and responding. Machine learning, a branch of Artificial Intelligence (AI) and computer science, that focuses on learning patterns from large amounts of data in order to make predictions.

The primary categories of machine learning are presented in Figure 4.1. The governing principle in distinguishing them is the type of learning. Reinforcement learning focuses on creating a system that can maximise its reward, i.e. improve its performance, relying on interactions with the environment. Unsupervised learning also lacks guidance, but its goal is to identify patterns in data and extract meaningful information (Raschka & Mirjalili, 2019).

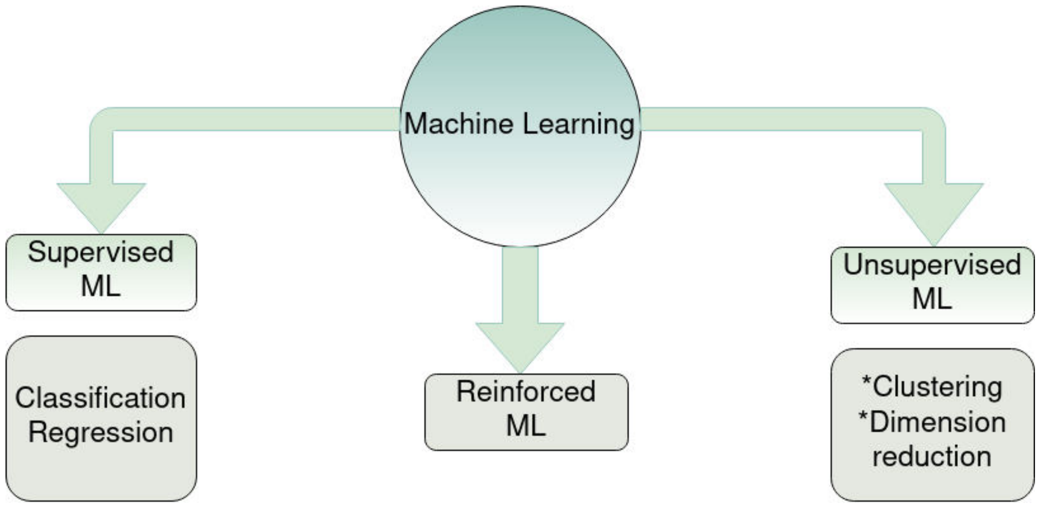


Figure 4.1: Major types within machine learning with further subcategories for supervised and unsupervised ML.

Supervised learning unlike two other types relies on labelled data. Such a learning algorithm catches patterns in data in regards to labels as ground truth. Thus a model can predict data unknown to it. Depending on the type of output supervised model can be either classification or regression type. Based on the estimation metrics values one can decide whether model is good to generalise from data or not.

4.0.2 Classification and Regression

In regards of response variable supervised algorithms can be divided into classification or regression algorithms. Problems with categorical or qualitative response are tackled by classification algorithms. The flow for a classification algorithm is illustrated in Figure 4.2. The classification problem can be either binary or multiple classes and label can be presented only in numerical form.

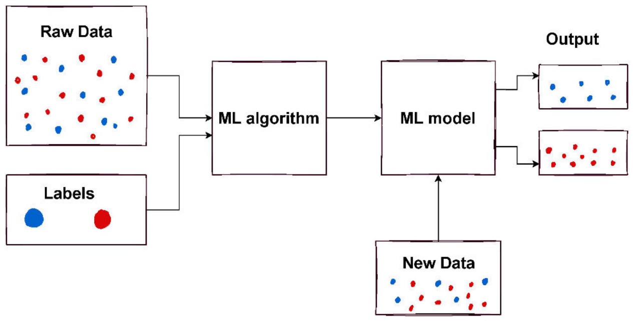


Figure 4.2: A diagram for a supervised binary classification process

Problems with continuous output as demonstrated in Figure 4.3 are handled by regression models. The input parameters are presented in the same format as in classification problem, while labels are quantitative. The patterns learnt from the input data are saved in the form of model parameters and this knowledge is reused later on new data to predict its output. The type of an output and thus a model is decisive in choosing evaluation metrics and suitable models.

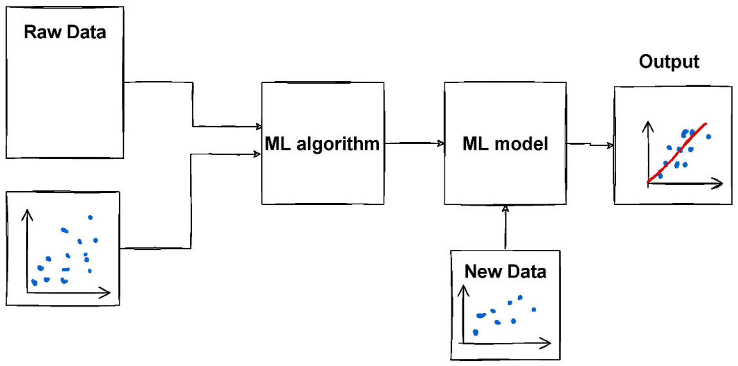


Figure 4.3: A diagram for a regression algorithm with a continuous output for labels.

4.0.3 Deep Learning

Deep Learning is a brunch of Neural Networks, which in their turn are part of Machine Learning. Deep learning algorithms are based on artificial neural networks and it requires at least four layer for an ANN to be considered deep (Kavlakoglu, E. , 2020). Presence of several layers with multiple nodes in each enables deep learning algorithms to automate data analysis and the use of large datasets.

The concept of artificial neural network was first formulated by Warren McCulloch and Walter Pitts in 1940s and later implemented by Rosenblatt as perceptron (Raschka & Mirjalili, 2019). A general representation of a single layer neural network, also perceptron, is shown in Figure 4.4. A single layer neural network is referred to as a node. The structure of the artificial neuron mimics one of a biological neuron. In Figure 4.4 the input data is presented by x_1, x_2, x_m which together represent an input layer. The inputs of all individual units plus a bias are combined together with corresponding weights to yield the net input function Σ. The bias allows to introduce a shift in the activation function and is analogous to a in the linear regression line $y = a + bx$. The bias insures activation even if all the inputs values are zero (Abdovic et al., 2019). The net input proceeds to the activation function which determines how the output is calculated. Output serves also for improvement of the performance.

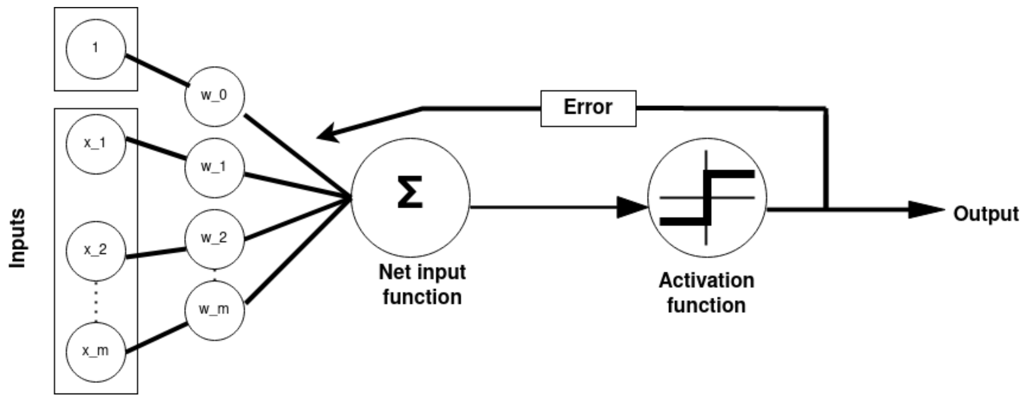


Figure 4.4: A diagram for general concept of perceptron or a single layer neural network. Inspired by (Raschka & Mirjalili, 2019)

A simple artificial neural network consists of at least three such simple neural layers as illustrated in Figure 4.5. The first layer referred to as the input layer is followed by the hidden layer in the middle and the output layer in the end. The input and output layers' function is to receive the input signals and to send out the output signal as it comes straightforward from their names. The hidden layer serves for computation purposes. Neurons in the layers are interconnected with weight assigned to each connection. Each neuron in a hidden or output layer is a weighted sum of neurons in the preceding layer. The net sum is to be passed over to the activation function and the result of it is sent to neurons in the next layer. The final step is the output layer whose shape depends on the category of learning. In case of supervised ANNs, classification algorithms have a node for each class in the output and regression models have a single node in the output. The output values are compared to labels and the result is referred to as a loss or cost function. The loss function is calculated as squared difference between two values. The loss function is used to adjust the output results via backpropagation process which means that the error in the output layer is carried backwards through the neural network.

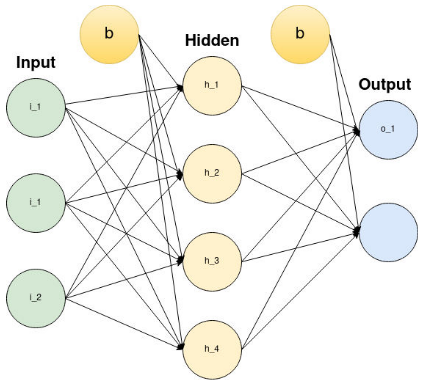


Figure 4.5: A simple artificial neural network with three layers and binary output implying classification problem (Burnett, C. M. L. , 2006)

The process of backpropagation is demonstrated in Figure 4.6, where every circle denotes a node. Green ones belong to the input layer, the yellow ones to the hidden layer and the blue ones to the output layer. Firstly the output is acquired from the model at step 1, then it is compared with the target at step 2 as

$$\sigma^{out} = a^{out} - y, \tag{4.1}$$

where a^{out} is the activation vector of the output layer and y is the target vector. The deviation between these two vectors is the error of the output layer and is denoted as the loss function. As step 3 the loss gradient is calculated in regards to the preceding hidden layer which is marked with yellow. The loss gradient $\frac{\partial}{\partial w^{out}_{i,j}} J(W)$ is the product of the activation function of the hidden layer a^h_j and the error term σ^{out}_i from the output layer for each node a^h_j respectively or as the mathematical expression

$$\frac{\partial}{\partial w^{out}_{i,j}} J(W) = a^h_j \sigma^{out}_i. \tag{4.2}$$

The subscripts i define the numbering of the the previous layer nodes, i.e. the output layer, and j the numbering of the current layer, i.e. the hidden layer in this case. The procedure for the error calculation and its propagation is repeated for the hidden layer at steps 4 and 5 correspondingly. The hidden layer error σ^h is calculated as the product of the output layer error σ^{out}, the weights matrix W^{out} transposed and derivative of the activation function a^h

$$\sigma^h = \sigma^{out} (W^{out})^T \odot \frac{\partial \phi(a^h)}{\partial a^h}. \tag{4.3}$$

And the loss gradient of the hidden layer $\frac{\partial}{\partial w^{h}_{i,j}} J(W)$is found in analogy to the the loss gradient of the output layer. The weights between the hidden layer and the input layer are adjusted by the hidden layer error in the same mannor as step 4.

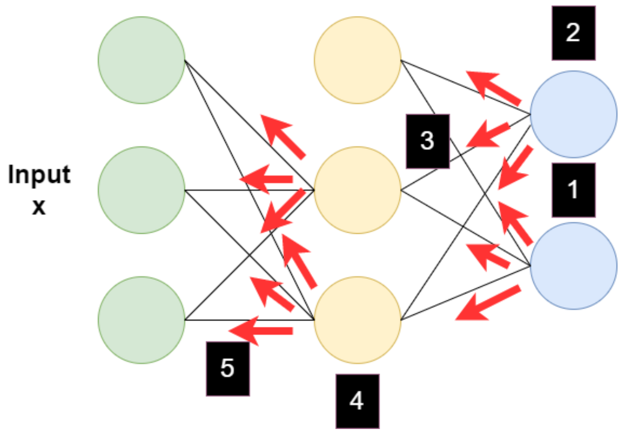


Figure 4.6: A diagram demonstrating backpropagation of loss function's gradient through the neural network. Inspired by (Jones, M. T. , 2017)

The backpropagation calculation is done for the whole output-input path at once and the process is repeated for each training epoch, i.e. individual training attempt. During the testing, the new data is fed to the neural network and the output error value is the classification accuracy metric or the regression Mean Squared Error (MSE) metric.

The ANN above is an example of Feed Forward Neural Networks (FFNNs), which belong to supervised learning. Other types of supervised neural networks are Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). Unsupervised neural networks include Autoencoder and Generative Adversarial Networks (GAN). Reinforcement neural networks are comprised of Networks for Actions, Values, Policies and Models (Fridman, 2019). The type that has been of particular interest for solving time series problems is Recurrent Neural Networks.

RNN

RNN is a category of artificial neural networks that includes weighted connections not only between layers, but also within a layer (Jones, M. T. , 2017). Such mechanism allows RNNs to 'remember' the previous input while processing new one as state nodes or context nodes as indicated in Figure 4.7 by nodes c_1, c_2, c_3, c_4. Sequence data is a type of data that requires such mechanism since sequence of the input values has as much meaning as values themselves. Time series data is one types of the sequence data where each data point is associated with a time stamp and order of data is determined by time stamps array (Raschka & Mirjalili, 2019).

The input and output of RNN models can both be sequences or one of them can be a sequence. Based om that the following categories of sequence problems can be identified as it is also demonstrated in Figure 4.8:

- one-to-one problems with one input and one output as image classification, for example. Both input and output thus are a single time-step with one or multiple features.
- one-to-many with one input and a sequence as output. A classical example is an image as input and its description with a sequence of words as output.
- many-to-one with a sequence as input and a single time step as output is an example of a text classification or alarm classification or regression.
- many-to-many with sequences for both input and output. Those can be either non-synchronised as in the first scheme or synchronised as in the second one. An example of non-synchronised is translation of the text from one language into another or regression problem with an output longer than one time step. While a synchronised many-to-many a video classification with every frame being a time step (Malik, 2019).

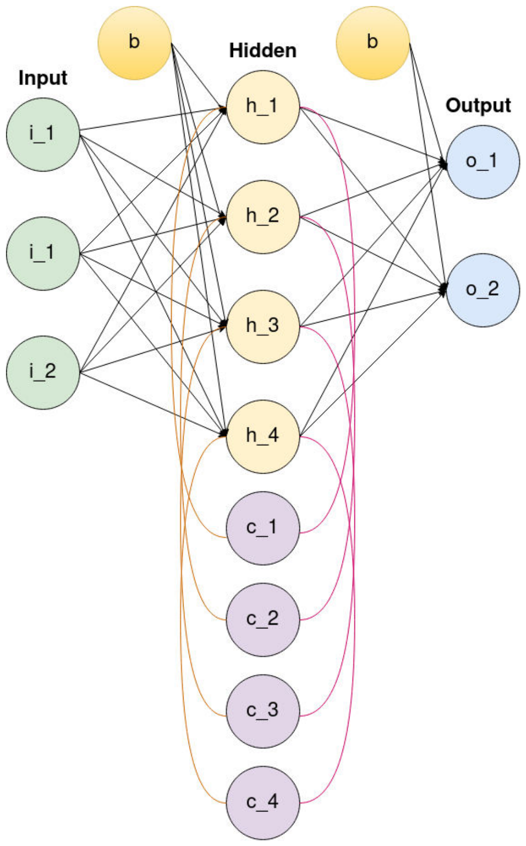


Figure 4.7: A diagram of a simple RNN model that is able to remember the previous input as context node - c_1, c_2, c_3 and c_4 in Figure. Inspired by (Jones, M. T. , 2017)

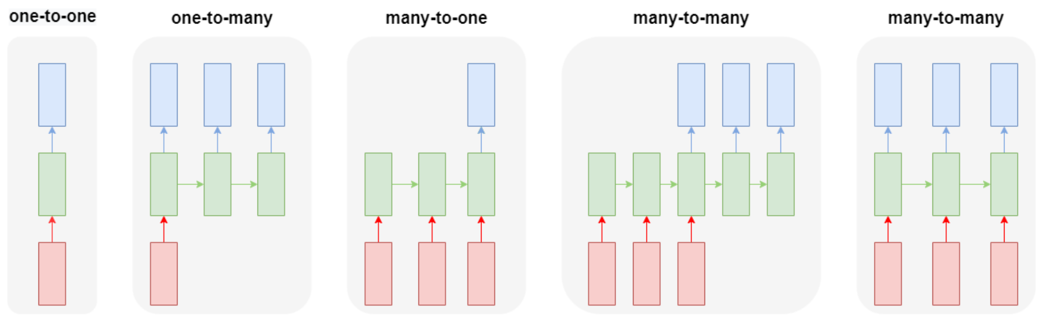


Figure 4.8: Various categories of sequence problems that can be modelled with RNN. Every rectangular cell is tensor and every arrow is a function between cells. Red cells represent input layer, green cells - RNN's state, blue cells - output layer. Inspired by (Karpathy, 2015)

It is more correct to define recurrent neural networks not as a class, but a collection of neural networks with different arrangements. Each of the topologies is suited best for different tasks (Jones, M. T. , 2017). Long Short-Term Memory topology has proven to perform outstandingly with time series data (Raschka & Mirjalili, 2019). This is due to the fact that tuning of the model is conducted with the help of back-propagation through time (BPTT). Time series with large input sequences suffer from vanishing or exploding of gradients. This happens because for the weights to be updated for one output they need to be multiplied by itself t-k times resulting in w^{t-k} where t is time step and k is the state. Eventually the gradient will vanish If $w < 1$ and explode if $w > 1$. This problem can be tackled by several techniques and Long Short-Term Memory type of RNN is one of them.

LSTM

LSTM networks are distinguished among RNNs by their ability to learn dependencies in data over long time series and thus able to cope with the gradient problems discussed above. Alike all RNNs LSTM have a chain-like structure where repeating element though is a memory cell and not a single layer neural network. Figure 4.9 presents a single LSTM memory cell as illustration.
The LSTM cell controls how much information to remember from previous cell, how much information to retain from the current one and how much to feed to the next one. These tasks are performed by so called gates. The first gate f_t, that stands for forget gate, decides how much information should be kept from the previous time step. The sigmoid function σ employed for that transforms input values x_t, that are concatenated with signal from the previous time step h_{t-1}, into values between 0 and 1. The value 0 implies that everything is forgotten and 1 means that everything is retained. Later the output of this gate is added to the cell's state signal C_t via multiplication with the state signal C_{t-1} from the previous time step.

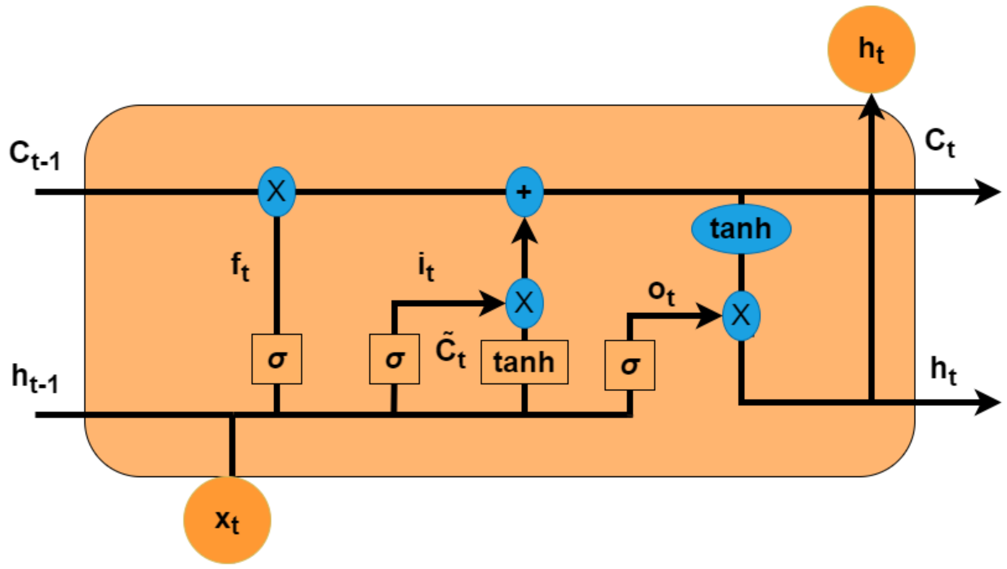


Figure 4.9: A diagram for LSTM memory cell. Inspired by (Biswal, 2021)

The input gate i_t decides upon which information should be added to the cell's state signal C_t. The input gate is a combination of sigmoid layer σ and $tanh$ layer. While the first decides which values to retain, the hyperbolic fucntion $tanh$ determines the level of values importance, ranging it from -1 to 1. The output of the product between these two layers is added to the cell's state signal C_t.

The third gate, referred to as the output gate o_t, receives values which were determined to be part of the output with the help of σ function and multiplies them with the cell's state signal with importance assigned to values via $tanh$ function. The cell's output is then transferred to the memory cells, one at next time step and one in the next layer as h_t (Biswal, 2021).

4.1 Missing data and imputation

It is impossible to prove that data is truly missing at random and it is unlikely that missingness is truly random in most real-world occurences.

Missing values can be handled with neural networks. In new study on applying those to missing values problem pointed out E^{2GAN} as state-of-the-art (Fang & Wang, 2020)

Any real world data suffers from missing values. The most common reasons for missing data are failure in acquiring or recording data, high noise or full storage space. An example of acquisition failure is malfunctioning of sensors or signal communication failure. A missing value is usually denoted NAN, i.e. Not a number, 999, ??? or unknown. Missingness of data is informative.

The data can be missing in three ways as illustrated in Figure 4.10:

- Missing Completely at Random (MCAR) refers to data that is missing irregardless to other factors. The data is absent completely independent of other parameters. Such missing values occur in case of sensor failure.

- Missing at Random (MAR) includes missing values of the parameter that depends on the other variable but the data points are absent randomly.
- Missing Not at Random (MNAR) defines missing values that are missing with the same pattern as other variables.

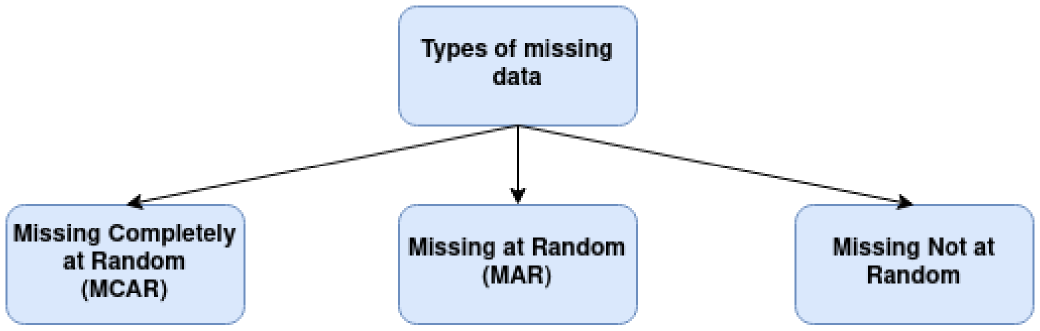


Figure 4.10: Types of the data points missing

The common approach to drop missing values is not recommended with time series data as it leads to gaps in time series and thus loss of sequence information (Sæther, 2021). Dropping can be performed only if the missing values represent small fraction of the dataset and only of the data is missing randomly. The common imputation technique used with cross-sectional data - to impute with global mean or global median - can not be applied either due to seasonality, trend and cyclic components in time series data. The understanding of these components is crucial for missing values imputation. The understanding of autocorrelation of the variables with itself can provide useful information which time periods can be used for imputation of a particular missing value. The autocorrelation method allows to measure how much the consequent time steps for the same variable are correlated with each other. If the autocorrelation is high the use of a variable as the training variable for forecasting itself may improve the performance of the model.

At Strata Data Conference in 2019 Jeon and Whitehead (2019) presented the following imputation methods for time series data as presented in Table 4.1.

For time series forecasting only the past values can be used. So the list of the imputation methods appropriate for time series data with ML/DL reduces to:

- Linear Interpolation
- Polynomial Interpolation
- Kalman Smoothing
- K-Nearest Neighbours
- Random Forest
- Multiple Singular Spectral Analysis

Imputation Methods for Time Series Data	
Univariate Time Series Imputation	Multivariate Time Series Imputation
Mean (Median)	K-Nearest Neighbours
Last Observation Carried Forward	Random Forest
Linear Interpolation (LI)	Multiple Singular Spectral Analysis
Polynomial Interpolation	Expectation-Maximisation
Kalman Filter	Multiple Imputation with Chained Equations
Moving Average	
Seasonal adjustment + LI (V., 2021)	
Random	

Table 4.1: Existing methods for data imputation for univariate and multivariate time series data.

- Expectation Maximisation
- Multiple Imputation with Chained Equation (Jeon & Whitehead, 2019).

A novel approach is to use RNN models for missing values imputation (Fang & Wang, 2020). One of the possibilities for a wind park it so extract the same features from the neighbouring turbines data as the one that requires imputation. Than fitting a RNN model onto the concatenated dataset with these features allows to predict the missing values.

The choice of whether to keep or to throw out the missing data depends on data analysed and if it permits to sacrifice the time periods. Deep learning models are generally considered to be able to handle well missing values. In (Chollet, 2018) the imputation with 0 is considered safe when applying neural networks, provided this value isn't a meaningful value.

(Jeon & Whitehead, 2019) " Missing values can arise from faulty sensors, incomplete records, mistakes in data collection, unavailability to report certain information (at a given time point) and others. The canonical ML/DL modeling would use both past and future values for predicting the missing values, while real time prediction can use only the past values.

4.2 Feature Selection

Feature selection allows to reduce the complexity of model and training time and the prediction loss. The goal of feature selection is to find the minimum subset of the features that contain just enough important information for the model to predict the target. The features are selected based on the optimisation criteria. That means that the methods for classification and regression models differ.

According to (Macas et al., 2014) reduction of the features by half still allows RNN models to keep the same performance rate.

One of the commonly used methods used to feature selection is Pearson's correlation coefficient. It is calculated as linear correlation between two features X and Y as

$$\rho = \frac{cov(X, Y)}{\sigma_X \sigma_Y}, \tag{4.4}$$

where ρ is the Pearson's correlation coefficient, cov(X,Y) is the covariance between X and Y and σ_X and σ_Y are the standard deviations of X and Y correspondingly (Zaiontz, n.d.). The values of the coefficient range from -1 to 1. Where the value defines how strongly features are correlated linearly with -1 and 1 being the strongest. The sign reflects the direction of the correlation: direct and reverse. As the Pearson's correlation coefficient measures the linear correlation it can not capture complex non-linear relation between features in time series data.

In (Tang, Liang, Wu, & Wang, 2021) XGBoost model was used for feature selection. This is done with the help of feature importance method. To understand how feature importance works, the concept of XGBoost or Extreme Gradient Boosting should be explained first. XGBoost relies on the idea that a single model is not sufficient even if complex, though an ensemble of base models, which are weak learners on their own, combines together in a sequence or in parallel allows to improve the total performance (Sundaram, 2018). The importance is calculated for every single decision tree relying on how much each split point improves the performance, corrected by the number of the observations the node is responsible for. The feature importances are then averaged over all decision trees (Brownlee, 2016). The resulting scores can be obtained a list of plotted as illustrated in Figure **??**

4.3 Literature Review

when talking about general wind prediction or power prediction, we do not need to apply normal model concept. This is the reason I did not do it. "A common approach to fault detection in wind turbines is to estimate an NBM - a regression model to predict the value of a state variable - and to compare the model's prediction with measured values. The NBM is estimated on a training set for which it is known that the turbine is functioning normally. In the language of statistical process control that corresponds to phase I, where the system is supposed to be in control. The difference between the predicted and actually measured value is called residual and the residuals are used for monitoring the turbine's condition - which corresponds to phase II in SPC. Residuals that exceed the a given error threshold - called control limit - indicate malfunctioning. Te residuals indicate how close the wind turbine's behaviour is close to the one observed during training of the NBM. It is reasonable to assume that in a normally functioning wind turbine, the residuals are homogeneously randomly distributed. Therefore, trends or excessively large residuals indicate a departure from normal functioning and an alarm is triggered if ht residuals exceed the control limits. (Helbing & Ritter, 2018)

The focus for this thesis is data-driven approach for yaw fault forecasting using deep learning models. Relevant literature has been researched.

The general conclusion from the point of time series modelling is the promising prospects of employing deep learning models for complex problems involving large scopes of data and multiple variables (Lazzeri, 2020) (Nielsen, 2019). When considering wind turbines monitoring a lot of research has been done using data-driven approaches based on SCADA data and machine learning models.

(Stetco et al., 2019) conducted an overview study of machine learning methods used for condition monitoring of various systems in wind turbines. The review classifies machine learning models based on the source for data, feature selection and extraction methods, model type and validation methods. In total 144 papers have been reviewed, dating from 2011 and onward. The conclusion is that most of the papers rely on SCADA generated data and almost two-thirds use classification models and the rest regression models. Neural networks, support vector machines (SVM) and decision trees are among the most widely used models.

One important observation regarding SCADA data for wind turbines is that there is no common set of features measured by SCADA systems as well as the names for the same signals vary. The regression models covered in the review are applied to model normal behaviour of a wind turbine. A normal behaviour of a system is the one without faults occurring. The idea is that a regression model is fit to the data that has been cleaned of the periods with alarms triggered as well as outliers. Ideally the data should be collected at the periods when a fault is least expected to occur. So for the components that suffer mostly from the wearing out, this data should be collected around the period of the installation.

An anomaly or a fault is registered when the model is predicting new data and the output variable deviates from the reference curve greater than the predefined value. The validation metrics mostly used are mean absolute error (MAE), mean absolute percentage error (MAPE), symmetric mean absolute percentage error (sMAPE), root mean squared error (RMSE) and the coefficient of determination R2.

When looking closer into how yaw system faults have been forecasted the number of research papers is smaller. (B. Chen, Xie, Li, & Gao, 2020) have based Bayesian network onto acoustic signal to predict the yaw system failure. This is based on the fact that the yaw system failure leads to abnormal vibrations in the system. They have also concluded that data-driven methods suffer from the the lack of reliable fault data as wind turbines control systems stop them before the fault actually occurs. Thus the hybrid models based on both mathematical and data-driven models is a solution.

For many studies the lack of alarm log information led to a natural step of building a normal behaviour model by filtering the power curve from the probable outliers (Xiao, Liu, Zhang, & Zhang, 2021).

(H. Chen, Liu, Chu, Liu, & Xue, 2021) compared LSTM, auto-encoder (AE), LSTM + AE and ANN models for anomaly detection in the stator operation based on SCADA data. The combination of LSTM + AE had best results.

(Astolfi, Castellani, Becchetti, Lombardi, & Terzi, 2020) focused on the rotor speed because of its direct correlation with the yaw angle in the region of the power curve when the rated speed is not reached yet. It was concluded that observing the plots of yaw error-rotor speed of a turbine in comparison with the rest of the wind park gives clear indications for fault forecasting.

(Jing, Qian, Pei, Zhang, & Yang, 2020) propose yaw misalignment method based on Maximum Power Capture and simulated data.

(Trizoglou, Liu, & Lin, 2021) have compared XGBoost and LSTM models for generator fault prediction in offshore wind turbines with the first one having better estimates based on oil temperature measurements.

Part III

Methodology

Chapter 5

Dataset background and data extraction for the experiment

This chapter sets the scene for the later section with presenting the background information of the data used for forecasting yaw alarms. The short presentation of the wind park is followed by detailed overview of the dataset's origin, contents and structure. The strategy to choose the representative data for the problem solving is discussed in detail.

5.1 Wind park

The thesis is based on the data from Smøla wind park. This park is located on the island Smøla in Møre and Romsdal. The highest point on the island is 70 meters above the sea level and it is located approximately 12 *km* from the most south edge of the wind park. Thus it have great influence on the wind conditions at the wind park cite. Otherwise the location of the wind farm is described as flat and open landscape with the elevation ranging between 10 and 40 m above the sea level (Statkraft, 2011).

The turbines are placed along the elevations in the landscape. The distances between the turbines in each row varies between 240 to 350 m and the distance between the rows varies between 700 to 1000 m (Statkraft, 2007). The layout of the turbines positioning is presented in Figure 5.1. The the park was installed in two steps: Smøla 1 with 20 turbines in 2002 and Smøla 2 with 48 turbines in 2005. The total installed capacity is 150 *MW*. The annual rated production is 356 *GWh*.

The rated characteristics of the turbines in both parts of the wind park are presented in Table 5.1. The turbines at Smøla 2 are produced by Siemenes and have larger rotor diameter, larger tower height and consequently higher rated power. They are equipped with pitch power control, while Bonus turbines at Smøla 1 are equipped with stall control. Otherwise the wind speed modes and rotor speed characteristics are identical. As stated in the Siemens' catalogue (Siemens AG, 2011) SWT-2.3-82 turbines are suited both for use on-shore and offshore.

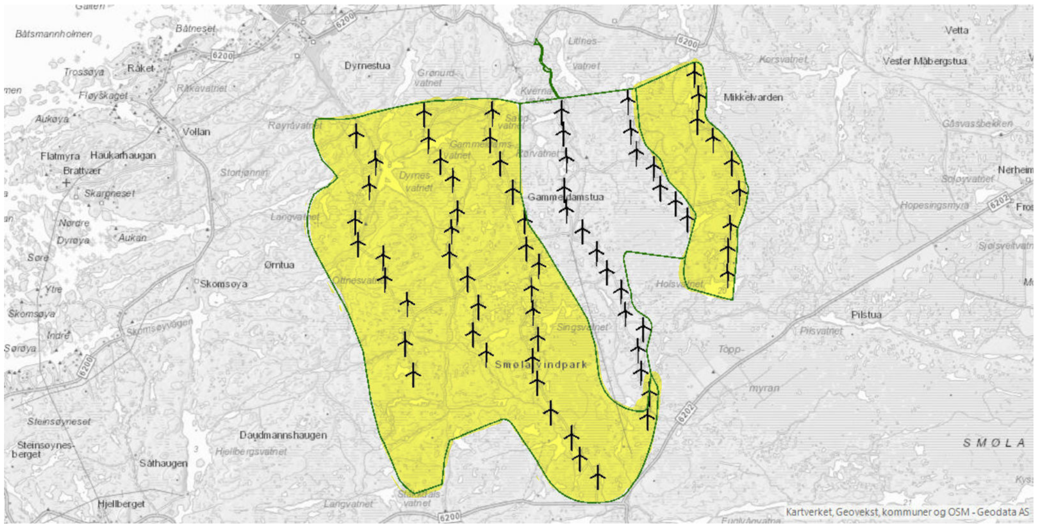


Figure 5.1: The layout of wind park Smøla, where Smøla 2 is identified with yellow colour. Inspired by (NVE, n.d.)

5.2 Dataset

The dataset for the wind park was provided by Statkraft. The original data contains the operational data gathered by SCADA system for the period between March 2007 and January 2018. The data is divided into Smøla 1 and Smøla 2 parts. These are further divided into subsets on a yearly basis with further subdivision into thematic subsets on monthly base. The number of available thematic subsets for Smøla 1 and Smøla 2 varies. Their number also varies over years for the same subpart of the park. The overview of the thematic subsets provided can be studied at Table 5.2. The original abbreviations for the datasets are given in bold. The document with the mapping of the operational data was lacking, thus the interpretation of the abbreviations for the datasets is sporadic and done based on researching the relevant literature. The generalisation of names of thematic subsets and names and number parameters measured for wind parks is complicated due to the lack of common taxonomy of measured parameters, as well as deviations in parameters measured. Insuasty et al. (2021) proposed to categorise measured parameters into the following groups: environmental, electrical, temperature, pressure, hydraulic, and control variables .

Many of the operational parameters are presented as mean, maximum, minimum and standard deviation values averaged over 10 *min* period. The frequency of data points between alarm and event logs and the thematic subsets does not coincide as the first ones have higher frequency of measurements. Thus the alarm data requires synchronisation with the thematic subsets which is described in Subsection 6.4 in Chapter 6. The document with alarms and events description provides information about alarm codes what allows to identify yaw alarms instances in alarm logs.

The weather measurements, such as wind speed and wind direction measured at various heights and temperature, were provided as the meteorological station dataset. This dataset contained

Parameters	Smøla 1	Smøla 2	Total
Number of turbines	20	48	68
Type of turbine	Bonus B76/200	Siemens SWT-2.3-82	
Rated power	2 MW	2,3 MW	
Rotor diameter	76 m	82.4 m	
Number of blades	3	3	
Power control	Stall	Pitch	
Cut-in wind speed	3.5 m/s	3.5 m/s	
Rated wind speed	17 m/s	17 m/s	
Cut-off wind speed	25 m/s	25 m/s	
Maximum rotor speed	17 rd/min	17 rd/min	
Tower height min - max	60-98 m	80-100 m	
Capacity per park	40 MW	110 MW	150 MW
Yearly production	120 GWh	330 GWh	450 Gwh
Area	4,5 km^2	13,5 km^2	18 km^2

Table 5.1: Rated characteristics for Smøla wind park with data presented for each subpart individually (Statkraft, 2007).

data points for 2017 only. The technical characteristics of each wind turbine were provided such as power reference curve values and the transfer function are also part of the dataset.

5.2.1 Choice of experiment dataset

Due to the lack of domain knowledge about correlation of yaw faults and other parameters, the strategy for improving the results by Tallaksrud (2021) relies on the idea of expanding the size of dataset. This can be achieved by increasing either the number of parameters, i.e. number of thematic datasets also, or the number of data points over time. The third option by combining data from several turbines in one dataset was declined since it is difficult to fulfil. The idea of employing a larger dataset is supported by the fact that deep learning models can handle big data data and catch hidden patterns which are not evident to the human eye (Walter et al., 2021).

In regards to the time horizon for the dataset the original plan was to use data for the period of several years. Due to the complexity of the data preparation process this strategy was abandoned. Some of the obstacles were the size of concatenated data and various patterns for missing values in datasets that required more advanced data cleaning. Moreover, the choice of the period was dictated by the initial presumption regarding the weather data. Since a wind turbine extracts energy

Data abbreviation/name	Smøla 1	Smøla 2
CNT Control	+	only 2007, 2017
FLG Flags	+	+
GRD Grid	+	+
Scd	+	+
Alarm log	+	+
Daily Summary	+	+
TMP Temperature	+	+
TUR Turbine	+	+
DIN	-	+
DOT	-	+
INT Internet	-	+
PRK Park	-	+
PRS Pressure	-	+
STD	-	not all months
SCWps	-	+
MET	-	01.2017

Table 5.2: Comparison of the data provided for Smøla 1 and Smøla 2

from the wind and the intermittent nature of the wind influences the yaw system operation it is an obvious choice to focus on wind measurements. The presumption was only the measurements from the weather station can be used. Such weather data for Smøla contains data points for 2017 only. Consequently, 2017 was chosen as the period for the experiment data. Although later it was discovered that in accordance with the IEC 61400-12-2 standard (ISO Central Secretary, 2013) the assessment of a wind turbine operation can be conducted based on the wind measurements at the nacelle provided correction by the transfer function. Though it is importnat to point out that the wind direction parameter is present in the thematic subset **MET** only. Since this subset covers the period for January, 2017 only, it means that the working dataset does not contain the wind direction parameter. It was decided to continue as it is since the parameters for the nacelle and yaw positions reflect the changes in the wind direction.

Since the time period for the experiment dataset is restricted to one year, the only option left is to expand the size via the maximum possible number of parameters and measurements within one year. As the choice was made to employ data from one turbine it was necessary to choose a representative subpart of the park first and a turbine afterwards.

The subpart of the wind park was chosen based on the assessment of the maximum available number of parameters and the number of measurements in the subsets. First of all, the original number of subsets was reduced as not all of them contained measurements on an individual turbine level. As the result the number of subsets for Smøla 2 is reduced to the following selection: **CNT, DIN, DOT, GRD, INT, PRS, SCWps, SCD, TMP, TUR**. Secondly, since Tallaksrud (2021) achieved better forecasting results with **TUR** data, this subset was chosen for calculating and analysing the number of available parameters and measurements over years. The time did not permit to conduct such analysis for all ten types of dataset. The results for this analysis are presented in Figure 7.2 and Figure 7.1 in Section 7.1.

After careful analysis of the provided data, Smøla 2 was chosen as the one with the largest number of thematic subsets and parameters. This choice can be also supported by the fact that the wind turbines by Siemens are more advanced and the research based on them has more relevance for the currently available designs.

5.2.2 Choice of a representative wind turbine

Next step is to choose a representative turbine. The representative turbine for the whole park is chosen based on average value of active power for the park. The averaged estimate was arrived at by summing accumulated active power of individual turbines over this period and dividing it by the number of the turbines. A turbine that has the values closest to the averaged estimate of the park is chosen as the representative turbine. As the number of measurements of active power differs per each turbine due to missing data points, the following steps for data synchronisations were applied:

- concatenate active power measurements for all turbines for all years in one file;
- drop the rows were all the columns, i.e. all the turbines, have missing values;
- investigate the autocorrelation statistics for active power sequences for all turbines and set a fixed value for maximum values for interpolation;
- interpolate with this limit and set the rest values to 0;
- keep the negative values as according to (Sahoo, Routray, & Rout, 2019) negative active power implies a wind turbine being in the mode of idle or stalled mode when it requires power from the grid to run some of its functions, e.g control systems or lightning;
- the sum of active power measurements over the whole period is calculated per each turbine;
- the total active power for the whole park for all years is calculated and averaged over the park;
- the turbine with the value closest to the resulting value is chosen as the representative one.

The result of the top 5 turbines which can be the representative turbine is presented in the table 7.1 in Section 7.1.

The second decisive factor for choosing of the representative turbine was the occurrence of the yaw alarms. This parameter needs to be taken into the consideration as the yaw faults belong to the group of extreme rare events. Extreme rare evens are defined so with the occurrence of one class less than 1 % of all labels. Which is the case for yaw labels versus non-yaw labels based on SCADA analysis. The governing principle is the same as in the case with active power: the representative turbines should have the number of yaw alarms per year close to the average number of alarms per park for this year. Figure 7.1 illustrates the comparison of the top five turbines from the step one in regards to the fault alarms statistics. Finally as the data set is chosen, the next step is to perform data wrangling or preprocessing which is discussed in the next chapter.

Chapter 6

Data preprocessing

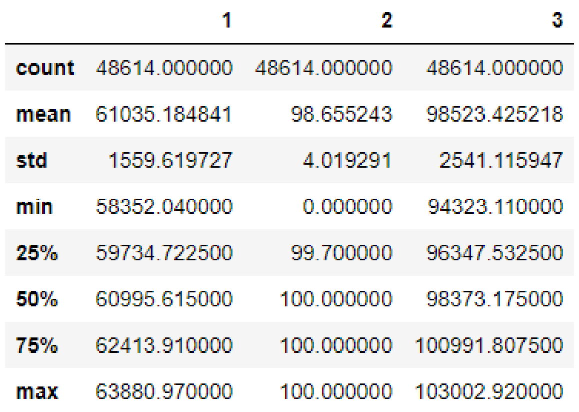
This chapter is dedicated to various data wrangling or preprocessing techniques used under the experiment. Some of them are required as part of general machine learning workflow, e.g data cleaning, feature selection and engineering, handling missing data, synchronisation of data and formulation of the problem in terms of machine learning. The other part preprocessing techniques is determined by LSTM model such as choice of the model architecture, data transformation for both features and targets. In addition the special aspects of applying regular ML methods to time series data are presented. The chapter is concluded with the details of tuning and evaluating methods.

6.1 Data cleaning

The initial data cleaning included simply the removal of the duplicate columns and rows. The first one arose due to the fact that the data from the first day of the next month is present in the previous one. The duplicated columns contained the identification for a turbine. The columns containing data of particular type were removed such as timestamps outside of the index and categorical data . The encoding of the categorical variables was considered, but the features with "*", "=" as entries were considered to have little informative value.

6.2 Outliers

Outliers identification can be performed either manually or applying automated methods. Both of the methods rely on statistical learning. The descriptive statistical analysis of the dataset can be acquired with the pandas function *df.describe*. The example of an output is presented in Figure 6.1. It provides the count of data points, the minimum, maximum and mean values of the parameter as well as the percentile distribution. As it can be observed for the column 2 in the dataframe, the median value or 50 % percentile is equal to 100, while 25 % of the values lie below 99.7 and the minimum value is 0.



	1	2	3
count	48614.000000	48614.000000	48614.000000
mean	61035.184841	98.655243	98523.425218
std	1559.619727	4.019291	2541.115947
min	58352.040000	0.000000	94323.110000
25%	59734.722500	99.700000	96347.532500
50%	60995.615000	100.000000	98373.175000
75%	62413.910000	100.000000	100991.807500
max	63880.970000	100.000000	103002.920000

Figure 6.1: An example of a method describe from pandas of a dataframe with 3 features.

The same information can be illustrated with a boxplot as in Figure 6.3 or Figure 6.3b. A boxplot is a graphical method in descriptive statistics reflecting the data distribution of a parameter through quartiles. As from the describe method, the box plot suggests that the values equal to 0 are outliers. The same can be said about the values below 100 which are marked as thick black line in Figure 6.3. Though without knowing the background or the domain knowledge for the parameter it is impossible to state this with confidence. These values can be seldomly occurring and within the parameter values range. Another reason not to conduct the outlier analysis is the large number of the parameters in the dataset.

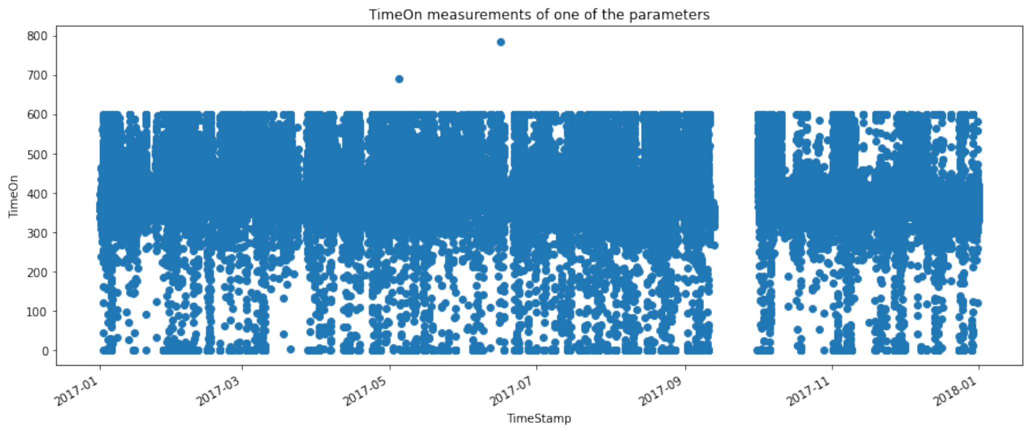
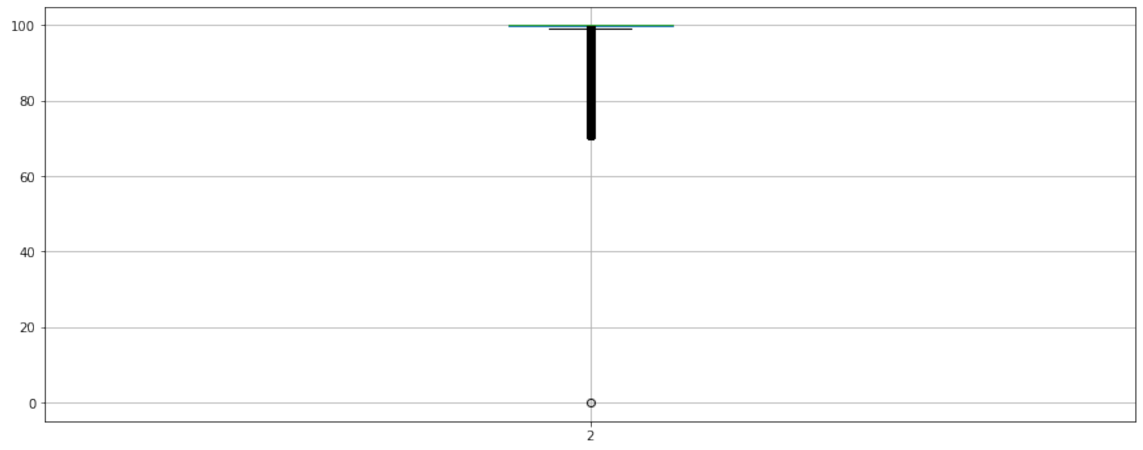
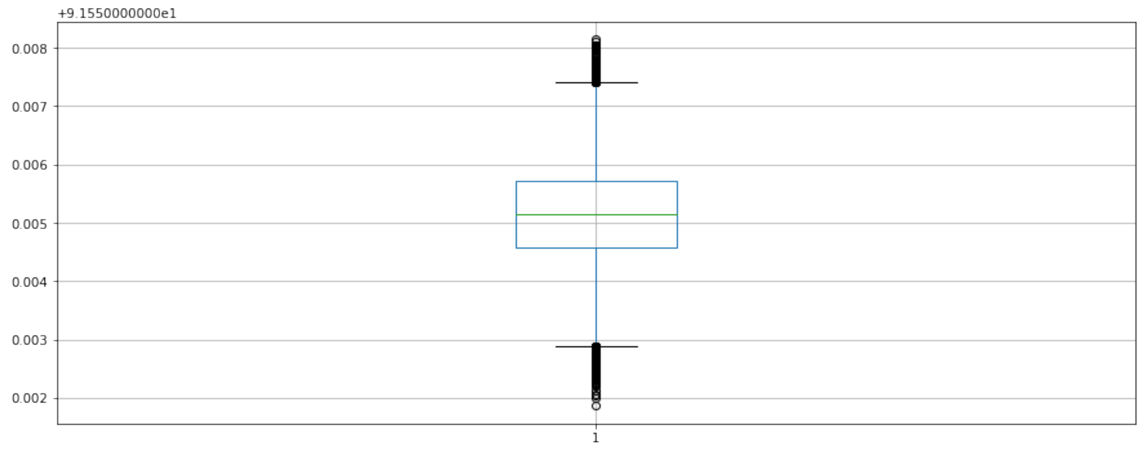


Figure 6.2: An example of outliers detected during a plot analysis of one of the parameters. Data points that lie above 600 are considered to be outliers.

Both the descriptive analysis method, boxplot and simply plotting of the variable as in Figure 6.2 were applied for the shallow analysis of outliers.



(a) An example of probable outliers with the value 0



(b) An example of probable outliers outside the upper and lower extremes.

Figure 6.3: Two examples of boxplot applied for analysis of outliers

6.3 Imputation

From a list of imputation methods for sequential time series specified in Section 4.1, the following ones were tested on with the dataset: forward filling, backward filling, KNN, moving average and linear interpolation.

- *Forward filling* is an imputation method that propagates last valid observation forward into the missing values positions. While backward filling is an imputation method that propagates the first valid observation after the missing values into their positions. It is possible to specify the maximum allowed consequent missing values to impute for both methods. It was set to no limit as the size of the final dataset does not allow to perform autocorrelation analysis of each variable in order to identify the imputation period. ¨

- *KNN* imputation method relies on the principles of k-nearest neighbours algorithm and fills in missing values with the mean value from the n-nearest neighbours. Here 5 nearest neighbours were used for data imputation.
- *Moving average* method relies on the idea that time series data does not have the constant values for mean values across the whole period of measurement as it happens with a non-time series data. The method can be implemented using a rolling mean method by pandas where the period for calculating a mean value is specified. Here 24 hours where specified as the period for averaging across. Since 1 hour contain 6 measurements, the period is specified as 144.
- *Interpolation* method in pandas package is class of imputation that can be conducted as linear, time based, nearest, padding, index-based, spline and polynomial interpolation. Time-based interpolation was applied in this experiment. This method allows to impute missing values for data with daily or higher resolution. The default method is forward and no limits for consequent values to impute. Time-based interpolation is a linear interpolation with respect to time steps. A regular linear considers all the data points equally spaced. Time based was applied as there are long periods of missing data in September which makes missing values patter not equally spread across data.

An example of applying KNN imputation method to one of the variables is illustrated in Figure 6.4.Here it can be observed that 13 points were imputed. The rest of the results are presented in Section 7.6.

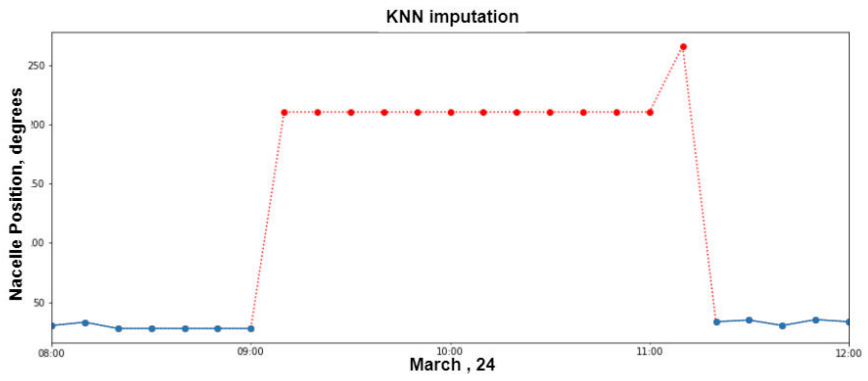


Figure 6.4: An example of imputation with KNN method applied to Nacelle position feature.

6.4 Synchronisation

Since the data in the various thematic datasets is collected separately from different subsystem, the differences in the number of data points collected are frequently observed. In order to be able to work with the data, these datasets need to be synchronised. It is performed via concatenating datasets together and re-indexing the whole dataset. As the result the features that lack some timestamps have more missing values. This concatenated dataset becomes X-data that is used as the experiment dataset or base for further imputation to acquire imputed X-train dataset.

The yaw alarms log requires also synchronisation with X-data as the frequencies of data measurements for the thematic datasets and the alarms log differ. The possible choices are either to upsample the thematic datasets to have one measurement at every second or to downsample the alarm log dataset to have a single data point at every 10 minutes.

It was chosen to downsample alarm dataset as it interferes less with the structure of the data. All the timestamps for yaw alarms are rounded up to the next 10 *min* interval. It means that if an a event occurred at 10:46:10, the new timestamp becomes 10:50:00.

6.5 Alarms log filtering

As the alarm logs contain all faults, the yaw alarms were filtered and labelled with 1, while the rest of the alarms were set to 0. Despite the scarcity of the yaw faults, there has been registered a single instance of two yaw alarms occurring at the same 10 minutes interval after the data was downsampled. As it has been the same the same yaw alarm triggered within a short time interval - about 2 minutes, the second alarm was dropped. Later the index for the yaw data which represents target was synchronised with the timestamp index from the parameters dataset. It was of importance not to loose yaw alarms instances due to the presence of missing values in the parameter dataset.

6.6 Feature Engineering

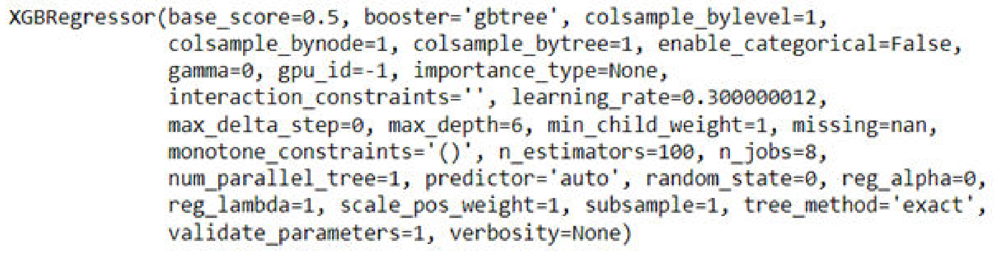
The performance of a machine learning model depends greatly on the data it is fitted onto. The nacelle position and yaw position parameters are measured in the range from 0 to 360 degrees. One more yaw position parameter has a range between 0 and 500 as it is assumed to measure the rotation distance from the starting point. Though acting as the maximum and minimum values of the features, these values are close to each other. Converting the units from Polar to Cartesian coordinates and combining them with the wind speed measurements allows to present these data in a better format for the model (TensorFlow, 2022b). The result of the transformation can be observed as the changes in the distribution of the wind direction and nacelle and yaw positions as presented in Figure 7.13a in 7.9.

6.7 Feature Selection

Feature selection was conducted applying Extreme Gradient Boosting (XGBoost) which is an implementation of gradient boosting that can be directly used for regression problems (Machine Learning Mastery, 2021). Gradient boosting belongs to the class of ensemble machine learning algorithms. Ensemble machine learning relies on combining multiple models, which can be weak on their own, to produce an improved common prediction (Lutins, Evan, 2017).

A XgBoost regressor is fitted to parameters data and the target data and the number of features is determined by the user. Under this research 99 significant features are chosen from the total feature pool equalling 704 to be retained as training dataset.

The imputation with a regression model resembles that of target prediction, that is why the procedure with XGBoost regressor is the same as any other machine learning model. We can tune the parameters of the model in order for it to be able to generalise the information well enough. The default parameters were used under the model fitting and they are illustrated in Figure 6.5.



```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.300000012,
             max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=100, n_jobs=8,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)
```

Figure 6.5: XGBoost default parameters used for feature selection

The ones which are of interest are $booster =' gbtree'$, where gbtree stands for gradient boosting tree. The booster is a leaner and a tree learner is the one that is good for catching non-linear relations in data. The $gamma$ parameter controls the strictness of the model through minimum loss reduction required to continue with next partitioning. And $'learningrate'$ is a measure for preventing overfitting. Finally the size of the model tree is determined by $maxdepth$, $numparalleltree$. The complexity of the model depends on the tree size and how many partitions as branches it has.

6.8 Target transformation

As the original target variable is presented as either ones or zeros, the appropriate choice for the type of deep learning model should have been classification. Thus 1 would define an occurrence of a yaw alarm and 0 - the absence. But due to the fact that the yaw alarms occur very rarely the dataset is heavily class imbalanced. In the case of a cross-sectional dataset it is possible to either upsample the minority class, the yaw alarms occurrence in this case, or to downsample the majority class, the absence of the yaw alarms. The dataset for forecasting though is time series data and the sequence of data points is of crucial importance. Thus the sequence structure prohibits to tackle the class imbalance in the same manner.

But as the goal of this thesis is to attempt in improving the model proposed by Tallaksrud, the target variable was transformed in the same manner from binary variable into continuous variable (Tallaksrud, 2021). The target variable becomes a sequence of increasing probability of a yaw alarm to be triggered. In aforementioned research both the exponential and the linear transformation function of the target variable were tested, with the exponential giving better results. Thus the exponential transformation is chosen for this research.

The proposed time periods for probability distribution of 2 and 10 hours were chosen as well in this experiment. The transformation of the target relies on the idea that the probability of an alarm is manifested in the data measurements and the assumption is that this can be estimated with an exponential probability distribution.The result of the target transformation is illustrated in Figure 6.6.

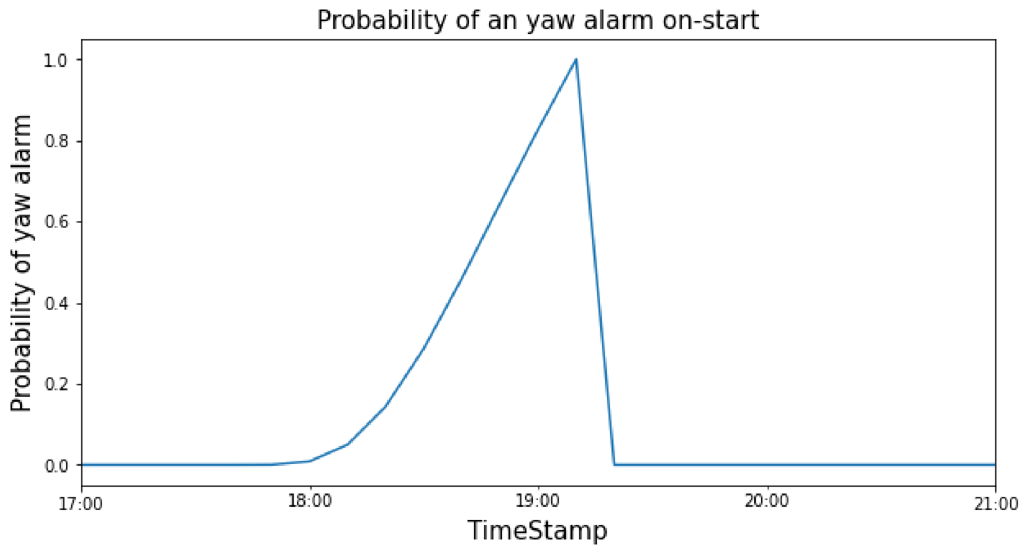


Figure 6.6: An example of transformed target variable presenting the probability of a yaw alarm on-strart.

The process of data preprocessing is not over as several more steps are required before it can be fed to a machine learning model. The order of the steps also matters as the switching some steps can lead to information leakage. The result of the information leakage is overfitting of the model.

6.9 Train test validation split

Firstly, the data needs to be split into training, validation and test sets. The training set is used for learning patterns in data and the corresponding output connected with them. The validation set is required for tuning the model based on its ability to predict output for the new data. The final test set is used for evaluation of the best model after tuning.

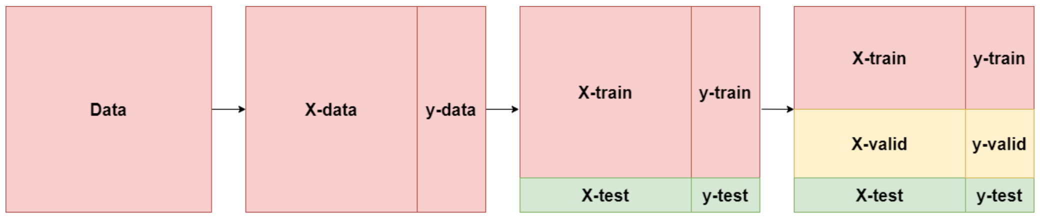
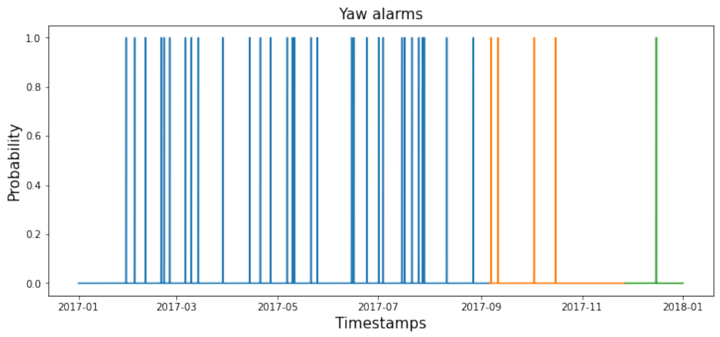


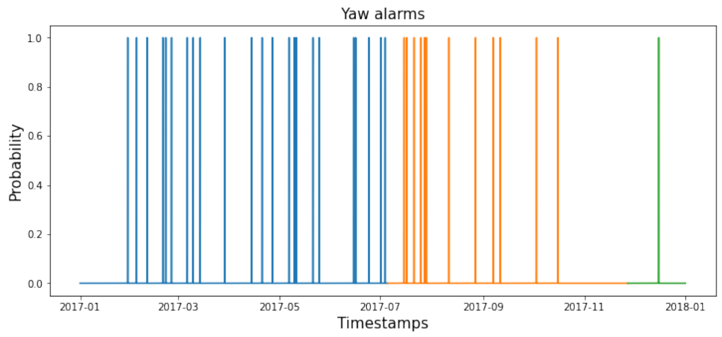
Figure 6.7: Caption

The reasoning behind splitting the data into three datasets is the validation part of the data is reserved for that the model trains on one larger chunk of data and is validated on smaller one with possibility to see the targets. The test set is withhold as completely new data for the model in order to test its power to estimate the target variable. ¨

The size of each set is one of the tuning parameters for the model. The common ratio for the splits is 80 / 20 , i.e. that 80 % of the data is kept for training and 20 % is kept for testing. The further split into training and validation follows the same ration. Thus the total ratio between the sets is defined as 80/20/20. The splitting can be either done manually for time series deciding by time stamps or applying *sklearn* train-test-split function. When applying the later an important parameter in regards to time series data is shuffling of the data. The data can not be shuffled in order to retain the sequence information.



(a) Ratio 80/20/10



]

(b) Ratio 60/40/10

Figure 6.8: Distribution of yaw alarms between train, validation and test sets with 80/20/10 and 60/40/10 ratios.

The guiding idea when choosing the size of each set is the variance of the training set and performance statistics of the validation and test sets is reversely proportional to the split size. It means if the train set is too small, then variance in the data the model is fit onto is to large for it to learn.

For the purpose of this thesis the influence of split ratio on the performance of some of the models was testes.The first set with models was run with 80/20/10. Ratio 60/40/10 was used with some models as an attempt to enhance the performance. The size of data set permits to allocate more data to validation set. The difference in the number of alarms being part of validation set with these ratios is illustrated in Figure 6.8.

6.10 Data Scaling

Deep learning models do not require scaling, but scaling allows for the model to converge faster when the data distribution is closer to normal distribution. As result of scaling the range of the whole dataset is a set from 0 to 1.

Some of the common methods are Standard Scaler, MinMax Scaler and RobustScaler. The RobustScaler was chosen for the initial models as the presence of the outliers still should be accounted for. The process of their removal could not be performed properly due to the size of the data and the lack of domain knowledge.

The mechanics behind the robust scaler include calculation of the meadian or 50th percentile, and 25th and 75th as well. The median is then substracted from each varibale and the result is divided by the interquartile range (IQR) which is the difference between 75th and 25th percentiles as

$$\text{Value} = (\text{Value} - \text{ Median})/(\text{p75} - \text{p25}), \tag{6.1}$$

where p75 and p25 stand for 75th and 25th percentiles correspondingly.

The scaler is first fitted onto the train data and then the scaling model is used for all the three sets. The resulting features have standard deviation of 1 and 0 mean. The change of the data range as the result of MinMaxScaler and RobustScaler is presented in Figure 8.1 in Section 8.1.

6.11 Choice of the model

As the data for this experiment belongs the category of time series the experiment builds around use of recurrent neural network. Section 4.0.3 explains that recurrent neural networks are one of the best networks to be used with sequence data. Though when applying them to long sequences the problem of exploding or vanishing descent arises what implies that model can not learn from the data as the weights are wrongly adjusted. The gradient problem can be tackled with LSTM model.

6.12 Rolling window

The input and output sequence formats used for the problem correspond to the category many-to-one. The input tensor is a sequence, while the output is a single value. The method used for data transformation is sliding window or rolling window.In accordance with the method the data is split into windows of certain length, where every next window sequence starts with the value a step later. For the purpose of this experiment the window length is tested with three values: 2 hours or 12 timestamps, 10 hours or 60 timestamps and 24 hours or 144 timestamps. The step of sliding is 1. If the window is the first t-23 ... t values, the target is t+1. This procedure for a sliding window with the length of 24 hours is illustrated in Figure 6.9.

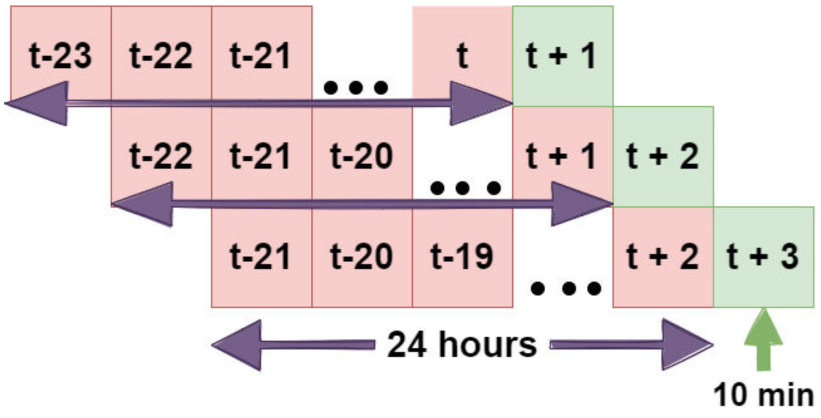


Figure 6.9: An illustration of sliding window method with the length of 24 hours.

The data transformation into window sequences was performed with the help of TimeSeriesGenerator provided by keras (TensorFlow, 2022a). TimeSeriesGenerator belongs to a class of preprocessing methods. The method takes Xdata and ydata as input. Various parameters are applied in order to define the input and the output shapes. The list of all of them is illustrated in Figure **??**. The length controls the number of time steps in the output sequence. The sampling rate and the stride control the period between consequent the input and the output sequences consequently.The start and end indexes help to define which part of the dataset should be used for training and test data. This preprocessing is performed via train-test-valid split by sklearn and is left with default values. The parameter shuffle is set to False to avoid shuffling time series data which will lead to loss of the sequence information and also leads to information leakage. The parameter batch size defines how many time series samples are in each batch. In this research size 32 was used.

Thus TimeSeriesGenerator was used to transform Xdata ann ydata into tensors of shape (32, 144, 704), where 32 stands for the number of samples, 144 for the time steps sequence and 704 for the number of features.

Listing 6.1: TimSeriesGenerator parameters.

```
tf.keras.preprocessing.sequence.TimeseriesGenerator(
    data, targets, length, sampling_rate=1, stride=1, start_index=0, end_index=None,
    shuffle=False, reverse=False, batch_size=128
)
```

The model for the experiment was built as stacked LSTM model, where the number of LSTM layers and their paramters are decided by Hyperban hyperparameter tuner from Keras. The hyperparameters values have direct impact on the model performance. These parameters can be decided via try and error process or automated solutions can be applied.

6.13 LSTM model build

The primary type of LSTM architecture used for this experiment is stacked LSTM what implies several LSTM layers put together. The bare minimum LSTM model contains an input layer and output layer. The concepts of these layers are considered first before looking into the possibilities of layers in-between.

The input layer can take the data as 3D vector or tensor. The example of possible shape of an input layer can is presented in Figure 6.10. There the input tensor has shape of [(None, 144, 702)]. The first value corresponds to the batch-size and is denoted as None because its size is defined during the fitting step of the model. The second value corresponds to the number of time steps. In this example it is equal 144 what corresponds to 24 hours. The third value stands for the number of features equal to 702 in this example.

The output layer is a Dense layer with the output equal to 1. That corresponds to 1 to one time step. This type of RNN model is defined as many - to - one as described in Section 4.0.3. The possible variations that models have been tuned for under this experiment include the choices presented in python code snippet in Listing 6.2

Listing 6.2: Code for building a model with parameters for hyperparameters tuning

```
def build_model(hp):
    model = Sequential()

    model.add(LSTM(hp.Int('input_unit',min_value=32,
                   max_value=512,step=32),
                   return_sequences=True,
                   input_shape=(win_length_1,num_features)))

    for i in range(hp.Int('n_layers', 1, 4)):
        model.add(LSTM(hp.Int(f'lstm_{i}_units',
                       min_value=32,max_value=512,step=32),
                       return_sequences=True))

    model.add(LSTM(hp.Int('layer_2_neurons',min_value=32,
                          max_value=512,step=32)))

    model.add(Dropout(hp.Float('Dropout_rate',min_value=0,
                     max_value=0.5,step=0.1)))

    model.add(Dense(1, activation=hp.Choice('dense_activation',
                   values=['relu', 'sigmoid'],default='relu')))

    model.compile(loss=tf.losses.MeanSquaredError(),
                  optimizer=hp.Choice('optimizer',
                  values = ['adam', 'RMSprop']),
                  metrics = [tf.metrics.MeanAbsoluteError()])

    return model
```

Thus the hyperparameter tuner Hyperband which is discussed in Section 6.14 can train models with possible combinations of the paramters specified in the code. The first added LSTM layer can have number of units from 32 to 512. Next the number of stacked LSTM layers is determined being between 1 to 4 with the same choice of units in each of them. Finally one more LSTM layer is added with the same choice of units tuning. The model layers a model have, the higher is the performance. Although the computational time is also direcly correlated with number of layers. Though it is worth trying since the data set has 702 features and around 48 000 timesteps. As having many layers can lead to model overfitting, i.e. poor generalisation of new data. In order to be able to counteract such behaviour a Dropout layer with the ratio for dropping some information before the output from 0 to 0.5. Finally the output Dense layer can take either ReLu or Sigmoid as activation function. ReLu is preferential for long time series sequences as it helps to handle the vanishing gradient problem as discussed in 4.0.3. ReLu is also aiding in the model's convergence. ReLU is the preferred option for the neural networks problems, but it's output is either 1 or 0, that doesn't correspond to the model's output. While Sigmoid transforms the values to a range between 0 and 1 what fits perfectly for forecasting probability (Wood, 2020).

In the last step for the tuner the optimizer is chosen between Adam and RMSprop. Though being an extension of stochastic gradient descent (SGD) Adam in the contrast to SGD adapts the learning rate for each parameter separately. Adam adpadtes the learning rate based on both the mean (as RMSprop and the uncentered variance (Brownlee, 2021). While Root Mean Squared Propagation (RMSprop) is known to tackle well non-stationary time series data.

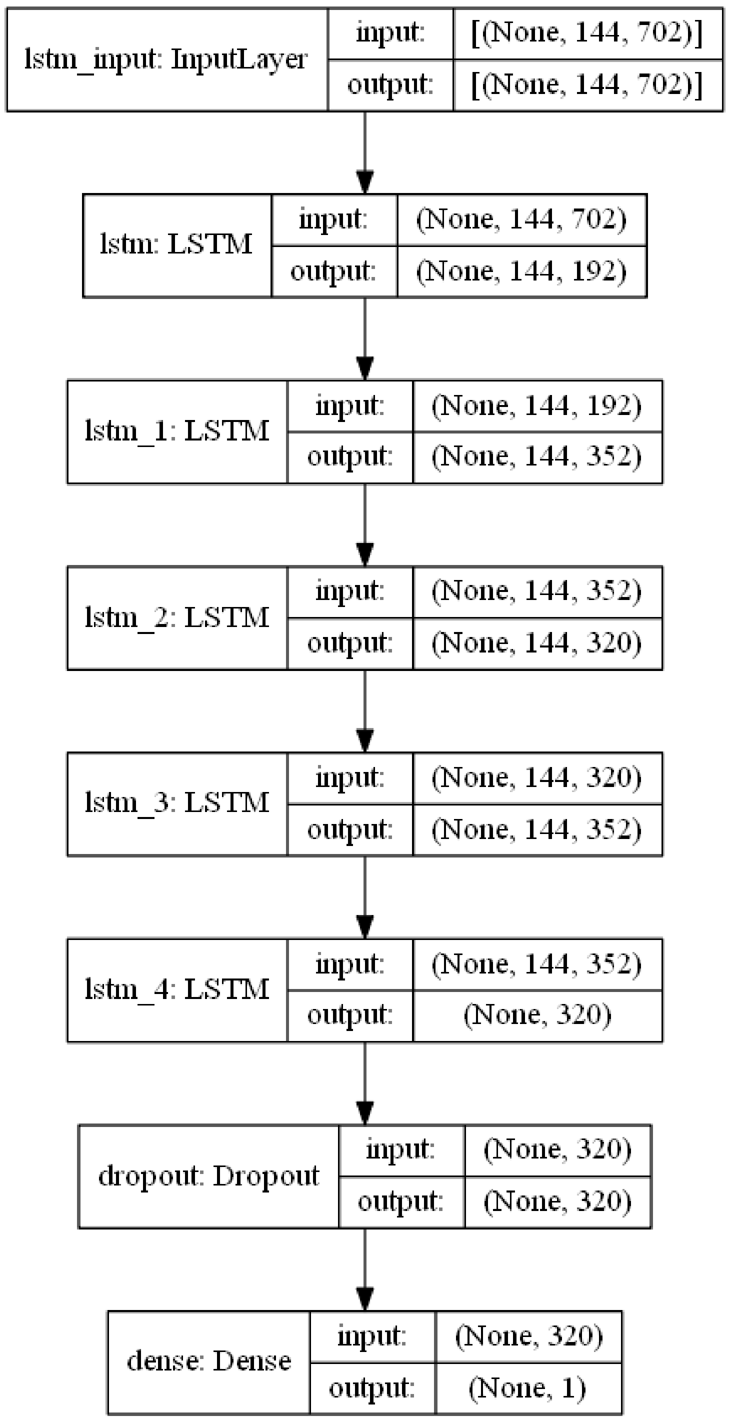


Figure 6.10: The architecture of the model used for round 2 with all models with all features.

6.14 Model tuning

Automated hyperparameter(HP) tuning allows to perform tuning of the hyperparameters in more organised manner. Though the seach space for hyperparameters should be chosen wisely in order to avoid long computation time.

There are three types available by keras tuner by TensorFlow as inbuilt classes:

- BayesianOptimization tuning with Gaussian process.¨
- Random search tuner.
- HyperBand algorithm.

The HyperBand algorithm provided by keras package was applied for models tuning. HyperBand tuning in the contrast to the Bayesian tuning and Random Search, this method allows to speed up the process of parameters optimisation based on the idea that the tuning path that gives bad results should not be followed. Hyperband can be run for a small number of epochs with the possibility to obtain the best parameters for a model.

But there is a bug in the current implementation with tf.keras tuner Hyperband which prevents the tuning process from stopping after defined number of epochs for tuning (Bonometti, 2020) (Mikulski, 2019). The HP search was stopped manually after approximately 10 epochs provided the the search objective value (validation mean absolute error) had not decreased during last 2 epochs.

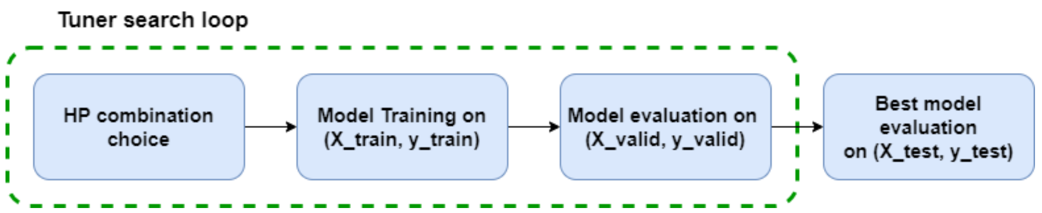
The parameters for Hyperband search are demonstrated in Listing 6.3. The first stands for the function for building a model with defined HP search domain. The objective parameter is to choose which evaluation metric the Hyperband should govern it's search. The number of epochs is defined by the parameter max-epochs. The last two parameters are used for saving the checkpoints from the search. It allows to resume the lengthy process of HP search without information loss.

Listing 6.3: Paramters for Hyperband tuner

```
Hyper_tuner = Hyperband(
    build_model,
    objective='val_mean_absolute_error',
    max_epochs = 10,
    directory = 'keras_tuner_dir',
    project_name='keras_tuner-2-10-20-imp'
)
```

The hyperparameters tuning space used in the experiment is demonstrated in Listing 6.2 in Section 6.13. The possible choices for tuning are discussed in the same section.

The whole process of hypertuning is demonstrated in Figure 6.11. With HP combination choice defined, the HP tuner is fit onto training data with the possibility to verify its performance on the validation data. The values for hyperparamers that has the least loss values, are used to build a model which is further used to predict test values.


Tuner search loop
HP combination choice
Model Training on (X_train, y_train)
Model evaluation on (X_valid, y_valid)
Best model evaluation on (X_test, y_test)
Figure 6.11: The flow of the tuning process with keras tuning functions. Inspired by (Prost, 2020).

6.15 Metrics

The common evaluation metrics for regression problems are Root Mean Squared Error (RMSE), Mean Squared Error (MSE), Mean Absolute Error (MAE). The purpose of the evaluation metrics function is to give a numerical estimate on how well the model predicts target values.

RMSE is an extension of the mean squared error (MSE) and measures the standard deviation of residuals. Residuals or forecast errors are calculated as

$$\text{Residual = Actual value - Predicted value.} \tag{6.2}$$

Thus the expression for RMSE becomes

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2}, \tag{6.3}$$

where y_i is the actual value of i-th target instance and $\hat{y}_i$ is the predicted value of the same instance. The ideal value of RMSE is equal to 0.0, i.e. perfect match, which is hardly possible in real world. A good RMSE depends on the particular dataset estimated. The benefit of RMSE is the ease of interpretation of the results due to same units a the input. The negative side is its sensitivity to outliers.

MAE is a measure of the avergae magnitude of residuals and can be calculated as

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}, \tag{6.4}$$

where y_i and $\hat{y}$ are the actual and predicted values correspondingly.

MAE is an absolute measurement of the residual and it can take only positive values, while the residual itself can be both negative and positive. As well as RMSE, MAE has the same units as the input, but instead of punishing only large errors, it handles all errors with the same weight. As a result the MAE scores increase directly with the value of the residual. MAE is more robust to outliers.

The performance metrics are measured for every epoch run and later can be extracted to be presented in the table and as learning curves. The optimal goal is that the train loss and validation loss should come as close to each other as possible while both in the proximity of 0.This is the indication of model converging. An additional criteria to observe is how fast a model converges as the quicker the better. Example of god fit model learning curve is presented in Figure 6.12.

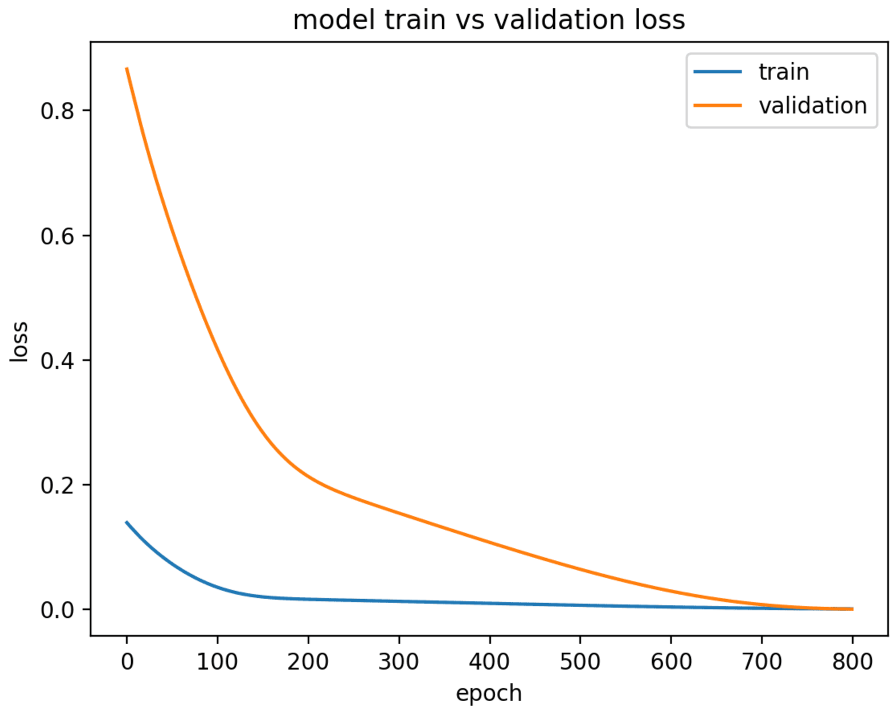


Figure 6.12: An example of a diagnostic plot for a model with god fit (Brownlee, 2020).

6.16 Tools and packages used

Python was used for all conducted data analysis. exploratory data analysis, data preprocessing, statistical methods as the model building, tuning and evaluation were carried out with the following packages: pandas, seaborn, missingno, windrose.

Machine learning and deep machine learning methods were imported from sklearn-library and Tensorflow and Keras packages. Additionally XGBoost library was used for feature selection.

Other libraries such as pandas, numpy, and scipy, were also used for data manipulation, calculation, and analysis.

6.17 Work flow

The experiment was conducted in order to test the following aspects:

1. How the number and the choice of features used in the dataset influence the performance of the model.

2. How the length of the time window influences the performance of the model.

3. How handling the missing values - via dropping or imputing - influences the model.

4. How the train-validation-test split influences the performance of some of the models.

This algorithm was arrived at first but is a result of long process of trying to find the other parameters that can change the performance. The experiment started with tuning 8 models with all features and the choice for the window size of 2h or 10h and the target with the probability distribution for 2h or 10 h as well. In addition the hypothesis whether the imputation can prove the results was tested. But since none of these models did not converge, the the next step was to test feature selection methods and feature engineering in order to test if changing this condition can improve the results. As this did not produce any good results either the sets with random features were tested with the otherwise the same choices, i.e. still 8 models. In the end as the results of these models were unsatisfactory, the idea of making the window size equal to 24 hours is tested with all others possible configurations.

The results of the first stage where all the models were tried randomly were not regarded properly for presentation but they will be still referred too as the model architectures there might be different. As for the final stage one common random stacked LSTM model presented in Figure 6.10 in Section 6.13. Thus the performance of a random model with the result of hypertuning will be discussed.

Part IV

Results and Discussion

Chapter 7

Results and discussion for preparation stage

This chapter contains the results of the exploratory data analysis as part of data preparation for deep learning models. This chapter opens with presenting the background information of the dataset such as the origin, contents and structure. To begin with, the observations supporting the choice of the representative part of the wind park Smøla are presented. This is followed by the results of data extraction for the experiment dataset and finalised with the choice of the representative wind turbine.

7.1 Representative park and turbine

Every machine learning model makes its predictions relying on the patterns it has learnt from data. Thus it is important to present model with properly prepared data so that it is able to extract the information relevant to the problem. The preparation of such prepared data begins with choice of representative park and turbine. For this purpose the file and data contents for Smøla 1 and Smøla 2 are compared. The strategy for the procedure is presented in Subsection 5.2.1 in Part III.

The results of the number of TUR-parameters and the number of TUR-measurements per month for each part of Smøla wind park are presented visually as plots. These are presented in Figure 7.2b and Figure 7.2a. Each column in the plots represents monthly data from TUR-dataset per subpart of the wind park. It is possible to compare the number of parameters for datasets directly. The number of TUR-parameters for Smøla 1 is one third of that for Smøla 2 with 32 parameters for the period 01/2007 - 04/2010 and drops to 7 for the rest of the period. Thus it is clear that based on the criteria of the number of the parameters Smøla 2 should be the representative turbine.

In order to compare the number of measurements per subpart of the wind park further data manipulation is required due to the fact that the number of the wind turbines differs between Smøla 1 and Smøla 2. By averaging the total values by the number of the turbines in each subpart the following values for the number of measurements are acquired:

- Smøla 1 : approximately 4 300 measurements per month per turbine;
- Smøla 2 : approximately 4 100 measurements per month per turbine.

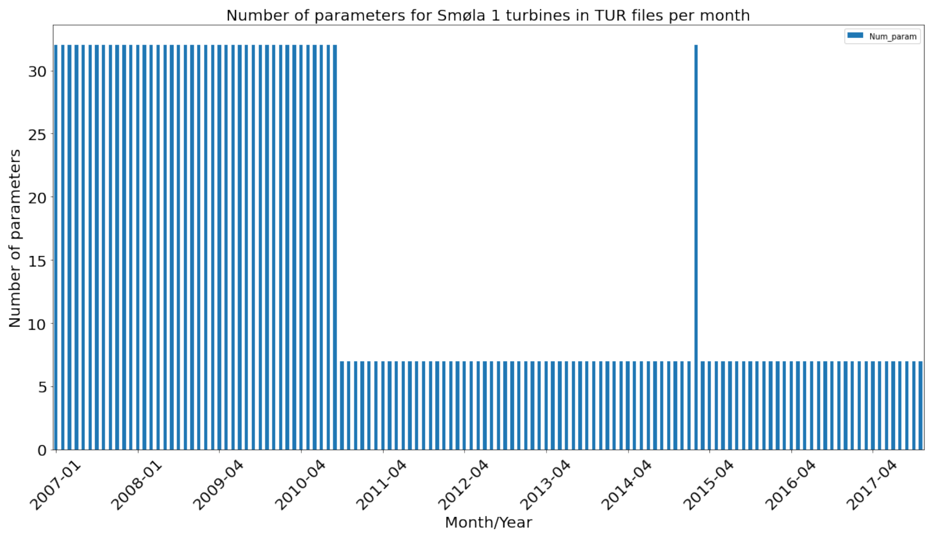
Thus the average number of the TUR-measurements for both subpart is approximately equal with the difference corresponding to one day and a half of measurements. Though it is important to notice that Smøla 2 has several months with long periods of missing measurements. Presence of long sequences of missing data can complicate the forecasting process as the sequence is important. By thus criteria both parks could be chosen as the representative one.

The graphical visualisation makes it easy to observe that the maximum number of parameters for Smøla 1 is one third of that for Smøla 2. That number decreases to slightly above five parameters after several years. The trend in the number of the parameters for Smøla 2 is the opposite. It grew from around 81 to 100. The analysis of number of measurements in TUR datasets per month for Smøla 1 and Smøla 2 shows that the datasets for Smøla 1 has on average about 80 000 observations per month. That means that the number of observations per turbine per month is about 4 000. The same parameters for Smøla 2 equals to about 200 000 and 4 166. Ideally it should be around 4300 - 4400 measurements per turbine.

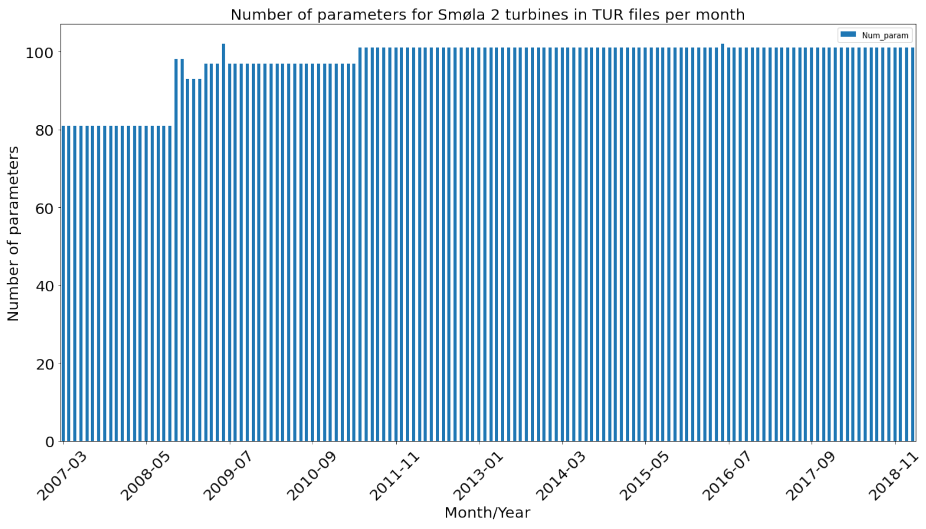
Relying on these two observations and the fact that Smøla 2 has newer model of wind turbines installed Smøla 2 set was chosen as the working dataset.

The time did not permit in-depth exploratory analysis of all features and only the few were analysed. As the power output is the major wind turbine performance indicator it was chosen as the governing feature for choosing the representative turbine for the analysis. Following the procedure described in 5.2.2, the top five turbines in Smøla 2 were chosen as the candidates for a representative turbine and presented in Table 7.1. Combining it with the analysis of yaw failures occurrence in a plot in Figure 7.3 allowed to observe which turbine's stats is closed to the annual average per park. T67 was chosen as the representative turbine.

The analysis of both parameters are performed for the whole period 2007 - 2018 as it was planned to collect data over longer period for the working dataset.



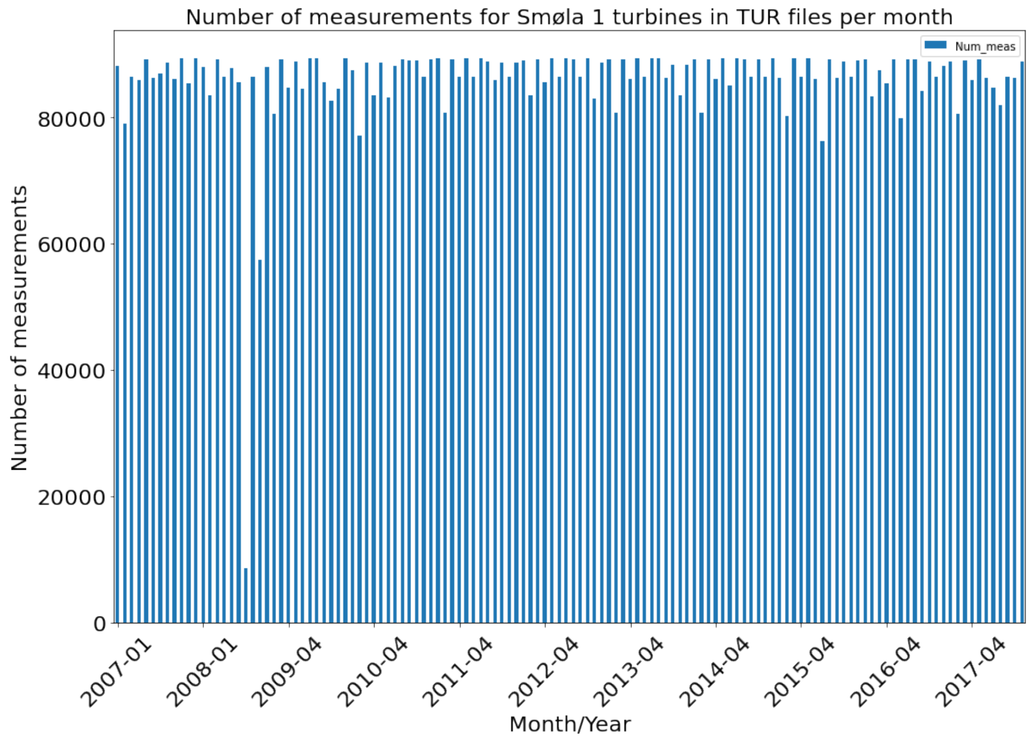
(a) Results for Smøla 1



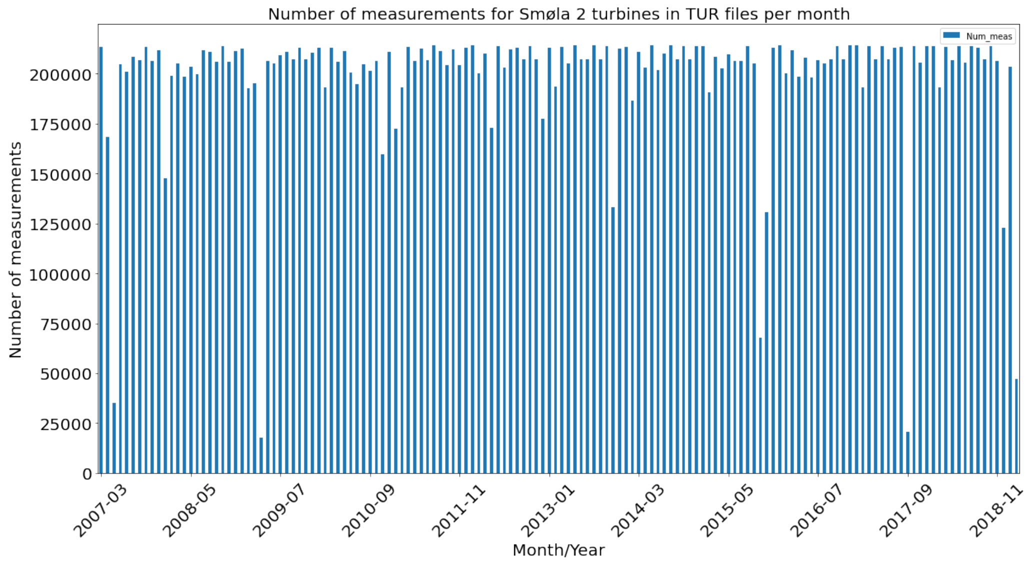
]

(b) Results for Smøla 2

Figure 7.1: Plots with the number of TUR-parameters per month for each part of Smøla wind park for the period 2007 - 2017.



(a) Results for Smøla 1



(b) Results for Smøla 2

Figure 7.2: Plots with the number of TUR-measurements per month for each part of Smøla wind park for the period 2007 - 2017.

Turbine	Annual Active Power, W	Difference, W	Percentage difference
T59	$3.314 * 10^8$	$5.609*10^5$	0.169533
T21	$3.315 * 10^8$	$6.248 * 10^5$	0.188842
T67	$3.315 * 10^8$	$6.606 * 10^5$	0.199673
T52	$3.322 * 10^8$	$1.345 * 10^6$	0.406409
T64	$3.324 * 10^8$	$1.536 * 10^6$	0.464404

Table 7.1: The top five turbines with the average active power closest to the the averaged active power across the park for the period 2007 - 2017.

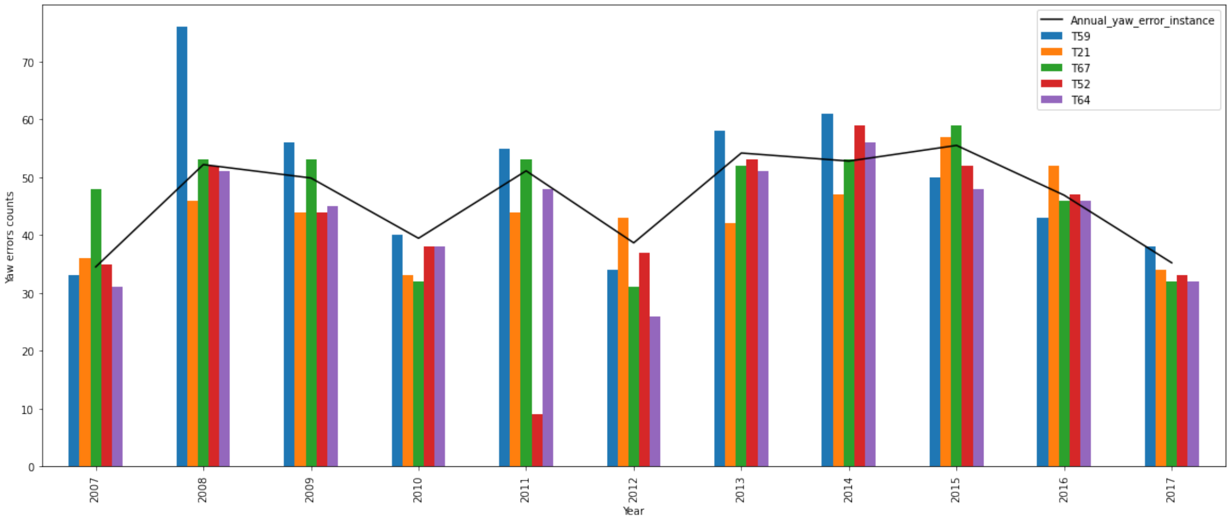


Figure 7.3: Comparison of yaw errors per five turbines chosen by the average active power

7.2 Weather observations

As part of general exploration of the data the analysis of wind measurements has been conducted with both simple wind speed vs time plot and also the wind rose.

The plot of wind speed data collected at the meteorological station is presented in Figure 7.4. Although this data is already smoothed over 10 minutes intervals the detection of the trend an seasonality in data is rather complicated to be performed. The general observations are that the period from mid-October to end of April has in general higher wind speed. This is as it was expected as the winter season is when wind turbines generate most power. This period is though also more prone to have more alarms being triggered as the load on turbine is larger.

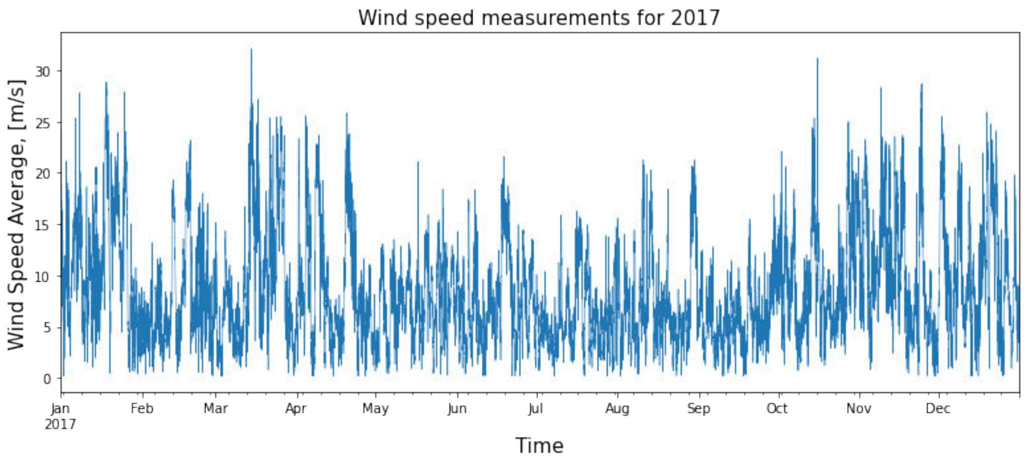


Figure 7.4: Wind speed measurements gathered at the meteorological station at 2017.

The wind rose plot for January, 2017 is presented in Figure A.1 in Section A in Appendix due to its size. This plot presents the wind speed measurements as spikes coming from the centre pointing towards the wind direction connected with them. Since this plot presents the measurements only for one month it is not representative for the experimental dataset. The wind direction is predominantly west, south-west and south with as 180 °represents West and 270 °represents South. The general wind speeds vary from 5.1 m/s to 9.9 - 14.7 m/s. This corresponds with the average wind speed at the coastal area of Norway discussed in Section 2.1.1. The statistics of the weather data can be although not representative for a single turbine as the meteorological station is placed at quite some distance and the local wind conditions would differ at each turbine due to the neighbouring turbine's wake as described in Section 2.2.3 and the surface profile as explained in Section 2.1.

7.3 Time series data components

As it was mention above in Section 7.2 the analysis of wind speed trend or seasonality is difficult to perform relying on simple plot like that. Then the statistical analysis of time series data and the decomposition of this data into components allows to see these patterns much easier. The decomposition plots of wind speed and active power for the whole period between 2007 and 2017 are presented in this chapter in Figure 7.5 and 7.6. The similar plots just for 2017 are presented in Section A in Appendix. The reason for providing both of them is the comparison of how the seasonality and trend components are hard to observe in a yearly data and how apparent they are in the data over several years. This supports the initial plan to use data for several years for the experimental dataset.Thus it will be possible to test the hypothesis if the present seasonality as well as the trend can help to forecast yaw alarm on-start better. For this it might be required to extract these components and perform feature engineering in order to present this information in more easily perceivable way for a model as it is proposed in Section 3.0.2.

Exploring the decomposition plots for active power and the wind speed for the whole period a range of observations is made:

- even though the actual plot of wind speed and active power differ, the trend plots tend to have similarities in general, e.g. the period between the mid-2010 to the end. There is only one spike present in the active power trend plot which is not present in the wind speed plot in approximately the end of 2008.
- the seasonal component plots occur at the same frequency and follow the same shape except for the more jagged top for the active power.
- the residual plots with the unexplained data seem to have relatively the same shape with the active power plot having a wider range. This is consistent with the theory that active power generation depends not only on the wind speed.

The improvement on this analysis would be to generate the decomposition plots for wind direction if the data was available over several years. Similarly the decomposition analysis for yaw alarms occurrence can unravel hidden patterns.

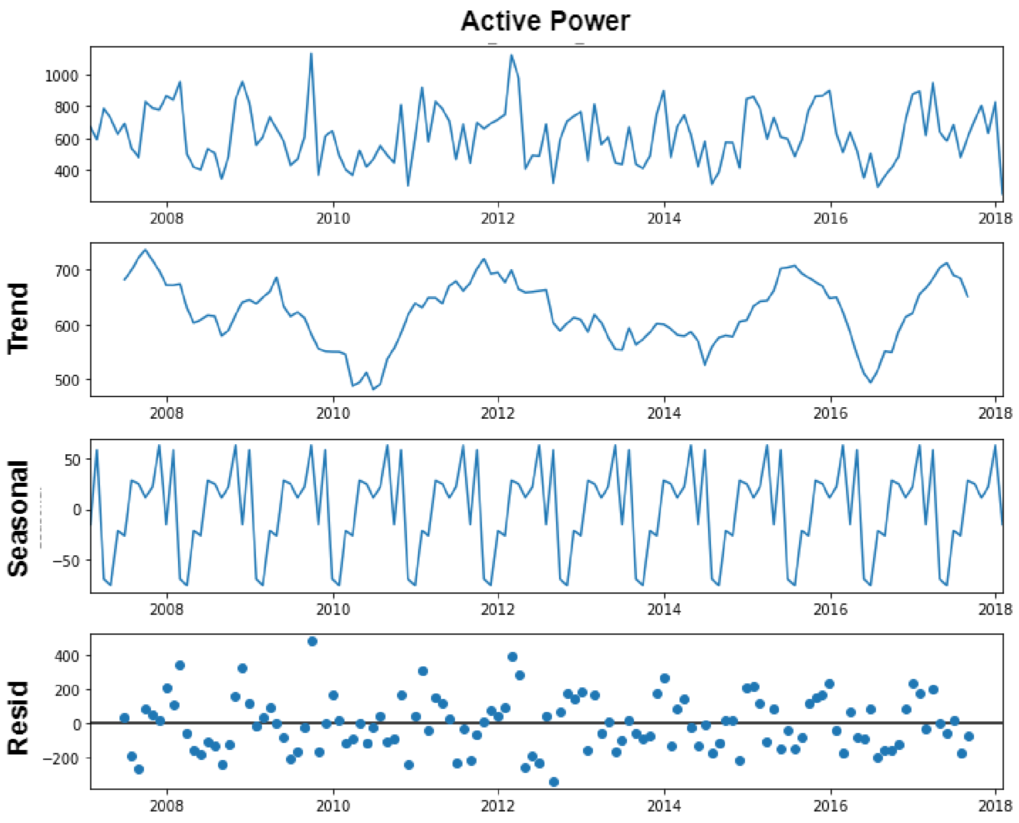


Figure 7.5: The decomposition plot of Active power for all years with seasonality, trend and residuals plotted separately

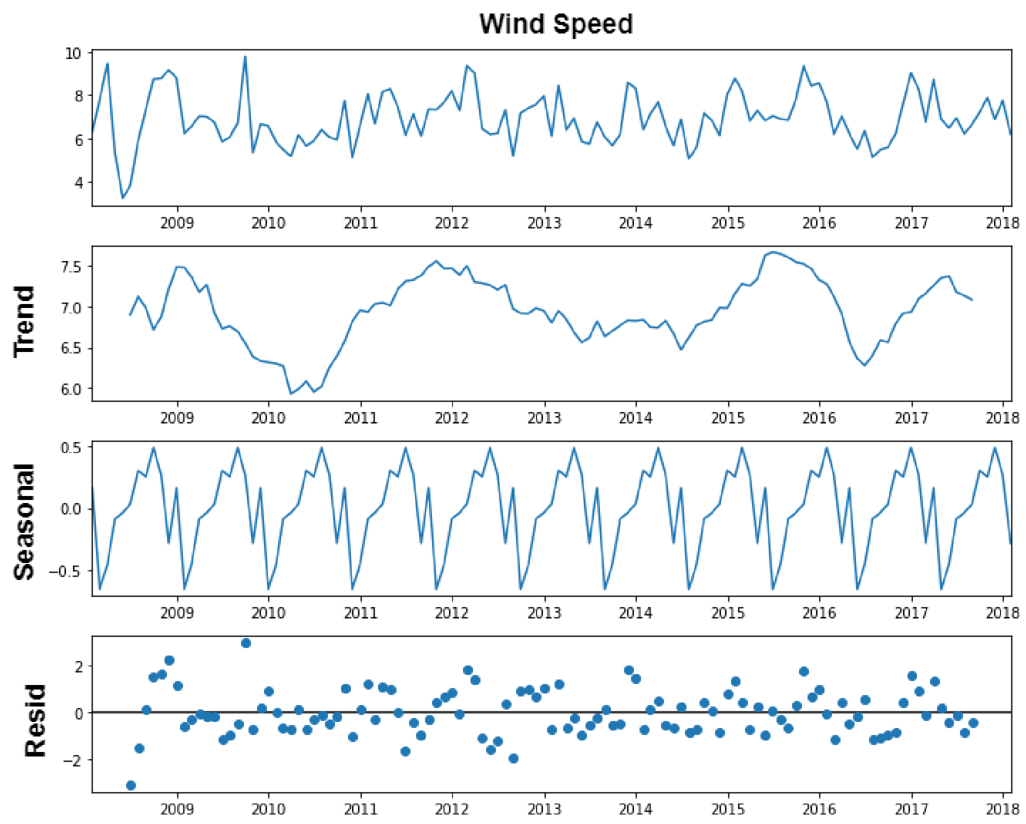


Figure 7.6: The decomposition plot of Wind speed for all years with seasonality, trend and residuals plotted separately

7.4 Synthetic parameter

As it was mention in Section 5.2.1 the wind direction measurement is not present in the experimental dataset. Otherwise the first attempt would have been to study how changes in wind direction reflect in active power and yaw alarms. Inspired by the experiment to forecast the yaw faults by Astolfi et al. (2020) based on the rotor speed an attempt to create a synthetic variable was made. It relied on exploring the correlation between yaw and nacelle positions together with the occurrence of the yaw alarms. The plots of yaw position and yaw alarms and similarly for nacelle position did not present any correlations. While the plot of a synthetic variable acquired as subtraction of yaw position from the nacelle position variable shows correlations with yaw alarms. This variable derivation can be expressed as

$$\text{Difference} = \text{Nacelle position} - \text{Yaw position}. \tag{7.1}$$

The result of such a plot for February, 2017 is presented in Figure 7.7. The yaw alarms are marked with orange dotted lines. It can be concluded that all the negative spikes in the plot of the synthetic variable coincide with the yaw alarms occurrence. The negative sign implies that the the yaw position is much larger than the nacelle position.

The lack of time prevented from further analysis of the reference points for both nacelle and yaw position. Otherwise an explanation for the observed data might have been arrived at. And this synthetic feature should be combined together with the class of engineered features which are discussed in Section 7.9.

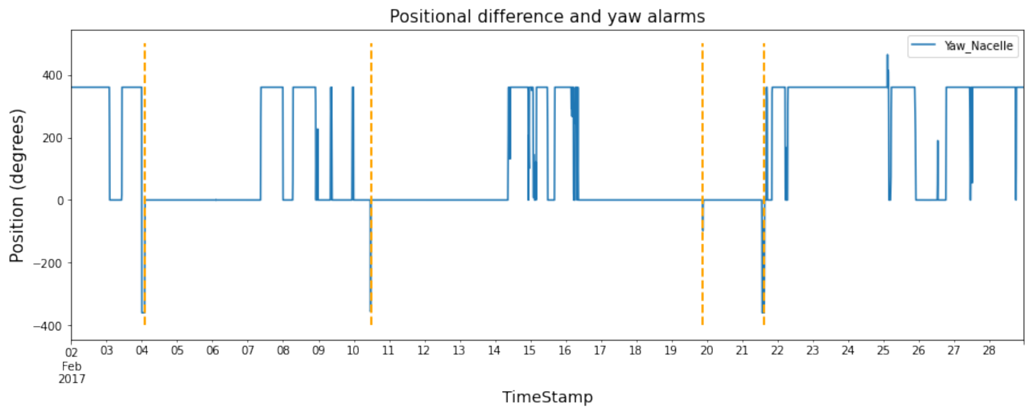


Figure 7.7: Caption

7.5 Frequently occurring yaw alarms by category

While conducting analysis of correlation between the aforementioned synthetic variable and yaw alarms, the in-depth analysis of yaw alarms code and their occurrence for all turbines for the whole period of 2007 - 2017 was conducted. It involved filtering out the yaw alarms by code and calculating the total number of occurrences. The conclusion that with confidence it can be stated that the most often occurring yaw alarm is connected to the untwisting of the cables. It number of instances for all turbines in Smøla 2 lies around 500. The untwisting is required as the yaw controller in order to adjust yaw error rotates the nacelle in one direction until a certain set limit. When this limit is reached the turbine has to be stopped and rotated back.

The second most occurring alarm codes is related to yaw direction error which was assumed previously as the leading cause of yaw faults. Its frequency is quite lower and lies around 40 instances for the whole period, i.e. around 3 instances a year.

The third most often occurring error code is connected to the forced yawing one round with the average number of instances lying around 20. It might be that this forced rotation is connected with the untwisting error.

7.6 Missing data

The process at acquiring a representative turbine required statistical analysis of active power parameter for all turbine over the whole period of 2007 - 2017. In order to compare the sum of the total active power generated by each turbine over this period the missing data analysis and imputation are employed. Figure 7.8. The plot allows to analyse visually with ease the amount of missing values and the assess if there is certain pattern occurring. It can be concluded that for in the period between 2008 and 2010 there are several periods of data missing for the whole park. In regards to 2017 the same is observed for September, 2017.

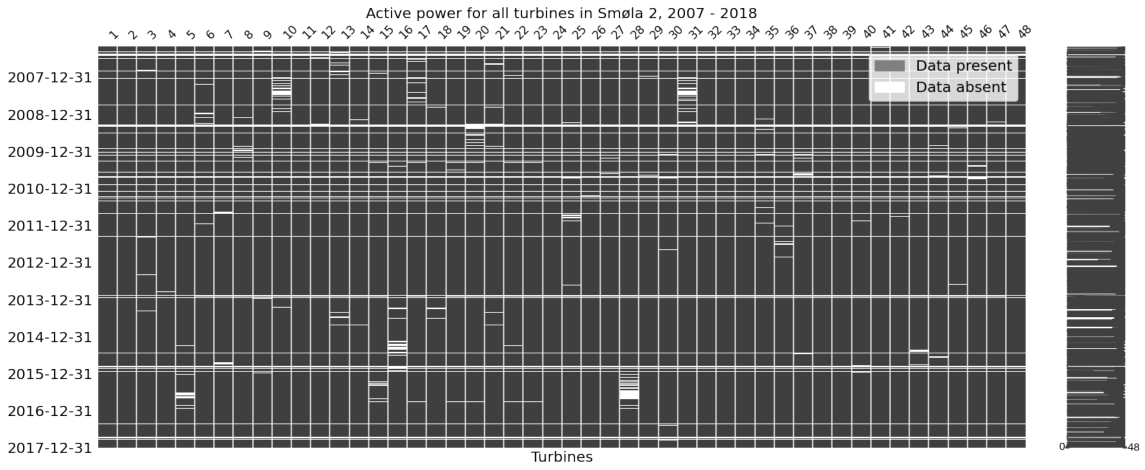


Figure 7.8: Missing data of all turbines at Smøla 2 for the period between 2007 and 2017

It was decided to remove such periods where the whole park lacks measurements for active power in order to minimise data in need of imputation. So the various types of imputation were applied to data with the limit set to 17 steps. This limit sequence was derived from the analysis of autocorrelation values for all turbines. The autocorrelation score was kept at 0.73 for all turbines. The autocorrelation scores and their plots allow to see if there is correlation between the various time stamps of the same variable. It means it allows to measure the correlation of the variable with itself over time what is highly relevant for analysis of time series data.

7.7 Imputation

The results of various imputation methods were studied on active power data as well as nacelle position data in order to choose the one that is used for imputation of the whole dataset. This the imputed data set is later used for LSTM model fitting. The imputation methods used are described in Section 4.1.

Linear interpolation is the method chosen for the experiment based on the analysis of all methods. Hence the results of only this method are presented in this section.The time based linear interpolation takes the time index into consideration, while the regular linear interpolation ignores the index. The result of imputation of nacelle position on March, 24 is presented in Figure 7.9 and Figure 7.10.

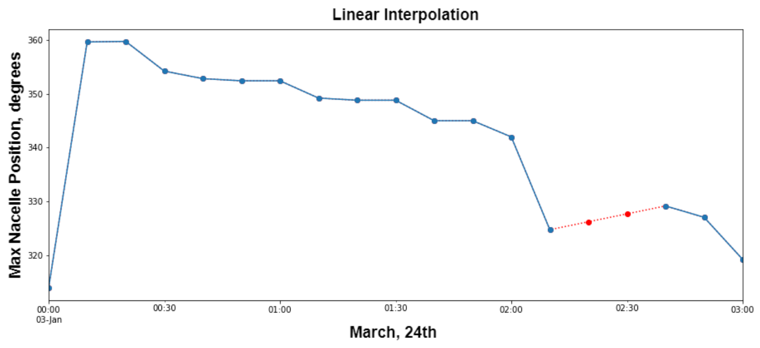


Figure 7.9: Linearly interpolated nacelle position data between 02:20 and 02:40 on March, 24.

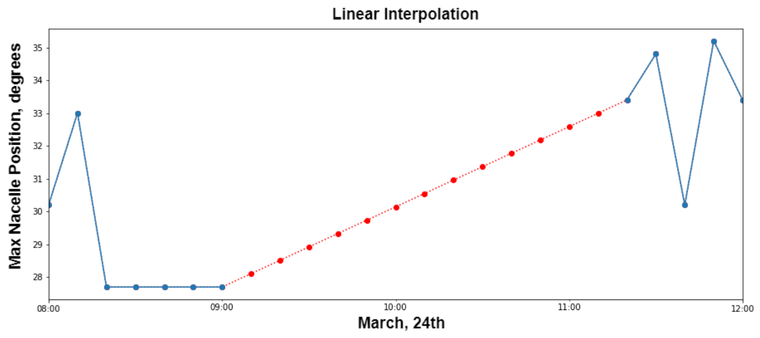


Figure 7.10: Linearly interpolated nacelle position data between 09:00 and 11:10 on March, 24.

7.8 Active power plot

The active power measurements were analysed visually together with the values from the power reference curve as the one presented in Section 2.2.2. The observational data follows well the theoretical curve and the presence of anomaly or outlier data is detected in the plot. Similar analysis is used for normal behaviour model employment which is presented in Section 4.3. Based on the results presented in Section 8.2 and literature review in Section 4.3 the forecasting of active power in order to identify the on-start of yaw alarms might be a better option for the problem.

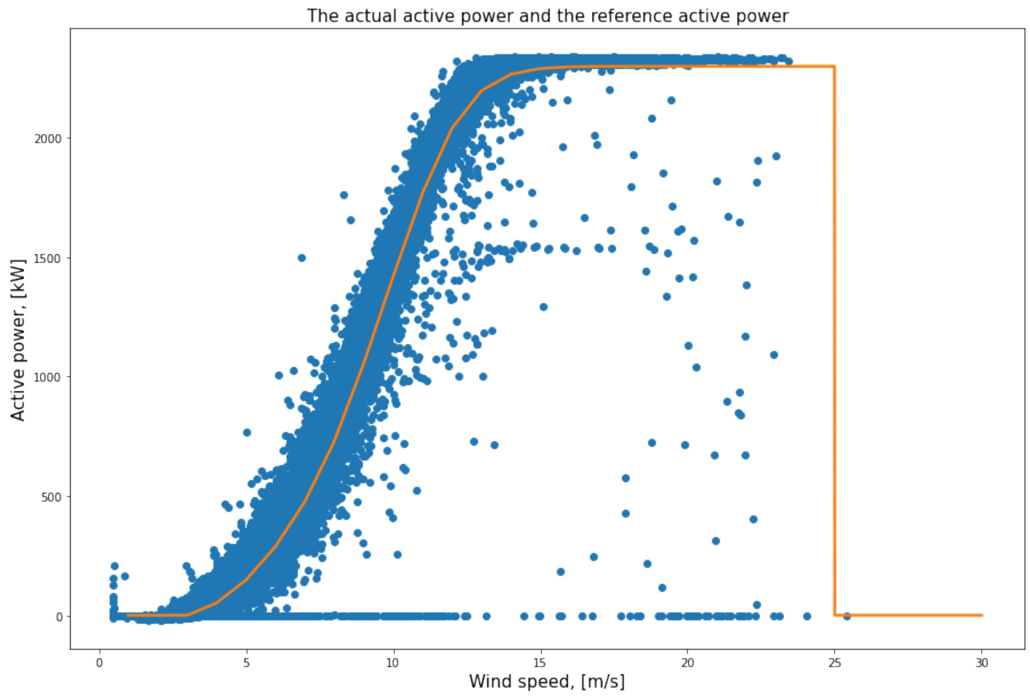
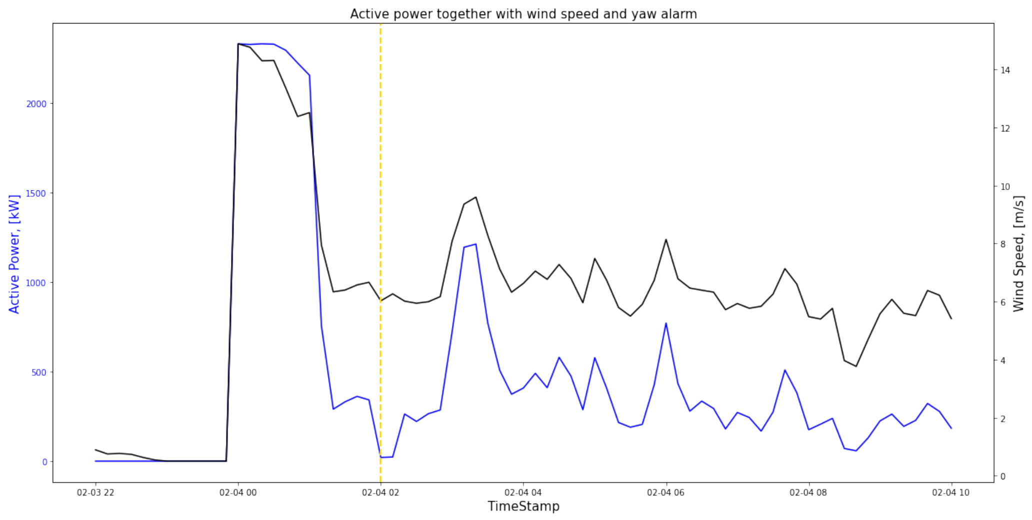


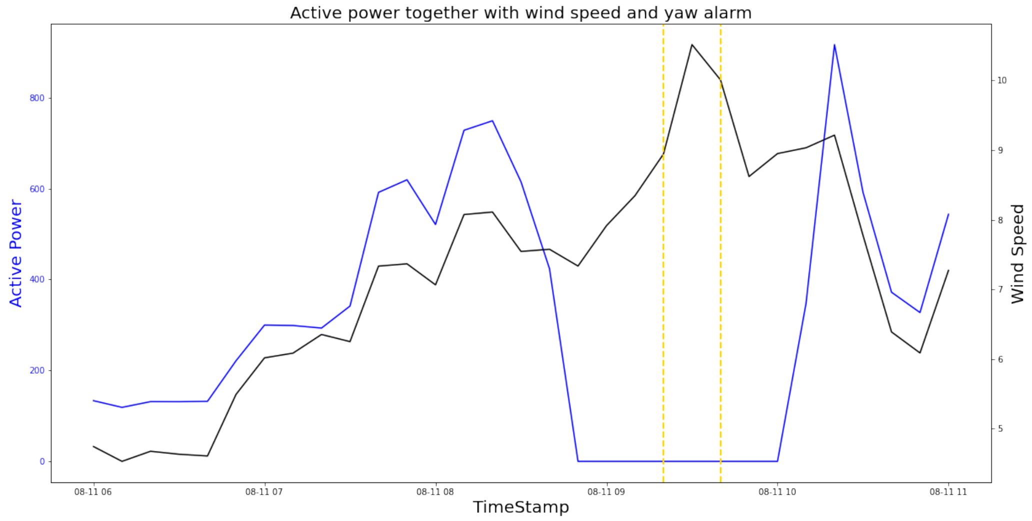
Figure 7.11: Caption

The combined plots of active power on one y-axis and the wind speed on the other y-axis with the yaw alarm as a vertical lines is presented in Figure 7.12a and Figure 7.12b. The first observation as well as from the decomposition plots in Section 7.3 is that how closely the active power graph follows the wind speed graph. Here it is possible to observe their correlation lowest possible level for SCADA data. The wind measurements observed in these plots lie almost entirely within the normal wind turbine operational mode as specified in Section 2.2.2. Hence if not the yaw faults instances the active power graph should have continued to follows the wind speed graph.

The behaviour of active power graph differs in these two plots. In the first case the active power is close to zero only for a short period of time, while in the second case with the period of zero active power is longer. The wind is still within the normal operation range and can not be the cause. The occurrence of two alarms, with the same code as it was found out, might be the reason. Moreover the instances of other alarms is not plotted and this period can be explained by other faults.



(a) Yaw alarm on-start at approximately 02:00 on February, 2.

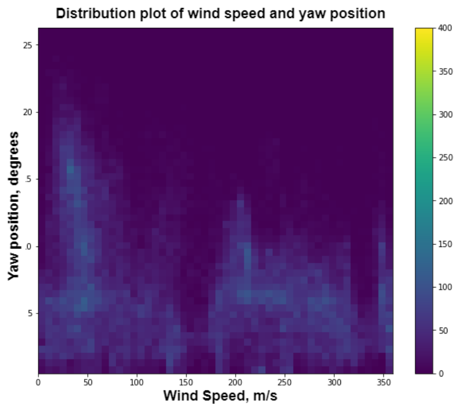


(b) Yaw alarms on-start at approximately 9:20 and 9:30 on August, 11.

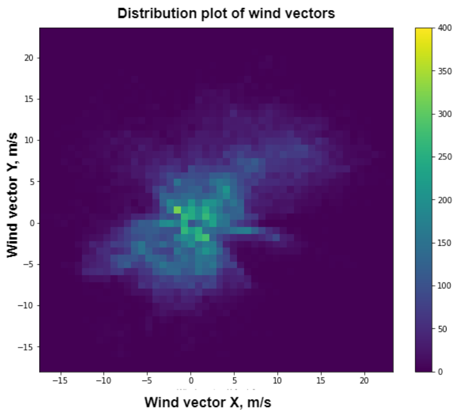
Figure 7.12: Combined plot of active power together with wind speed and the yaw alarm on-star for two time periods.

7.9 Feature Engineering

The method of feature engineering described in in Section 6.6 transforms wind speed and yaw position measurements into two wind vectors. This influences the correlation between the variables distributions. Such technique allows a machine learning model to interpret variables correctly easier (TensorFlow, 2022b). The lightest green colour in the plot corresponds to 0 value and this density plot with two variables can be seen as 3D normal distribution plot with two of the dimensions presented by each variable.



(a) Distribution plot of wind speed and yaw position.



(b) Distribution plot of wind vectors.

Figure 7.13: Distribution plots of yaw position and wind speed before and after preprocessing.

7.10 Feature Selection

The initial analysis of the features relation to the target variable was conducted applying the correlation matrix. The correlation plot would not be difficult to decode as there are far two many features for visualisation. The conclusion is that the target value, which the probability of the yaw error to occur, has low correlation to the most of the parameters in the dataset. The top five features are presented in Table 7.2. The correlation matrix contains values for Pearson correlation between all the features. The low correlation score can be explained by the complexity of the data set and the non-linearity of data. Correlation matrix is optimal to be used with linear data and thus low correlation implies only that there is no linear relationship.

That is why the second attempt was done with XGBoost Regressor as this model has been used for time series forecasting as mentioned in 4.3. The results for the feature selection based on feature importance score are obtained as the result of fitting XGBoost Regressor on the dataset. The graphical output of the model is presented in Figure B.1 in Section B due to its size. The names of the parameters were substituted with numbers due to the aforementioned fact that there is no common taxonomy for the names. Thus the names for Smøla dataset can be unique. The manual substitution of the features it too time consuming. The 99 features that has the highest score with XGBoost feature importance can be grouped in the following manner:

- Yaw position measurements mean, max and min;
- Active and reactive power;
- Mean wind measurements, gust wind;
- grid frequency, voltage;

Correlation matrix for the dataset with all parameters and missing values present		
Feature name	Correlation value for raw target	Correlation for probability
Yaw unwanted counts	0.72	0.47
Yaw unwanted endvalue	0.69	0.45
Accumulated environmental error	0.66	0.43
Feature counts	0.65	0.42
Frequency error endvalue	0.53	0.37

Table 7.2: Correlation matrix for the full dataset with missing values

- various measurements connected with communication services;
- nacelle temperature;
- pitch position

Chapter 8

LSTM models results and discussion

This chapter presents at first the results of data preprocessing related to the machine learning such as data scaling, train-validation-test ratio. The major part of the chapter is dedicated to the main goal of this thesis - to test various hypothesis regarding LSTM models performance improvement. The results for various models modifications are presented and assessed with learning curves and performance metrics.

8.1 Scaling

As it mentioned before neural networks models' learning capacity allows to process data without scaling. Though application of scaling methods allows to shorten the convergence time which is an important parameter for model assessment.

The transformation results of MinMAx and RobustScalers are presented in Figure 8.1. Subplot **a** presents the original features if various range. In the subplot **b** the range of the variables is squeezed into the range between 0 and 1 with, while the subplot **c** demonstrates scaling of wind speed between -1 and 3 and nacelle position between -1 and 1. These results are obtained by ignoring the mean value for each variable correspondingly. This technique allows to avoid using the mean value influenced by the outliers. As the data set is expected to contain outliers and their removal can not be performed the RobustScaler was used for scaling data for almost all the problems.

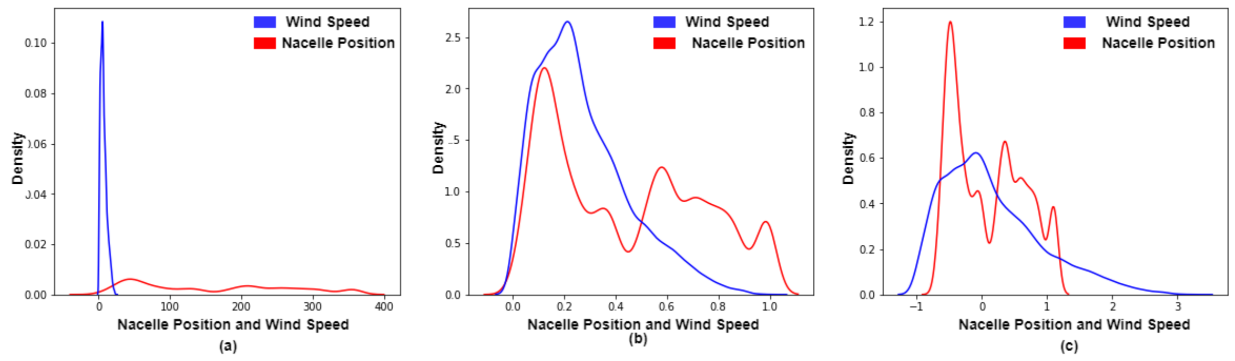


Figure 8.1: A example of how the range of the original data (a) is changed with MinMaxScaler (b) and RobustScaler (c).

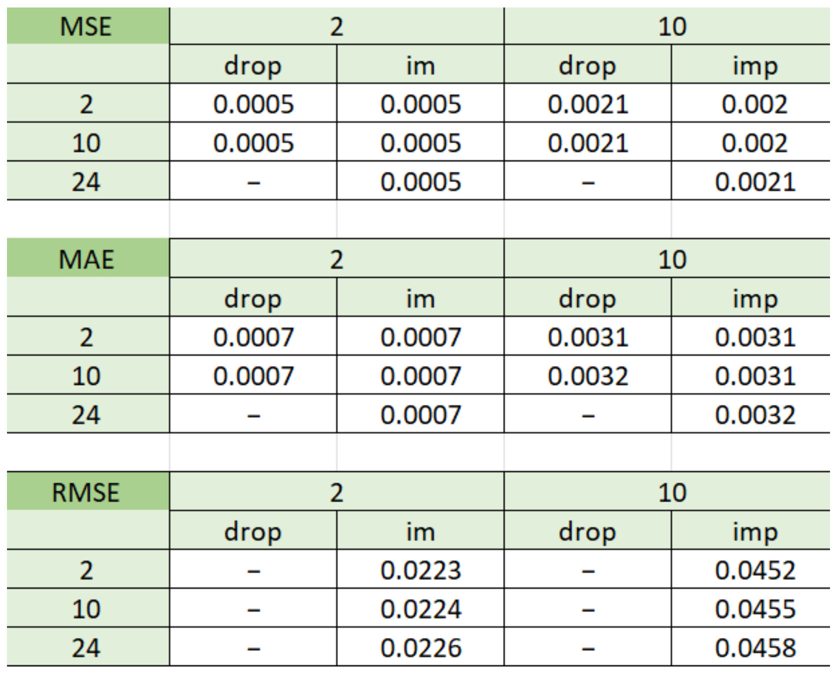
8.2 LSTM models

The LSTM models in the experiment can be at first divided into three groups:

- Models trained on the data set with all features;
- Models trained on the data with the selected features in accordance with XGBoost scores;
- Models with randomly selected by qualitative guess.

8.2.1 Results for the models with all features

The results from show that all the models have low MSE and MAE results, but what is more interesting is to see how they changed over the training epochs. Here only one result is presented as Figure **??**, while some of the learning curves are presented in Annex D. The reason for not presenting all is that they all have rather similar trend similar to one in Figure 8.3.



MSE	2		10	
	drop	im	drop	imp
2	0.0005	0.0005	0.0021	0.002
10	0.0005	0.0005	0.0021	0.002
24	–	0.0005	–	0.0021

MAE	2		10	
	drop	im	drop	imp
2	0.0007	0.0007	0.0031	0.0031
10	0.0007	0.0007	0.0032	0.0031
24	–	0.0007	–	0.0032

RMSE	2		10	
	drop	im	drop	imp
2	–	0.0223	–	0.0452
10	–	0.0224	–	0.0455
24	–	0.0226	–	0.0458

Figure 8.2: Figure with results presented as table for 12 models run on all features.

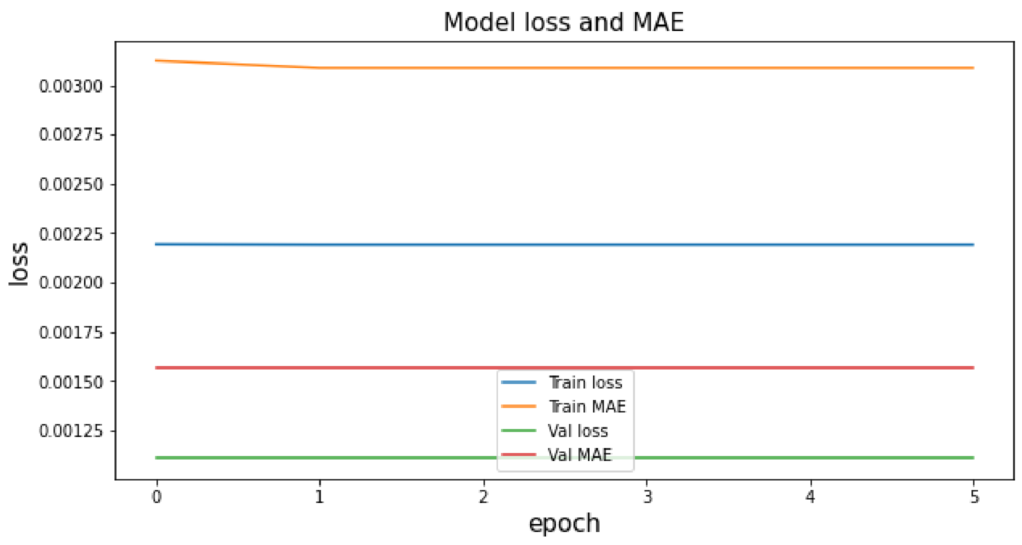


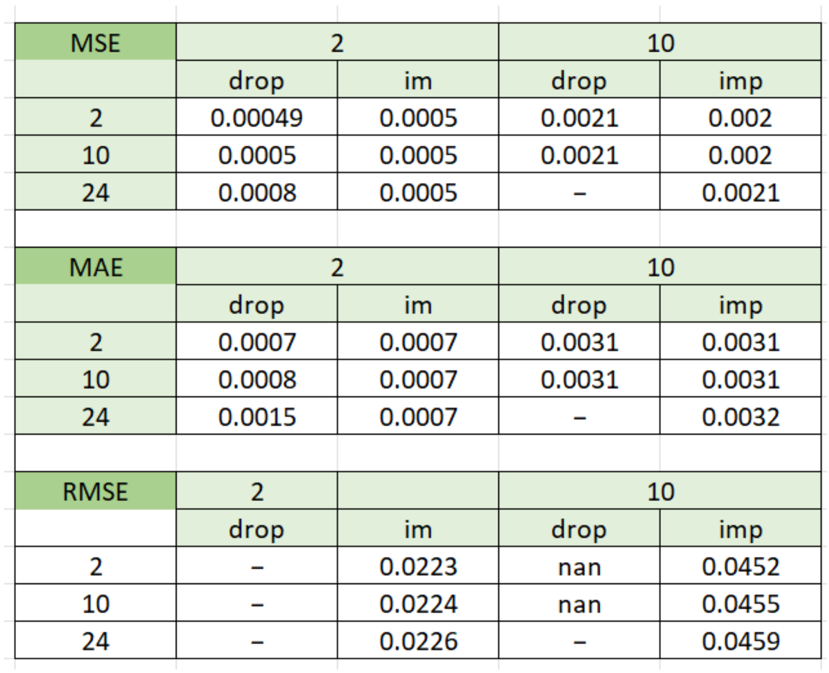
Figure 8.3: Learning curve for the model with all features, 2 hours as target and 2 hours for window length. The missing values were dropped.

The learning curve first of all illustrates that the model is not learning anything which is manifested as no change in the loss function for the model. Second conclusion that can be made from the plot is that training loss is larger than validation loss. This is a clear indication of underfitting. Underfitting means that the model is not complex enough to be able to capture the relation between the target variable and the features. That is why the second block with the same models was trained on the data with extracted features in order to lessen the complexity of the data.

Conclusion for the first block: Despite the fact that the metrics values are low, they experience no changes in time. Some of the models did have an indication of some learning after one or two epochs, but they turned to the straight line as the rest of 12 models.

8.2.2 Results for the models with XGBoost selected features

The results for the models trained on 99 features selected with XGBoost model present the same learning curves behaviour and have low MAE and MSE metrics as well. But as it was mentioned before the metrics can not be relied on alone as they should always be put int he context.



MSE	2		10	
	drop	im	drop	imp
2	0.00049	0.0005	0.0021	0.002
10	0.0005	0.0005	0.0021	0.002
24	0.0008	0.0005	–	0.0021

MAE	2		10	
	drop	im	drop	imp
2	0.0007	0.0007	0.0031	0.0031
10	0.0008	0.0007	0.0031	0.0031
24	0.0015	0.0007	–	0.0032

RMSE	2		10	
	drop	im	drop	imp
2	–	0.0223	nan	0.0452
10	–	0.0224	nan	0.0455
24	–	0.0226	–	0.0459

Figure 8.4: Figure with results presented as table for 12 models run on features selected with Xgboost features.

In this section only one out of 12 learning curves is presented for the same reasons as in Section 8.2.1. Figure 8.5 and Figure 8.6 present similar trend of no learning process by the model at all. This can be explained by the long window length that makes data too complex. While learning curve for the model with 2 hours probability distribution as target and 10 hours window length, imputed missing values, presented i Figure D.4 in Annex D, showed initial learning process.

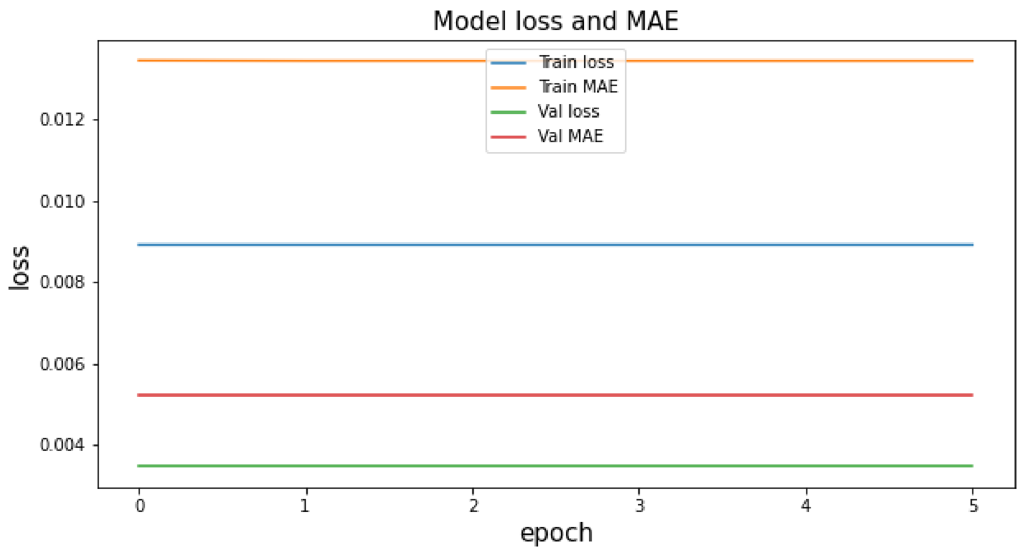


Figure 8.5: Learning curve for the model with 99 features 10 hours as target and 24 hours for window length. The missing values were imputed.

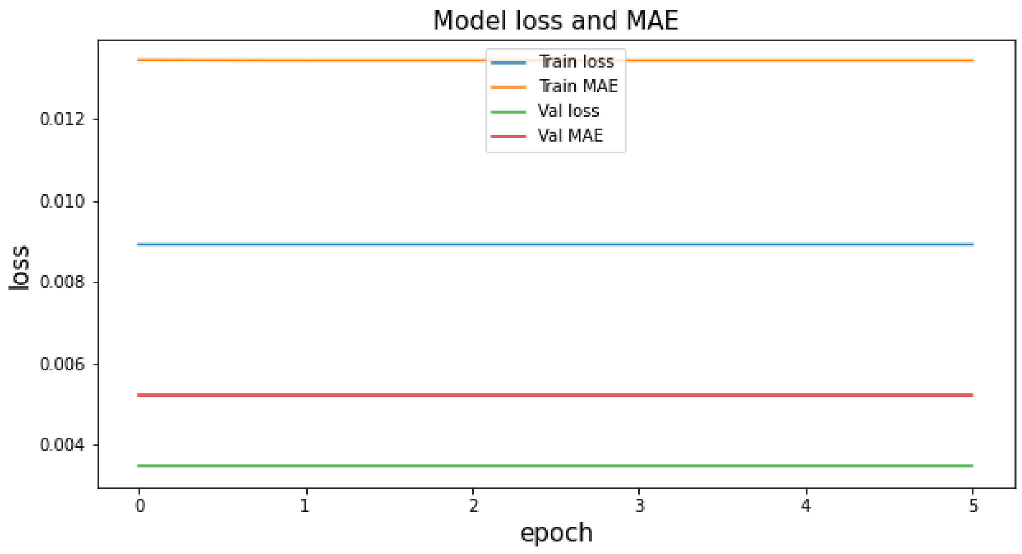
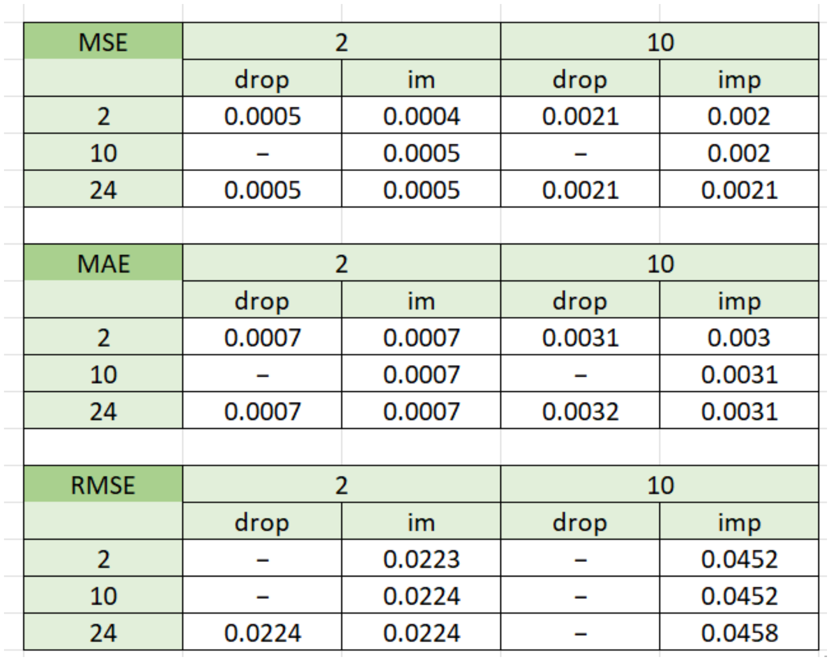


Figure 8.6: Learning curve for the model with 99 features 10 hours as target and 24 hours for window length. The missing values were dropped.

Conclusion for the second block: Removal of more than 600 features chosen to the feature importance score with XGBoost did help to solve the problem with the data being too complex for the model. Thus the next step of the experiment is to try a set with few randomly selected features.

8.2.3 Results for the models with randomly selected features

The MAE and MSE metrics values are still low, but when analysed together with the learning plots curves the same conclusions as earlier can be made. This section contains only two plot as an example for the rest of 12 models.



MSE	2		10	
	drop	im	drop	imp
2	0.0005	0.0004	0.0021	0.002
10	–	0.0005	–	0.002
24	0.0005	0.0005	0.0021	0.0021

MAE	2		10	
	drop	im	drop	imp
2	0.0007	0.0007	0.0031	0.003
10	–	0.0007	–	0.0031
24	0.0007	0.0007	0.0032	0.0031

RMSE	2		10	
	drop	im	drop	imp
2	–	0.0223	–	0.0452
10	–	0.0224	–	0.0452
24	0.0224	0.0224	–	0.0458

Figure 8.7: Figure with results presented as table for 12 models run on randomly selected features.

The learning curves for models in Figure 8.8 and Figure 8.9 show that the model managed to learn some infomation from the dataset with dropped values.

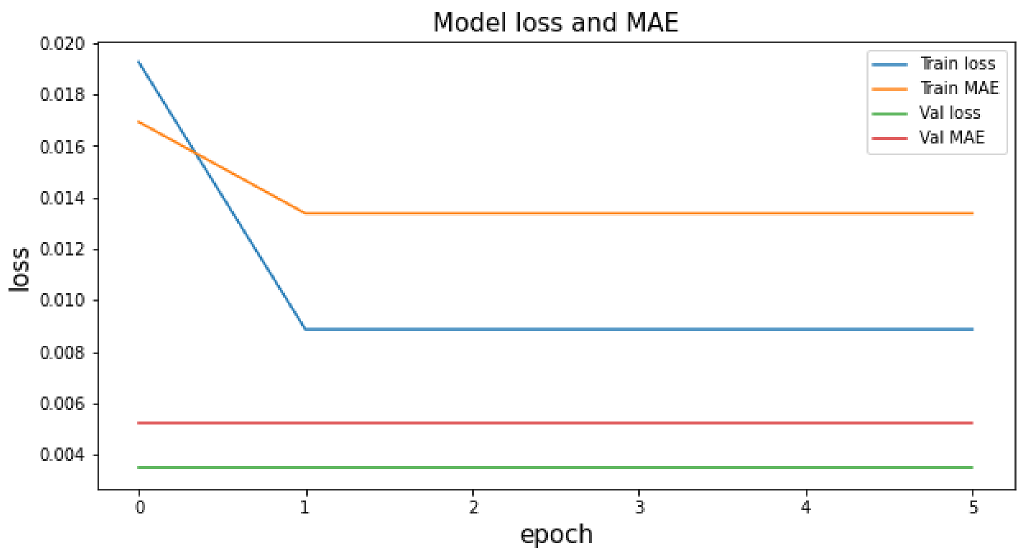


Figure 8.8: Learning curve for the model with 17 features, 10 hours as target and 24 hours for window length. The missing values were dropped.

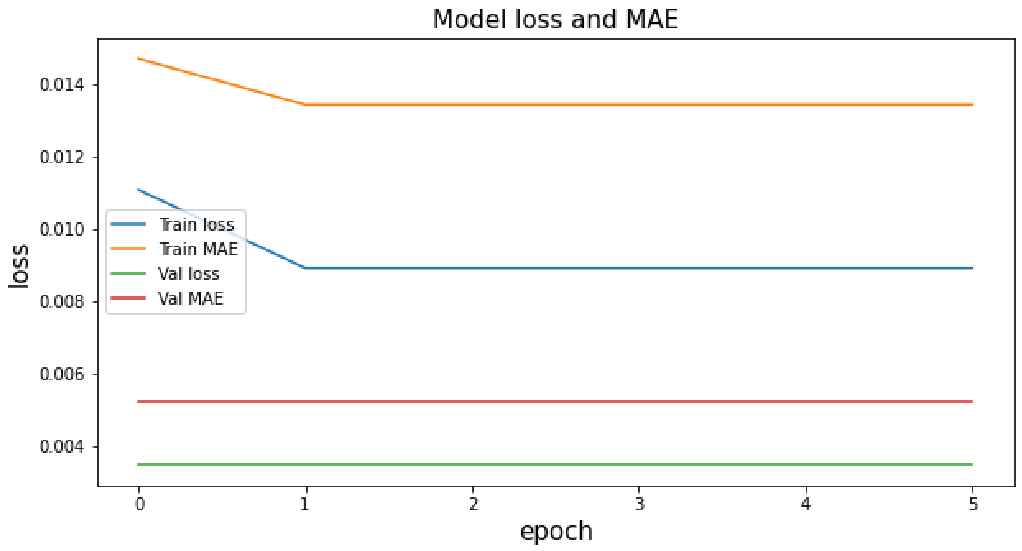


Figure 8.9: Learning curve for the model with 17 features, 10 hours as target and 24 hours for window length. The missing values were imputed.

8.2.4 Simple model

As an attempt to find the best appropriate model for the forecasting problem, a simple model with one input LSTM layer with 32 nodes and Dense layer output trained on the data set with randonmly selected 17 features, dropped missing values showed surprisingly much better results. The different parameter used was made was the ratio of the train-validation split. It was changed to 60/40. As well as the scaling method was tried both with Robust and MinMaxScalers, with the later showing a bit better result.

The results of the model are presented in Figure 8.10 and Figure 8.11 which present the learning curve for the model and the forecasted test alarm. It is obvious that the model converges and it does manage to learn from the data. As well as the test probability distribution for a yaw alarm on-start is predicted well. Though these results are not consistent and various runs of the model lead to different results, even to the similar behaviour as the previous 12 models - no converging and underfitting.

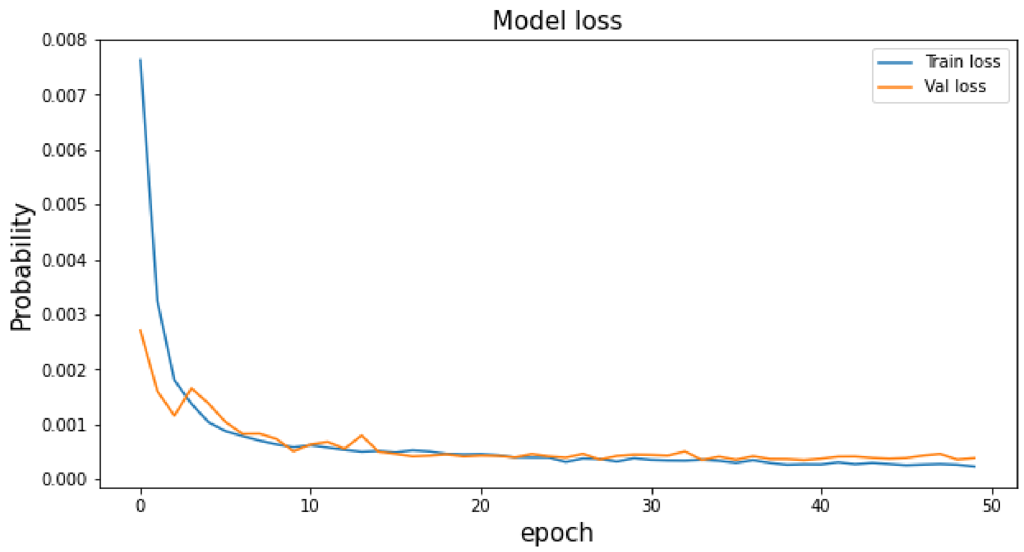


Figure 8.10: Learning curve for simple model with 1 LSTM layer

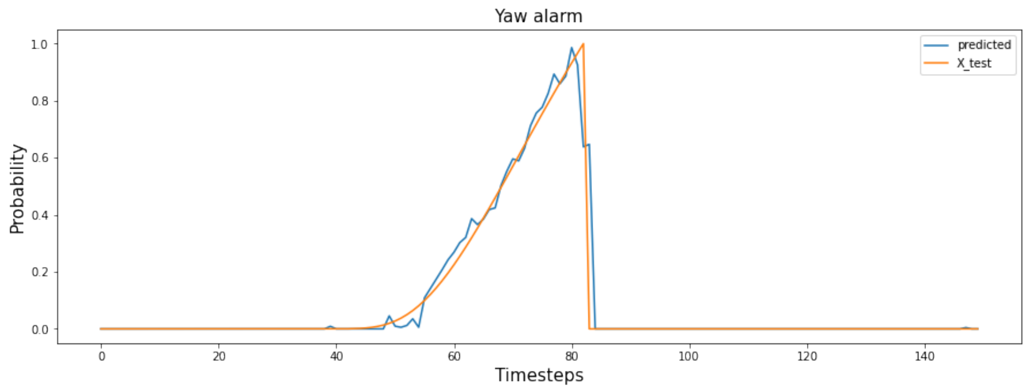


Figure 8.11: Forecasting made for test data with simple model with 1 LSTM layer.

8.3 General remarks to the LSTM models chapter

The previous presented section with the results for training 12 different models, varying in the number of features used in the data, in the method used for handling missing data, in window length for input sequence and in the length of probability for alarm to be triggered. None of those models show a sign of being able to learn from the data as the dataset might be too complex for the models. The differences between the various 12 models were minimal and thus it is not possible to state if some observed variations are random or something that should be examined further. A surprising result with a simple model with 1 LSTM layer implies that the choice in increasing the data available to the model is a wrong strategy.

Consideration if the longer tuning period both for hypertuner and the model itself could improve the situation were tested without results presented here as it had no effect on the model. It was observed that while hyperparameter was tuning on complex models it used 2 epochs maximum per a trial, while hypertuning for a simple model had the maximum number of epochs per trial equal to 5. Thus it might be concluded that exploration of a simpler model with a smaller dataset containing handpicked parameters relying on the domain knowledge can bring better results.

The correlation matrix method can be used to reduce the number of features by removing the ones with high linear correlation score.

Later it was also detected that the method for yaw alarms synchronisation with the rest of the data was conducted wrongly and about 30 alarms, i.e. approximately a half, were missing. Thus it puts the whole experiment under the scrutinisation as the data provided to the deep learning models was not preproccessed properly after all.

Chapter 9

Conclusion

In order to improve the maintenance of wind turbines in regards of occurring yaw faults stacked LSTM models were studied as possible solution. Unfortunately, the result shows that with the formulation of problem as prediction of the probability of a yaw alarm on-start by stacked LSTM model can not be solved as the trained models experienced difficulty at learning information from the dataset.

The influence of the number of features, missing values handling method, window size and target probability length can not be conclusively decided on as all the problems have experienced difficulty to approximately same extent.

A simple model tested for a comparison gave a much better result but its behaviour is to randomised for it to be defined as optimal solution.

9.1 Suggestions for further research

An interesting direction for solving this problem is to try to predict the faults based on the normal behaviour problem and also to find other continuous variable as the target for prediction. Various modifications of LSTM models can be employed such as autoencoder-decoder architectures and bidirectional architecture.

References

Abdovic, S., Cuk, M., Cekada, N., Milosevic, M., Geljic, A., Fusić, S., & Bastic, M. (2019, 09). Predicting posterior urethral obstruction in boys with lower urinary tract symptoms using deep artificial neural network. *World Journal of Urology*, *37*(9), 973-1979. (https://doi.org/10.1007/s00345-018-2588-9)

Astolfi, D., Castellani, F., Becchetti, M., Lombardi, A., & Terzi, L. (2020). Wind Turbine Systematic Yaw Error: Operation Data Analysis Techniques for Detecting It and Assessing Its Performance Impact. *Energies*, *13*(9). (https://doi.org/10.3390/en13092351)

Badurek, C. A. (2015). *wind turbine.* Encyclopedia Britannica. (Available at: https://www.britannica.com/technology/wind-turbine#/media/1/645101/125134(accessed: 12/02/2022))

Berg, J., Mann, J., & Nielsen, M. (2013). *Notes for dtu course 46100; introduction to micro meteorology for wind energy.* [Lecture notes]. (Available at: https://core.ac.uk/reader/13797797(accessed: 12/02/2022))

Biswal, A. (2021). *Recurrent Neural Network (RNN) Tutorial: Types, Examples, LSTM and More.* (Available at: https://blog.tensorflow.org/2019/02/mit-deep-learning-basics-introduction-tensorflow.html(accessed: 12/02/2022))

Bjørnstad, J. (2017). *Varians.* (Available at: https://snl.no/varians(accessed: 12/02/2022))

Bonometti, V. (2020). *GitHub Repository: Kears tuner/Issues /Epochs argument in Hyperband search method.* (Available at : https://github.com/keras-team/keras-tuner/issues/212 (accessed: 12/02/2022))

Brownlee, J. (2016). *Feature Importance and Feature Selection With XGBoost in Python.* (Available at: https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/ (accessed: 12/02/2022))

Brownlee, J. (2020). *How to Diagnose Overfitting and Underfitting of LSTM Models.* (Available at: https://machinelearningmastery.com/diagnose-overfitting-underfitting-lstm-models/ (accessed: 13/02/2022))

Brownlee, J. (2021). *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning.* (Available at: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/(accessed: 12/02/2022))

Burnett, C. M. L. . (2006). *Illustration of the topology of a generic Artificial Neural Network (ANN).* (Available at: https://commons.wikimedia.org/w/index.php?curid=1496812 GNU Free Documentation License (accessed: 12/02/2022))

B V Vishwas, A. P. (2020). *Hands-on Time Series Analysis with Python: From Basics to Bleeding Edge Techniques*. Apress.

Chen, B., Xie, L., Li, Y., & Gao, B. (2020). Acoustical damage detection of wind turbine yaw system using Bayesian network. *Renewable Energy*, *160*, 1364-1372. (`https://doi.org/10.1016/j.renene.2020.07.062`)

Chen, H., Liu, H., Chu, X., Liu, Q., & Xue, D. (2021). Anomaly detection and critical SCADA parameters identification for wind turbines based on LSTM-AE neural network. *Renewable Energy*, *172*, 829-840. (`https://doi.org/10.1016/j.renene.2021.03.078`)

Chollet, F. (2018). Deep learning for timeseries. In *Deep Learning with Python* (chap. 10). Manning. (`https://livebook.manning.com/book/deep-learning-with-python-second-edition/chapter-10/`)

Danish Wind Industry Association. (2003). *Wake Effect.* (Available at: `http://xn--drmstrre-64ad.dk/wp-content/wind/miller/windpowerweb/en/tour/wres/wake.htm`(accessed: 12/02/2022))

EEA. (2021). *Wind — mean wind speed.* (Available at: `https://www.eea.europa.eu/publications/europes-changing-climate-hazards-1/wind/wind-mean-wind-speed`(accessed: 12/02/2022))

Encalada-Dávila, , Puruncajas, B., Tutivén, C., & Vidal, Y. (2021, March). Wind Turbine Main Bearing Fault Prognosis Based Solely on SCADA Data. *Sensors (Basel, Switzerland)*, *21*(6). (`https://doi.org/10.3390/s21062228`)

Fang, C., & Wang, C. (2020). Time Series Data Imputation: A Survey on Deep Learning Approaches. *CoRR*, *abs/2011.11347*. (`https://arxiv.org/abs/2011.11347`)

Fridman, L. (2019). *MIT Deep Learning Basics: Introduction and Overview with TensorFlow .* (Available at: `https://blog.tensorflow.org/2019/02/mit-deep-learning-basics-introduction-tensorflow.html`(accessed: 12/02/2022))

Gasch, R., & Twele, J. (2011). *Wind power plants: Fundamentals, design, construction and operation.* Springer Berlin Heidelberg. Retrieved from `https://books.google.no/books?id=c-QB9PiC_GMC`

German Wikipedia. (2011). *Schema Windenergieanlage.* [Image]. (Available at: `https://commons.wikimedia.org/wiki/File:Schema_Windenergieanlage.svg`(accessed: 12/02/2022))

Hau, E. (2006). *Wind turbines: Fundamentals, technologies, application, economics* (2nd ed.). Berlin: Springer. (Available at `https://doi.org/10.1007/3-540-29284-5`)

Helbing, G., & Ritter, M. (2018). Deep Learning for Fault Detection in Wind Turbines. *Renewable and Sustainable Energy Reviews*, *98*, 189–198. (`https://doi.org/10.1016/j.rser.2018.09.012`)

Insuasty, A., Tutivén Gálvez, C., & Segui, Y. (2021, 09). SCADA Data-Driven Wind Turbine Main Bearing Fault Prognosis Based on One-Class Support Vector Machines. *Renewable Energy and Power Quality Journal*, *19*, 338-343. (`http://dx.doi.org/10.24084/repqj19.290`)

ISO Central Secretary. (2013). *Wind turbines - Part 12-2: Power performance of electricity-producing wind turbines based in nacelle anemometry* (Standard No. ISO/IEC 61400-12-2:2013). Geneva, CH: International Organization for Standardization. Retrieved from `https://www.iso.org/standard/62711.html`

Jeon, C., & Whitehead, A. (2019, Sep 23–26). HANDLING DATA GAPS IN TIME SERIES USING IMPUTATION [Conference Presentation]. In *Strata Data Conference 2019,New York, NY, United States.* (Available at: `https://machinelearningmastery.com/time-series-data-stationary-python/`(accessed: 12/02/2022))

Jing, B., Qian, Z., Pei, Y., Zhang, L., & Yang, T. (2020). Improving wind turbine efficiency through detection and calibration of yaw misalignment. *Renewable Energy*, *160*, 1217-1227. (`https://doi.org/10.1016/j.renene.2020.07.063`)

Jones, M. T. . (2017). *Recurrent neural networks deep dive.* (Available at: https://developer.ibm.com/articles/cc-cognitive-recurrent-neural-networks/?mhsrc=ibmsearch_a&mhq=rnn(accessed: 12/02/2022))

Kandukuri, S. T., Robbersmyr, K. G., & Karimi, H. R. (2018). Toward farm-level health management of wind turbine systems: status and scope for improvements. In H. R. Karimi (Ed.), *Structural Control and Fault Detection of Wind Turbine Systems* (p. 149 - 167). The Institution of Engineering and Technology. (https://doi.org/10.1049/PBPO117E_ch6)

Karpathy, A. (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks.* (Available at: http://karpathy.github.io/2015/05/21/rnn-effectiveness/(accessed: 12/02/2022))

Kavlakoglu, E. . (2020). *AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?* (Available at: https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks(accessed: 12/02/2022))

Kim, M., & Dalhoff, P. (2014). Yaw Systems for wind turbines – Overview of concepts, current challenges and design methods. *Journal of Physics: Conference Series*, *524*, 012086. (https://doi.org/10.1088/1742-6596/524/1/012086)

Landberg, L. (2015). *Meteorology for wind energy* (1st ed.). Chichester: John Wiley Sons. (Available at https://doi.org/10.1002/9781118913451)

Lazzeri, F. (2020). *Machine Learning for Time Series Forecasting with Python®.* John Wiley Sons, Inc. doi: 10.1002/9781119682394

Liu, Z., & Hilmisson, R. (2014). Yaw bearing system fault detected. *UPTIME MEGAZINE*, *1*, 8-11. (Available at https://staging.bkvibro.com/fileadmin/mediapool/Internet/Case_Stories/YAW_BEARING_SYSTEM_FAULT_DETECTED_en_ext.pdf (accessed: 12.02.2022))

Lutins, Evan. (2017). *Ensemble Methods in Machine Learning: What are They and Why Use Them?* (Available at: https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f(accessed: 12/02/2022))

Macas, M., Moretti, F., Lauro, F., Pizzuti, S., Annunziato, M., Fonti, A., … Giantomassi, A. (2014). Importance of feature selection for recurrent neural network based forecasting of building thermal comfort. In A. Bouchachia (Ed.), *Adaptive and intelligent systems* (pp. 11–19). Cham: Springer International Publishing. (https://doi.org/10.1007/978-3-642-23857-4)

Machine Learning Mastery. (2021). *XGBoost for Regression.* Retrieved 04.02.2022, from https://machinelearningmastery.com/xgboost-for-regression/

Malik, U. (2019). *Solving Sequence Problems with LSTM in Keras.* (Available at: https://stackabuse.com/solving-sequence-problems-with-lstm-in-keras(accessed: 12/02/2022))

Malt, U. (2020). *Kovarians.* (Available at: https://snl.no/kovarians(accessed: 12/02/2022))

Manwell, J., McGowan, J., & Rogers, A. (2010). *Wind Energy Explained: Theory, Design and Application* (2nd ed.). Chichester: John Wiley Sons. (Available at https://doi.org/10.1002/9781119994367)

Mikulski, B. (2019). *Using Hyperband for TensorFlow hyperparameter tuning with keras-tuner.* (Available at : https://www.mikulskibartosz.name/using-hyperband-for-tensorflow-hyperparameter-tuning-with-keras-tuner (accessed: 12/02/2022))

NASA. (n.d.). *'celled model'.* (Available at: https://sealevel.jpl.nasa.gov/files/ostm/6_celled_model.jpg(accessed: 12/02/2022))

Nielsen, A. (2019). *Practical Time Series Analysis.* O'Reilly Media, Inc.

NVE. (n.d.). *NVE Atlas 3.0.* (Available at: https://atlas.nve.no/Html5Viewer/index.html?viewer=nveatlas#(accessed: 12/02/2022))

NVE. (2008). *NVE Vindressurser.* (Available at: `https://temakart.nve.no/link/?link=vindressurser`(accessed: 12/02/2022))

Office of Energy Efficiency and Renewable Energy. (2021). *Wind Turbines: the Bigger, the Better.* (Available at: `https://www.energy.gov/eere/articles/wind-turbines-bigger-better`(accessed: 12/02/2022))

Pal, A., & Prakash, P. (2017). *Practical time series analysis: Master time series data processing, visualization, and modeling using python.* Packt Publishing. Retrieved from `https://books.google.no/books?id=EJpGDwAAQBAJ`

Prost, J. (2020). *Hands on Hyperparameter Tuning with Keras Tuner.* (Available at: `https://www.sicara.ai/blog/hyperparameter-tuning-keras-tuner`(accessed: 12/02/2022))

Raschka, S., & Mirjalili, V. (2019). *Python machine learning: Machine learning and deep learning with python, scikit-learn, and tensorflow 2, 3rd edition.* Packt Publishing. Retrieved from `https://books.google.no/books?id=sKXIDwAAQBAJ`

Röckmann, C., Lagerveld, S., & Stavenuiter, J. (2017). Operation and maintenance costs of offshore wind farms and potential multi-use platforms in the dutch north sea. In B. H. Buck & R. Langan (Eds.), *Aquaculture perspective of multi-use sites in the open ocean: The untapped potential for marine resources in the anthropocene* (pp. 97–113). Cham: Springer International Publishing. Retrieved from `https://doi.org/10.1007/978-3-319-51159-7_4` doi: 10.1007/978-3-319-51159-7_4

Sahoo, B., Routray, S., & Rout, P. (2019, 01). Repetitive control and cascaded multilevel inverter with integrated hybrid active filter capability for wind energy conversion system. , *22*. (`https://doi.org/10.1016/j.jestch.2019.01.001`)

Sheng, S. (2013, June). *Report on Wind Turbine Subsystem Reliability A Survey of Various Databases* (Tech. Rep. No. NREL/PR-5000-59111). National Renewable Energy Laboratory. (Available at `https://www.nrel.gov/docs/fy13osti/59111.pdf`(accessed on:10.12.2021))

Siemens. (n.d.). *Turbine alarms and events description.* PDF-file. (Internal document)

Siemens AG. (2011, 9). *Built on experience SWT-2.3-82.* (Available at:`https://pdf.archiexpo.com/pdf/siemens-gamesa/swt-23-82/88089-134475.html`(accessed: 12/02/2022))

Statkraft. (2007). *Smøla vindpark.* (Available at: `https://web.archive.org/web/20071106170211/http://statkraft.no/pub/vindkraft/vindparker/Smola/index.asp`(accessed: 12/02/2022))

Statkraft. (2011, 9). *Smøla Wind Farm.* (Available at: `https://www.statkraft.com/globalassets/old-contains-the-old-folder-structure/documents/faktaark-smola-wind-farm-eng-sept-2011_tcm9-17664.pdf`(accessed: 12/02/2022))

Stetco, A., Dinmohammadi, F., Zhao, X., Robu, V., Flynn, D., Barnes, M., … Nenadic, G. (2019). Machine learning methods for wind turbine condition monitoring: A review. *Renewable Energy*, *133*, 620-635. (`https://doi.org/10.1016/j.renene.2018.10.047`)

Sundaram, R. (2018). *XGBoost Algorithm | XGBoost In Machine Learning.* (Available at: `https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/` (accessed: 12/02/2022))

Sæther, B. B. (2021). *Wind Power Prediction with Machine Learning Methods in Complex Terrain Areas* (Master's Thesis). Tromsø: UiT The Arctic University of Norway. (Available at: `https://hdl.handle.net/10037/21939`(accessed 12/02/2022))

Tallaksrud, S. (2021). *Wind turbine fault prediction using deep learning at Smøla wind farm* (Master's Thesis). Ås : Norwegian University of Life Sciences. (Available at: `https://hdl.handle.net/11250/2786877`(accessed 12/02/2022))

Tang, M., Liang, Z., Wu, H., & Wang, Z. (2021). Fault Diagnosis Method for Wind Turbine Gearboxes Based on IWOA-RF. *Energies*, *14*(19). (https://doi.org/10.3390/en14196283)

TensorFlow. (2022a). *tf.keras.preprocessing.sequence.TimeseriesGenerator.* (Available at: https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/sequence/TimeseriesGenerator(accessed: 12/02/2022))

TensorFlow. (2022b). *Time series forecasting.* (Available at: https://www.tensorflow.org/tutorials/structured_data/time_series (accessed: 12/02/2022))

Tony L. Burton, E. B. D. S. M. G., Nick Jenkins. (2021). *Wind energy handbook.* Wiley.

Trizoglou, P., Liu, X., & Lin, Z. (2021, 12 1). Fault detection by an ensemble framework of Extreme Gradient Boosting (XGBoost) in the operation of offshore wind turbines. *Renewable Energy*, *179*, 945–962. (https://doi.org/10.1016/j.renene.2021.07.085)

V., L. (2021). *Dealing with Missing Values for Data Science Beginners.* (Available at: https://www.analyticsvidhya.com/blog/2021/10/guide-to-deal-with-missing-values/ (accessed: 12/02/2022))

Walter, W., Haferlach, C., Nadarajah, N., Schmidts, I., Kühn, C., Kern, W., & Haferlach, T. (2021, Jun). How artificial intelligence might disrupt diagnostics in hematology in the near future. *Oncogene*, *40*, https://doi.org/10.1038/s41388-021-01861-y. (https://doi.org/10.1038/s41388-021-01861-y)

Wang, K.-S., Sharma, V., & Zhang, Z. (2014, 03). SCADA data based condition monitoring of wind turbines. *Advances in Manufacturing*, *2*, 61 - 69. (https://doi.org/10.1007/s40436-014-0067-0)

Wood, T. (2020). *What is the Sigmoid Function?* (Available at: https://deepai.org/machine-learning-glossary-and-terms/sigmoid-function#:~:text=Asigmoidfunctionisatypeo20activation,thesefunctionsusefulinthepredictionofprobabilities. (accessed: 13/02/2022))

Xiao, C., Liu, Z., Zhang, T., & Zhang, X. (2021). Deep Learning Method for Fault Detection of Wind Turbine Converter. *Applied Sciences*, *11*(3). (http://dx.doi.org/10.3390/app11031280)

Yang, M., Chengbing, H., & Xinxin, F. (2012, 12). Institutions Function and Failure Statistic and Analysis of Wind Turbine. *Physics Procedia*, *24*, 25–30. (https://doi.org/10.1016/j.phpro.2012.02.005)

Zaiontz, C. (n.d.). *Basic Concepts of Correlation.* (Available at: http://www.real-statistics.com/correlation/basic-concepts-correlation/(accessed: 12/02/2022))

Zhang, X., Yang, C., Li, X., Yan, D., & Li, H. (2021, aug). Research on Yaw Error Detection of Wind Turbine Based on Particle Swarm Optimization. *Journal of Physics: Conference Series*, *2005*(1), 012213. (https://doi.org/10.1088/1742-6596/2005/1/012213)

Appendices

Appendix A

Weather observations

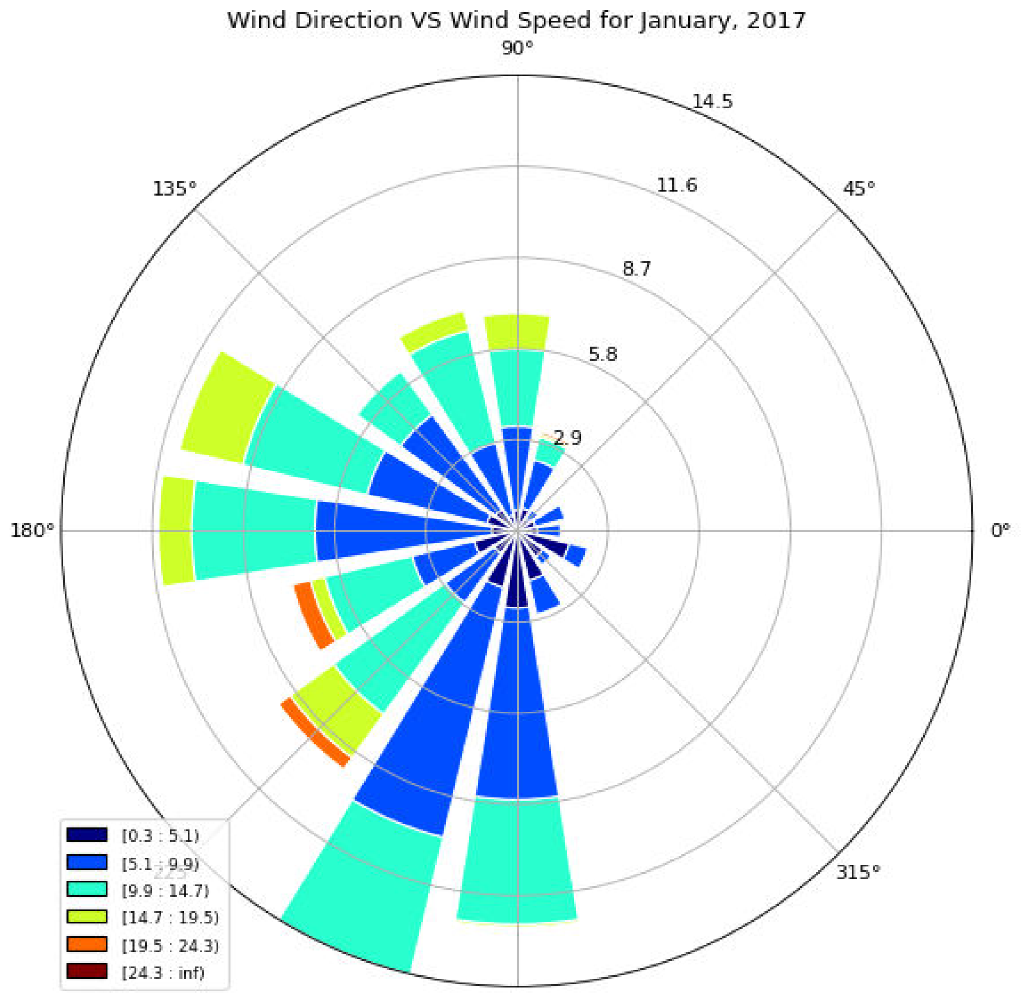


Figure A.1: Plot of Wind speed vs Wind direction data collected at the station for January 2017.

Appendix B

Feature Selection

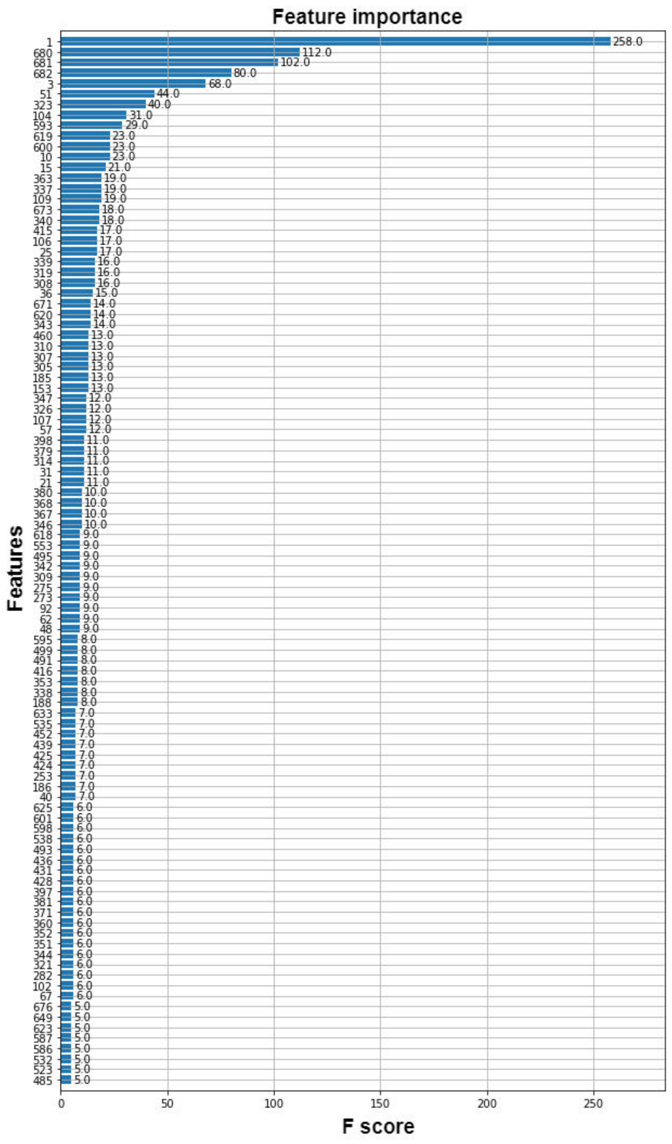


Figure B.1: Plot with the resulting 99 features selected with XGBoost Regressor model

Appendix C

Time Series Data Components

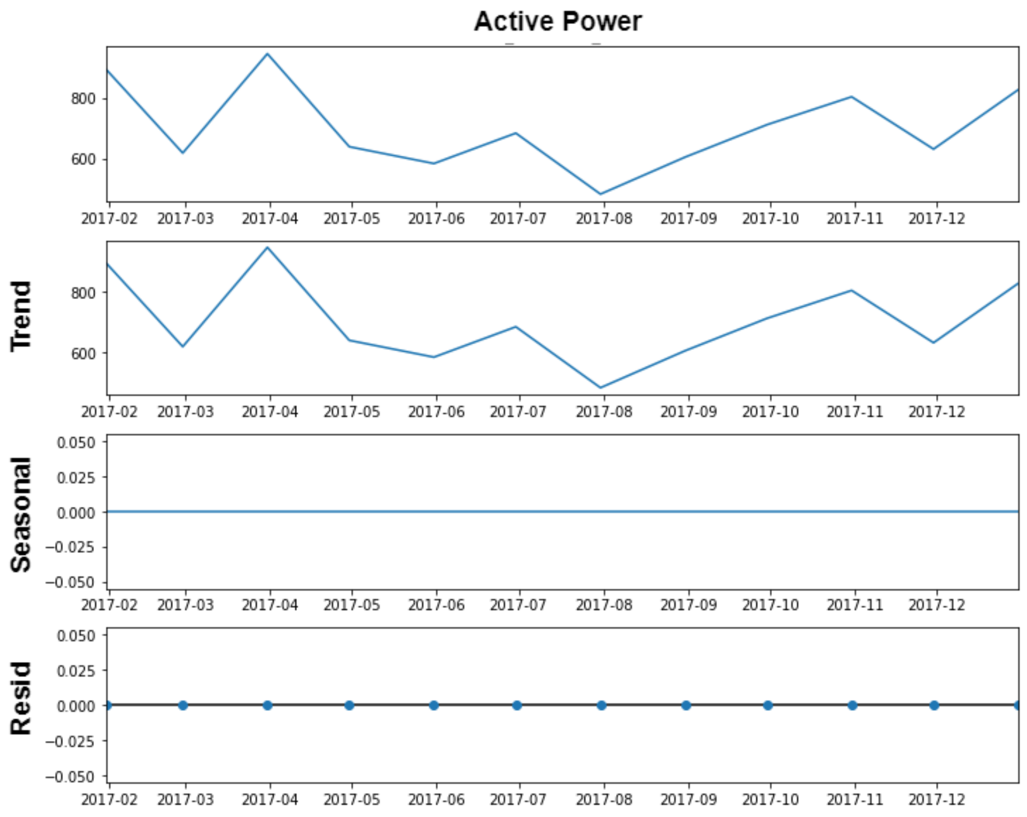


Figure C.1: The decomposition plot of Active power for 2017 with seasonality, trend and residuals plotted separately.

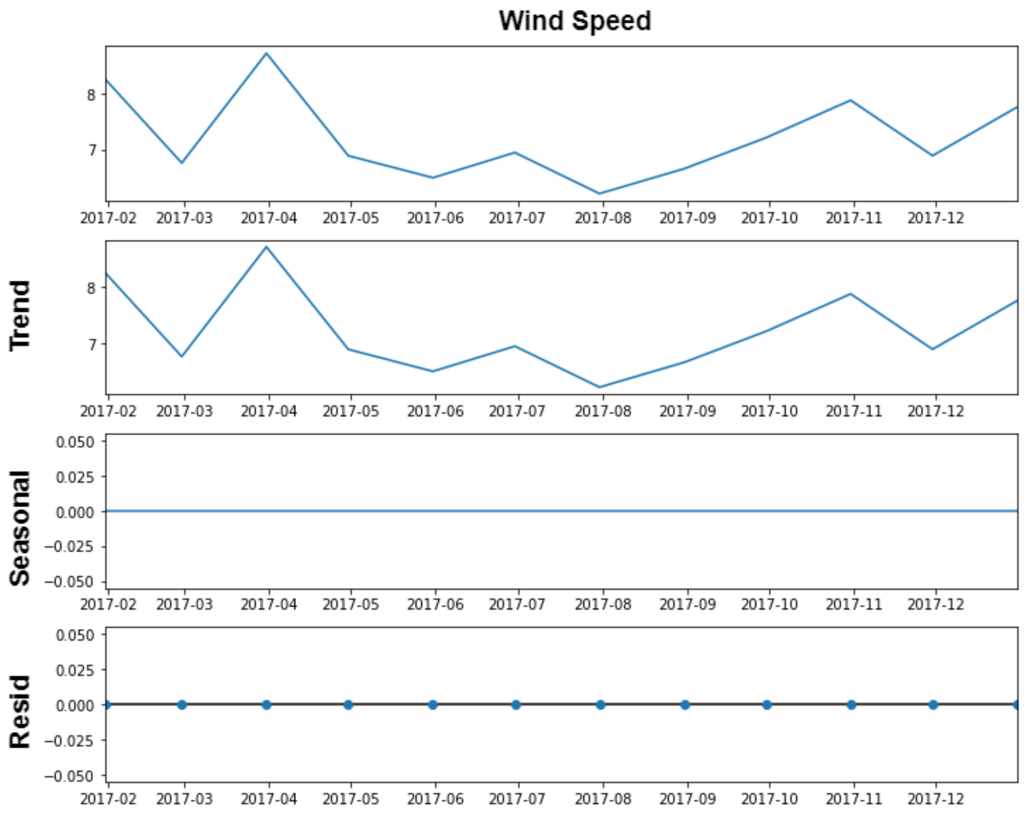


Figure C.2: The decomposition plot of Wind Speed for 2017 with seasonality, trend and residuals plotted separately.

Appendix D

LSTM Models Learning Curves

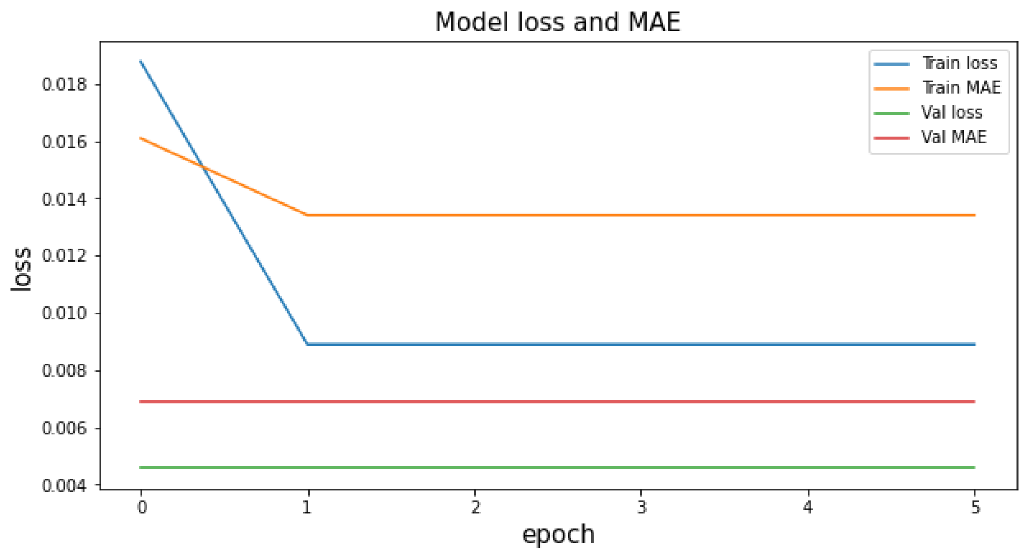


Figure D.1: Learning curve for model with all features , 10 hours for both target probability distribution and for window length. Missing values were imputed.

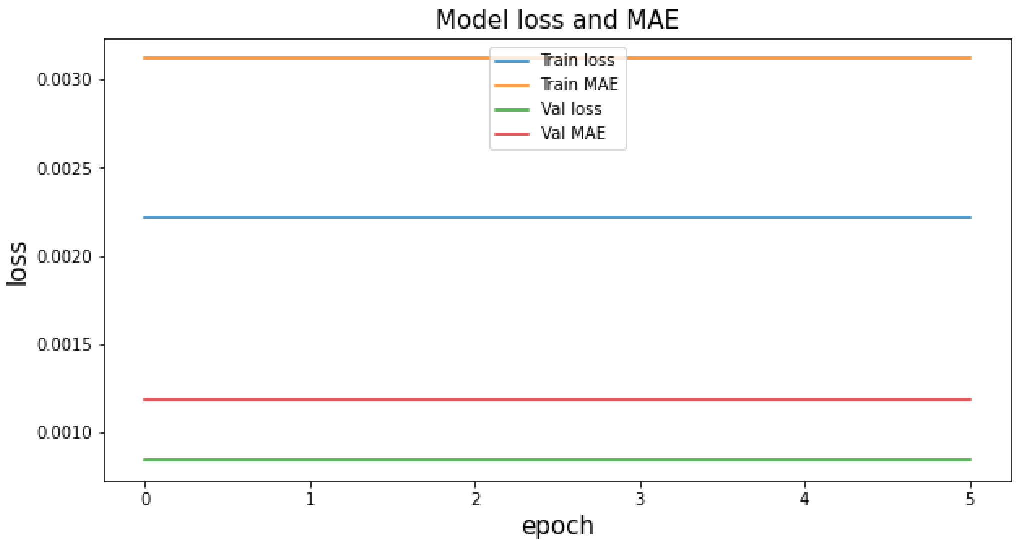


Figure D.2: Learning curve for model with all features , 2 hours for target probability distribution and 24 hours for window length. Missing values were imputed.

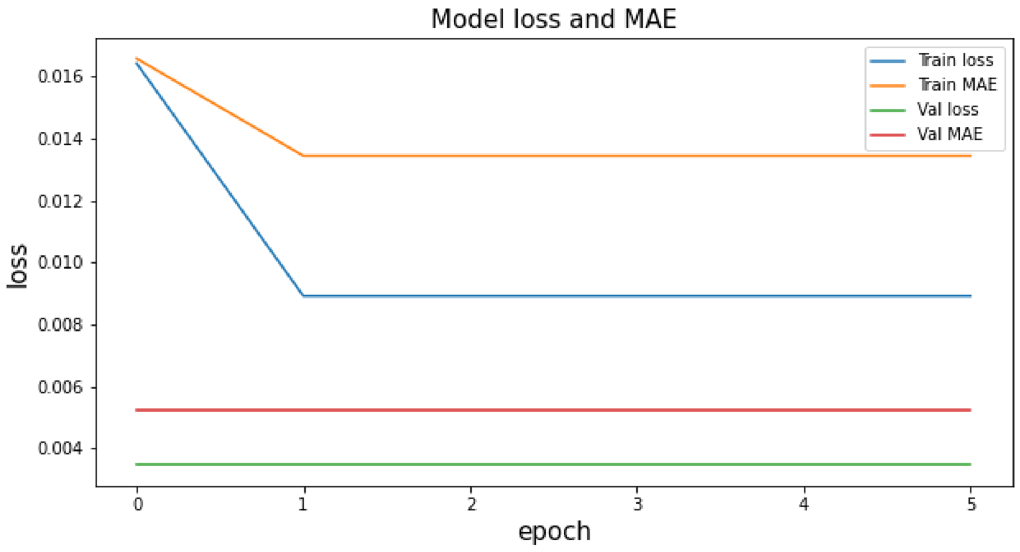


Figure D.3: Learning curve for model with all features , 10 hours for target probability distribution and 24 hours for window length. Missing values were imputed. Plot is expressing some signs of initial learning.

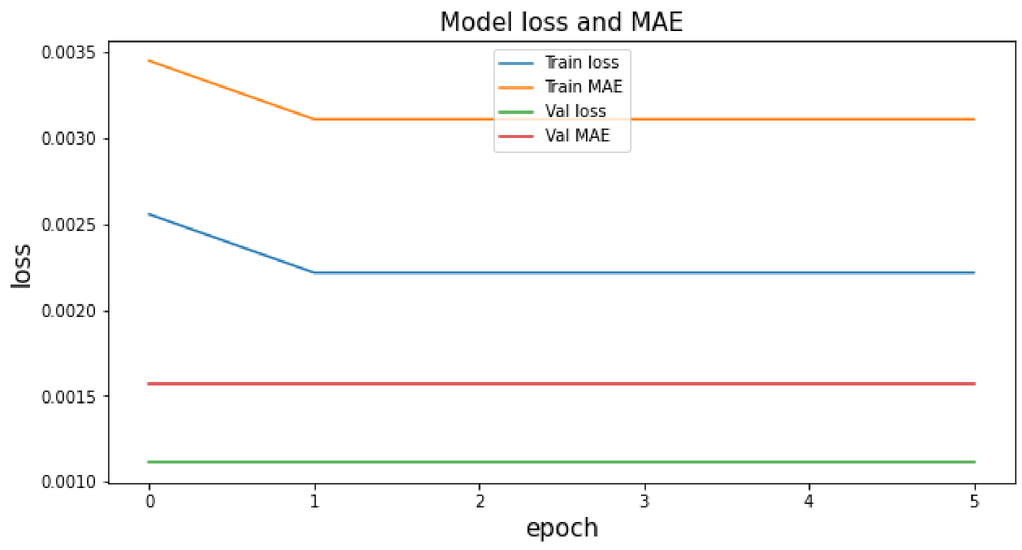


Figure D.4: Learning curve for model with all features , 2 hours for target probability distribution and 10 hours for window length. Missing values were imputed. Plot is expressing some signs of initial learning.