Norwegian University
of Life Sciences

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

Optimising maintenance operations in photovoltaic solar plants using data analysis for predictive maintenance

Gøran Sildnes Gedde-Dahl
Data Science

Acknowledgements

An acknowledgement from me is that I have been working as a consultant on IT systems for Scatec, namely EAM (Enterprise Asset Management) system and corresponding analysis, while working on this thesis. The work is done through my employer, Prevas AS. Despite this, I have tried to perform an objective analysis of the problem at hand.

I want to express gratitude to my supervisor Stefan Schrunner, providing good guidance and interesting insight on the topics we have discussed. The many discussions about the thesis and the different approaches has been of great educational value for me, as well as value of quality for the thesis. I have really appreciated having you as my supervisor.

I would like to thank Scatec ASA for providing me the opportunity to do this thesis on their solar plant, using their data and use case. Predictive maintenance is a topic I am largely interested in, and Scatec has allowed me to pursue this to the fullest.
I would also like to extend my thanks to Heiko Birkholz, Kristian Hauff, Halvard Haug, Zeinab Ramadan and Meron Haile Tesfasion for the good discussions and input during the thesis work.

Over the last three years, I have had the opportunity to pursue my dream of studying Data Science in addition to working. For this, I want to express my gratitude towards my employer, Prevas AS. This is an example of high end people management. A special thanks to my boss, Alex, for making this possible, as well as Tom and Bjørn for making the workload manageable.

Lastly, a big thank you to my partner, Hannah, for supporting me during the thesis work, keeping me sane, fed and healthy - especially during the last hectic month. A big thanks to my family and friends; Especially my mother and father for the support, as well as their teaching and motivation for me to work hard to achieve my goals.

Gøran Sildnes Gedde-Dahl
Ås, May 16th 2022

Abstract

In PV (photovoltaic) solar power plants, high reliability of critical assets must be ensured—these include inverters, which combine the power from multiple solar cell modules. While avoiding unexpected failures and downtime, maintenance schedules aim to take advantage of the full equipment lifetime. Predictive maintenance schedules trigger maintenance actions by modelling the current equipment condition and the time until a particular failure type occurs, known as residual useful lifetime (RUL). However, predicting the RUL of an equipment is complex in this case since the equipment condition is not directly measurable; it is affected by numerous error types with corresponding influencing factors. This work compares statistical and machine learning models using sensor and weather data for the purpose of optimising maintenance decisions. Our methods allow the user to perform maintenance before failure occurs and hence, contribute to maximising reliability.

We present two distinct data handling and analysis pipelines for predictive maintenance: The first method is based on a Hidden Markov Model, which estimates the degree of degradation on a discrete scale of latent states. The multivariate input time series is transformed using PCA to reduce dimensionality. This approach delivers a profound statistical model providing insight into the temporal dynamics of the degradation process. The second method pursues a machine learning approach by using a Random Forest Regression algorithm, on top of a feature selection step from time series data. Both methods are assessed by their abilities to predict the RUL from a random point in time prior to failure. The machine learning approach is able to exploit its favourable properties in high-dimensional input data and delivers high predictive performance. Further, we discuss qualitative aspects, such as the interpretability of model parameters and results. Both approaches are benchmarked and compared to one another. We conclude that both approaches have practical merits and may contribute to more favourable decisions and optimised maintenance operations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the recent years, climate change has lead to an increased focus on the human's impact on the environment. As a consequence, large scale plans have been rolled out to minimise environmental pollution, and several countries have committed to reducing carbon emissions in the Paris agreement [2]. A key to achieve the goals in this agreement, is replacing fossil energy sources with renewable energy sources [3]. In practice, this means that the demand for electrical energy produced in a non-polluting way is growing [4]. To meet this increased demand, effective and reliable methods of harvesting energy must be used.

One of the largest sources of energy accessible to humans is the sun. With almost 4 million exajoules reaching the surface of the earth, and 50000 of them being easily harvested, this is a great potential to be exploited [3]. The potential of harvesting solar energy is widely perceived, hence the installed capacity of photovoltaic (PV) solar power has increased 7.7 times since 2012 [5]. The increased capacity indicate large growth in this industry, due to improved technology and demand for "green" power. Combined, these factors have allowed new companies to start producing solar power. An example of such a company is Scatec ASA.

Scatec ASA was founded in 2007 as a Norwegian startup company in solar power, having its first installation in Italy in 2009. Since then, their business has expanded to numerous countries across almost all continents of the world. As of today, their global portfolio contains 1894 MW in solar power [6] for grid production, and the number is growing. Scatec also pursues projects in remote industrial sites not being able to connect to the grid. Across all renewable energy sources, the company has a capacity of 3.5 GW in operation and construction across four continents. Thereof. 380 MW of this originate from the Benban solar plant in Egypt [6], the largest solar power source in Scatec's portfolio, split across six sub-plants.

1.1 Background

While the capacity of renewable energy is increasing, there are requirements to the quality and reliability of the power produced in the power grid. In order to comply with grid regulations, a certain amount of energy needs to be produced at a given time. The reason for such regulations is to contain the power frequency of the grid. Containing the frequency is important in the process of providing the consumers with power that is safe to use in their devices. If the frequency deviates too much from the target value for the grid, it can have cause deprecation of electric devices, as well as hazard in using them [4]. As a consequence, strict rules apply for producers of power to the grid [7], and oblige them to monitor and optimise the reliability

of their production assets.

Despite high quality standards and good routines for commissioning, the probability for failure in production equipment is present at all times during operation [8] [9]. Against all precautions, random production equipment failures are possible and in general hard to predict [8]. The distribution of failure as a function of equipment lifetime can be seen in Figure 1.1. The Figure's phase 1 shows that the distribution of failure is high in the beginning of equipment lifetime, known as burn in. The high failure rate in this phase is caused by faulty mounting, production error in the equipment, incompatibility, etc. [8]. Phase 2 displays that there is a constant failure rate once the equipment has passed the burn in phase. Phase 3 shows that the equipment reaches a state of wear-out towards the end of the lifetime. In this phase, the failure rate increase exponentially.

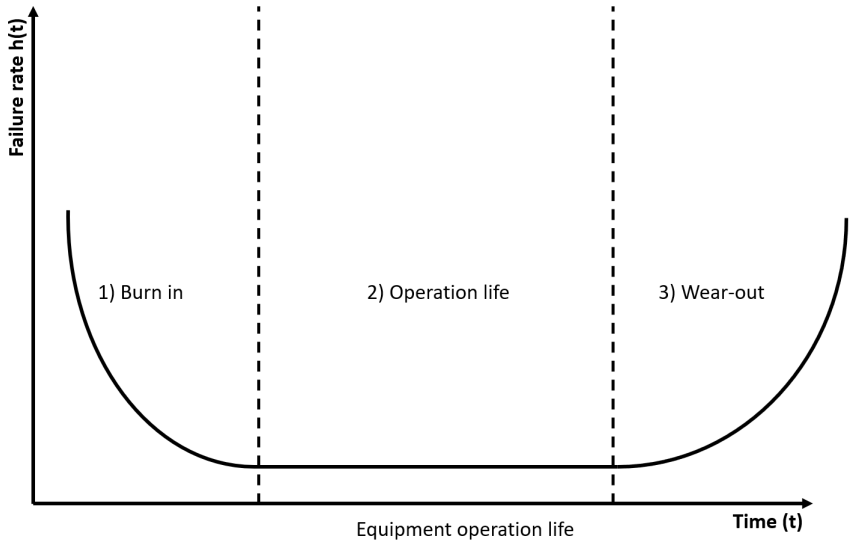

Figure 1.1: The Weibull bathtub curve, showing the rate of failure for electrical equipment as a function of life time. Phase 1 is the Burn in when the equipment is newly installed. Phase 2 is the equipment Operation life, where the equipment is running properly and fine-tuned. Phase 3 is when the equipment is becoming of age, and starts to get worn out.

1.1.1 Scatec photovoltaic solar power plants

Scatec uses PV (photovoltaic) technology in solar power plants. PV is one out of three main technologies in the solar power industry, and by far the most used [10]. Summarised, the PV technology works by the illumination of modules consisting of positively and negatively charged semiconductor materials in cells. In turn, the illumination of the cells leads to electron movement, giving electric energy. The modules are connected in serial, in strings. Strings can either be connected directly to a string inverter or in parallel to form an array; which is connected to a central inverter. For the Benban plant, central inverters are in use. The inverters turn DC power produced by the solar cells into AC power, compatible with the Grid. Central inverters are mostly used in utility-scaled power plants [11]. One of the downsides to this approach is that it makes the consequence of inverter failure quite large; if the inverter has downtime, the number of modules affected will be much larger than with the string inverter approach. Thus, the operation of inverters is highly critical for power production, which is a good reason for close monitoring of these components.

1.1.2 Maintenance

Failures have potential to cause a lot of damage and loss; especially if they happen unexpectedly. A severe failure in equipment can have large consequences both in terms of HSE (Health, Environment and Safety) and economy. The HSE perspective is triggered by the risk of using or being near a piece of malfunctioning equipment. In the case of power plants, electrical hazards may occur. As for the economic perspective, such failures can take a long time to repair, leading to production loss and occupation of personnel resources. The maintenance organisation will have to find necessary spare parts and mobilise personnel, which is considered to be a significant part of repair time in maintenance [9].

Maintenance is defined as any work done to prevent or correct failure [9]. In general, one distinguishes between two main categories; preventive and corrective maintenance. The difference between them is that preventive maintenance is done prior to failure; in order to prevent it, whereas corrective maintenance is done as a consequence of failure, in order to restore the intended function of the component [9] [12]. Preventive maintenance is often considered the better option of the two when unexpected failure and downtime has a large consequence.

Preventive Maintenance has two main approaches: Periodic preventive maintenance and condition based maintenance. Periodic preventive maintenance is done based on an interval in time or use. For instance, changing engine oil and oil filters on a car every year, or every 5000km driven; The first instance is calendar based, and the latter is usage based [9]. Condition based maintenance, as the word suggests, involves monitoring the condition of the equipment, and performing maintenance when needed [9] [8]. The monitoring can be done both by manual inspection or installing sensors [13]. Predictive maintenance falls under this category, and differs from the condition based maintenance in that it estimates the time until a failure will occur. The hierarchy of maintenance can be seen in Figure 1.2.

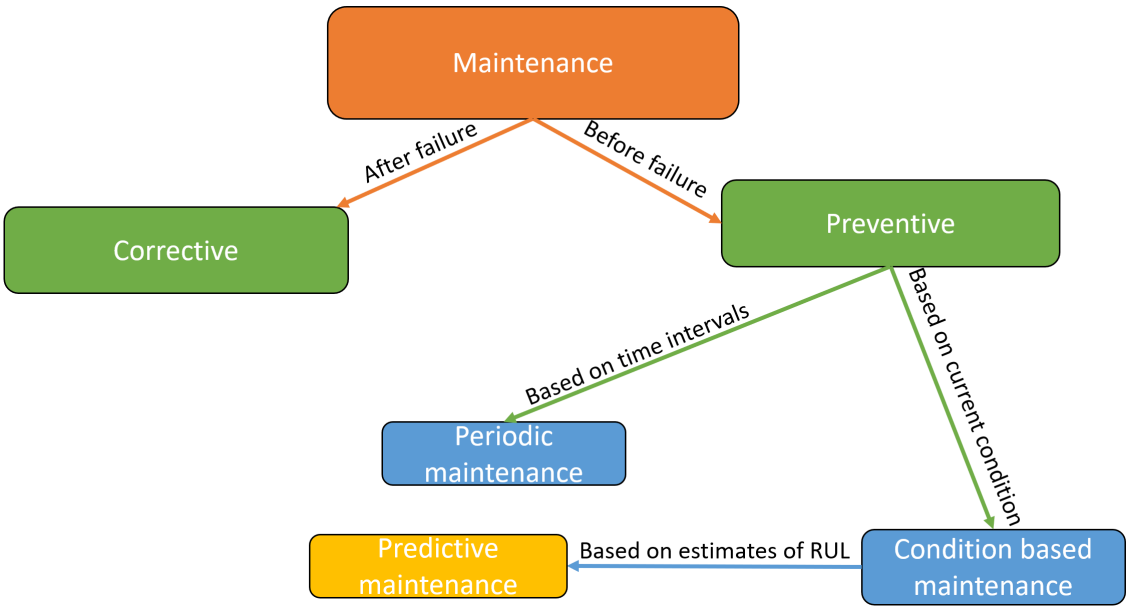

Figure 1.2: The maintenance hierarchy, showing the various approaches for maintenance, on their timing and decision base.

Unplanned corrective maintenance activities generally has a large negative effect on efficiency and productivity [14] [15]. During all the reaction and repair time, the equipment will not be in operation, leading to lost potential production. The impact of a failure will be multiplied

if multiple equipment or multiple parts of equipment are harmed when it occurs. In a preventive maintenance activity the possibility for planning is better. For planned activities, the time to repair is significantly lower [9], and can, in the case of solar power, take place when there is no potential to produce power (e.g. at night). Such advantages are the reasons why organisations choose a preventive maintenance approach for their critical equipment. Performing periodic maintenance actions to prevent failure is the most common strategy.

The key principle of the periodic preventive maintenance approach is to replace parts on equipment after predefined time intervals, which is less costly than letting the equipment fail at an inconvenient time [8]. However, these intervals leads to parts being changed potentially at an early point, leaving a large part of their lifetime unused. The difference between interval length and potential lifetime is known as unused lifetime, and is a large cost factor in periodic maintenance [16] [9] [17]. Imagine changing the engine of a car every two years, when the potential lifetime of the engine is four years, leading to the maintenance being twice as expensive as necessary. Such replacements are wasteful, and not sustainable when it comes to both human and material resources. Authors of [18] claim that 30% of periodic preventive maintenance is necessary, another 30% is wasteful, and 30% more is directly harmful. The latter is due to the increased failure rate in the period after the installation of new equipment or parts [9].

To avoid unnecessary or even harmful periodic activities, some companies are performing maintenance based on the condition of the equipment or parts. The condition of a part or equipment can be estimated through manual inspection done by people or measurements done by sensors surveying the equipment. As manual inspection can be less accurate due to human error [19] and lack of detail and consistency in measurement [9], sensor values are considered the best option for most applications [9]. When these sensor values are used or analysed to estimate RUL (residual useful lifetime) of equipment, it is known as predictive maintenance. Predictive maintenance is made possible by technological advances in computational methods and hardware for data analysis. Such RUL analysis can be done in three ways; Predicting failure based on values exceeding thresholds [20], mathematical models - model-driven or through data analysis - data-driven. The latter is often done by means of machine learning algorithms and expert knowledge.

Deciding between periodic, corrective and predictive maintenance is a key decision when it comes to saving costs. Predictive maintenance may not be feasible if the consequence of failure is small, therefore does not justify the procurement of condition monitoring equipment. In this case, one can use the run-to-failure approach involving corrective maintenance, repairing the equipment only when it fails. Other cases disfavouring predictive maintenance is if failure happens at regular intervals, or if the replacement of parts has a low cost. In this case it is easy to perform periodic maintenance at the correct time prior to failure.

For solar power, predictive maintenance has a large, unused potential [21]. Today, a common mindset is to "install and forget" plants [21] - meaning that the plants are maintained to a low degree, due to the high reliability of their components. This mindset leads to unexpected failures and downtime, in turn leading to lost production, increased cost and reduced power quality. Based on reports from pilot projects in solar plants, significant improvements can be made in solar plants if predictive maintenance is implemented [22]:

- Costs can be reduced by 12%
- Uptime can increase by 9%

- Risks for HSE and quality can be reduced by 14%
- Lifetime of assets increase by 20%

For energy producers in the solar power market like Scatec, predictive can be a very good investment.

1.1.3 Predictive maintenance

The occurrence of papers related to predictive maintenance have increased significantly over the last years. According to these, there are two main types of predictive maintenance. The first type is model based predictive maintenance, which is mainly based on mathematical functions describing the equipment degradation [23]. An example of such an algorithm is Archard's wear law, being used by the authors of [24], among others. The law is based on volume loss due to the normal force, force of contact, sliding distance and a wear constant. In order to set up the model properly and achieve accuracy in the model based method, one needs access to a high level of domain knowledge and observations of failure [25]. Since profound expert knowledge is often unavailable, the model is not applicable in many cases. The second type of predictive maintenance is data driven predictive maintenance. Here, data from equipment is analysed with machine learning or statistical methods. The objective of these is to find measurement variables correlating with degradation or indicating impending failure [13] [26]. Data is most commonly sampled by sensors, but can also be read manually. Two important aspects for data-driven predictive maintenance is to monitor the equipment as it runs until failure (also known as Run-To-Failure (RTF) approach [27], presented in chapter 1.1.2), and to measure the correct parameters prior to failure [26]. The focus of this work will be on the data-driven approach.

A divided approach between the work done on data driven predictive maintenance on sensor data, is how the failure prediction problem is solved in a machine learning manner; some choose to do a binary failure/no failure classification problem [28] [29] [12] [30] [31] [32] [33]. For these cases, a classification algorithm is trained to recognise failure. Using this approach, the algorithm will return e.g. 0 when no failure is present, and 1 if failure is present. An alternative approach is to solve the failure prediction problem as a regression problem - estimating the level of degradation and the RUL (Residual Useful Lifetime) [13] [34] [8] [16] [35] [36] [37]. A regression algorithm is then trained on the data to estimate the degradation or RUL with a number.

The approach used by the author of [13] uses a Random Survival forest to group individual components with similar degradation profiles and estimate the reliability function of these. Then, a LSTM (Long Term Short-Memory) RNN (Recurrent Neural Network) is used to estimate the state of health on the equipment. RNN's are also used by the authors of [35], where they are compared to an approach with a Deep Belief Network feeding data into a Feed Forward Network for prediction of RUL. The authors of [34] test several regressors; Linear Regression, Random Forest Regression and Symbolic Regression to estimate RUL. Their work presents that the prediction quality of the algorithms increase as the RUL approach 0.
The Random Forest based methods is also used by the authors of [36], using Random Forests with feed-forward-propagation, compared to ANN's (Artificial Neural Networks). A slightly different approach compared to the above is used by the authors of [8], using a statistical method - ARMA (Auto Regressive Moving Average) to predict the future development of a

parameter, and then using a General Linear Model to predict the RUL based on the output from ARMA.

1.1.4 Predictive maintenance in photovoltaic solar power plants

There are numerous works related to predictive maintenance in PV plants. Several approaches are examined and reviewed [21]:

- Manual diagnostics - Manual inspections done on site [38] [39]
- FMEA - maintenance based on equipment criticality [40] [41]
- Machine learning and forecasting based on manual measurements [42], [43]
- Real-Time Sensors monitoring the values in the system [44] [45] [46]

Arguably, some of the points above are not to be considered predictive, as they do not include analysis of the data and RUL estimation. In fact, merely a minority of the papers discussed in the literature study [21] actually fit into the definition of predictive maintenance according to Section 1.1.2 - the majority tackles fault detection and monitoring, and other factors concerning PV plants. As this thesis aims to estimate the RUL, these are less relevant here. However, a relevant piece of information in these papers is provided by the parameters monitored in the implementations.
The authors of [47] monitor the following parameters: current, power, module temperature, solar radiance and losses. Combined, these parameters indicate whether it is likely that a failure occurred or not. The authors of [48] use the I-V curve, comparing the current and voltage with regards to the power output. The I-V curve analysis is a method mostly used in failure detection for modules, as different failure types tend to have different characteristics on this curve. A similar approach is taken by the authors of [28].
Meanwhile, authors of [49] only use current and temperature to model the potential output, and compare to the actual output. Thereby, a gap between potential and actual output can indicate an error. The approach in the mentioned paper use sensor values with one hour interval. On the contrary, authors of [50] claim that the sample frequency should be on 5-15 minutes for failure detection. Authors of [51] suggest that one hour interval for measurements actually gives a better accuracy in fault detection than 10 minute sample interval.
The differences in conclusions between these papers can be explained by the fact that predictive maintenance and anomaly detection problems are quite variable from different domains, applications and geographical locations [26]; even solar plants with the same technology can have different failure patterns due to local variables geographically, human factors, etc.

1.1.5 Data-driven operation and maintenance in Scatec

In 2021, a PhD thesis was published at the University of Oslo in cooperation with Scatec, using the same data, which will be used in this thesis [1]. The PhD thesis presents ways to filter out irrelevant data for fault detection on different type of equipment, so that the data used for analysis is as relevant as possible.
Figure 1.3 provides an overview on inverter data, which is suggested to be filtered out in [1], divided into 3 categories.

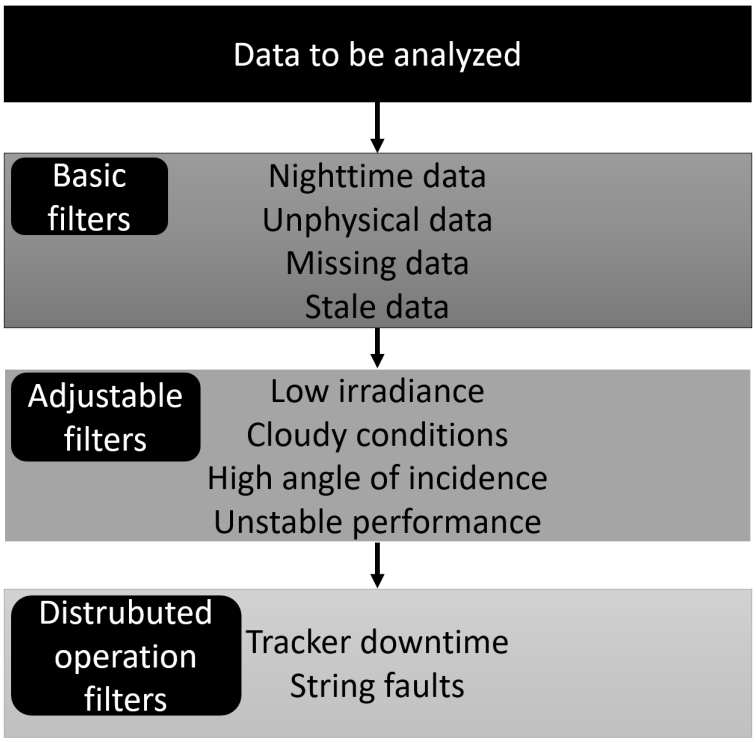

Figure 1.3: Preprocessing and data filtering pipeline suggested by [1]

The basic filters contains the most obvious data to filter out: Nighttime data without power production, nonphysical data (obvious erroneous sensor measurements), data missing and stale - outdated data.
Adjustable filters are to be adjusted for the use. Low irradiance, cloudy conditions and high angle of incidence are related to the weather conditions. Unstable performance is related to other failures happening.
The distributed operation filters are based on the operation on equipment connected to the inverter to analyse. A failure in these could affect the input of the inverter and be misinterpreted as a failure in the inverter.
In addition to suggesting data to be filtered out for detecting failure, the thesis also discussed the importance of dividing soiling from degradation in online fault detection. The main difference is that soiling is caused by accumulation of dust on the solar panels, while degradation means that the equipment is affected by wear-out.
Statistical methods are used to distinguish between external factors and systematic errors such as soiling and seasonal changes from degradation. As well as the statistical analysis of electrical data, imaging and are also used to detect failure and degradation in equipment [1].
The PhD thesis presents the following aspects:

- Ways to assure data quality for different analysis purposes,
- A setup for string-level fault detection based on data from one of the solar plant, using numerical tools
- Correlation between hot-spots and performance in solar panels
- A proof of concept combined soiling and degradation detection algorithm based on a Kalman Filter, STL.model (Seasonal- and Trend decompositioning using Locally estimated scatterplot smoothing) and a year-on-year-model. This model also estimates degradation rates of solar modules.

However, the PhD thesis does not investigate and exploit the full potential of statistical and machine learning models for predictive maintenance.

Another master thesis done in cooperation with Scatec, [11], discusses the use of machine learning algorithms to detect power loss and abnormal behaviour in solar plants, based on operational data. The thesis use inverter data combined with weather data and string data. Predicted power production based on weather conditions are compared to actual power production in current and voltage to detect loss and events when the deviance between potential and actual power production reaches a certain threshold. The loss and events detected originated from the modules and strings connected to the inverter. The author of [11] also presents machine learning algorithms being able to detect strings with decreased performance. Several regression algorithms are used in the thesis to estimate the predicted power production based on solar conditions. In this way, large deviance could be detected and raised as potential loss or decreased performance.

The work of this thesis is related to the foundation laid and have some similarity with the data structure in the work done by the authors of [11] and [1].
On the other hand, these works are more focused on detecting events and losses that has already occurred - except for the solar module degradation rate model from the author of [1].

1.2 Data infrastructure

With the transition towards Industry 4.0, more companies see large value in digitisation and make large investments in this area. Scatec is no exception, and have put large amounts of resources towards digitisation in operation and maintenance.

Maintenance data

For asset management, work planning and spare part management, Scatec has implemented a cloud based EAM (Enterprise Asset Management) system from Hexagon, called HxGn EAM. This system provides good opportunities for customisation and data exchange, and has been in use since 2019. The information available in an EAM system includes historic events and work done on equipment. If equipment fails, the technicians are to select standard codes for the following:

- Problem code - Symptom of failure
- Failure code - How the equipment has failed
- Cause code - The cause of the failure
- Action code - The actions performed to correct the failure

Failure events are known in the system as work orders, and are separated by types and codes. An example case of the system, is if an inverter fails. Here, a work order for repairing the inverter will be created, and assigned to a technician. The technician will change the status of the work order to "In progress", and complete the work. When finished, the technician will register the time spent to correct the failure, parts that have been used, as well as the closing codes. The closing codes are structured in a hierarchy, so if the technician selects

FAIL - failure as problem code, the cause code will be restricted based on that choice. The failure code can for instance be INV-IGBT - Power Block IGBT, leading to the options for cause code are restricted only to those related to internal failures. A potential cause code in this case could be high temperature. In the same manner, the action codes are restricted based on the choice of cause code. REPL - Replacement can be an action code for this example case.

Operational data

Scatec has a regional process monitoring and control centre for each continent, called CMC. This centre has the overall responsibility of monitoring plant activity and system status, as well as alarms on monitored equipment. The monitoring is done in a SCADA (Supervisory Control And Data Acquisition) system, a database system for monitoring operation in the plants. Operation is monitored using live measures from equipment in operation, such as voltage, temperature, operational status, etc. Weather conditions are also monitored; temperature and radiance shows the conditions for producing solar power and can assist in decision-making. Combined, SCADA-data can assist in condition monitoring, as they are thoroughly captured from several aspects of the equipment. Therefore, this data will be the key input to the algorithms for predictive maintenance in this thesis. All data from sensors in the plant are stored and visible in the SCADA system in real-time, and historically in a multivariate dataset with 1 minute resolution. Alarms are raised in this system based on threshold values set on sensor values. When such an alarm occurs, the CMC operator can escalate the alarm into a work order in the EAM system through an integration between the two systems. This integration was implemented in the middle of 2020, and allows for tracking where and when a failure has occurred.

1.2.1 Data acquisition

A significant part of work done for the purpose of predictive maintenance is the data acquisition; failure prediction is not feasible without monitoring the correct aspects on the equipment. Therefore, the idea of predictive maintenance should be considered already on the drawing board when setting up sensors [52]. A good way to ensure that the right parameters are chosen to be monitored, is to perform an FMEA (Failure Mode and Effect Analysis) of the equipment. An FMEA will return which failures can happen, and what their consequence will be. From there, one can determine relevant parameters from experience and thereby help indicate the development of the failures one would like to predict [20]. The data needs to be recorded at a sufficient quality and fit into the system, like described in the ISO 8000-8 standard. In this standard, there are three aspects of data quality to take into account:

- **Syntactic quality**: How well does the data fit into the data system? This involves whether or not it fulfils technical requirements.
- **Semantic quality**: How representative the data is for what it is measuring? Does it correspond with the real world?
- **Pragmatic quality**: Can the data be used for the desired objective?

A strength in the data acquisition from the SCADA system used in this thesis, is that each measurement is recorded with a quality index indicating whether the sample is reliable

or not. The quality index is triggered by the sensor status, but it is no warranty against measurement errors. The index can be used to filter out some extreme data.

1.3 Objective

The objective of this thesis is to combine data from the SCADA-system and weather stations, as input for data-based predictive maintenance models. In particular, the work aims to make use of two approaches:

- One machine learning regression algorithm based on a Random Forest Regressor trained on output from measurement variable engineering using the TSFRESH method
- One statistical approach using an HMM (Hidden Markov Model) will be used to estimate the hidden state - degree of degradation on the equipment, based on the output from a PCA (Principal Component Analysis) dimensionality reduction.

After developing the necessary concepts and implementations, the goal is to compare both approaches with each other from both, a quantitative and a qualitative viewpoint. Experimental evaluations are based on data collected from inverters in the Benban plant in Egypt, restricted to a single failure type. The methods will be evaluated by relevant metrics on their output, defining the quality of prediction. A common benchmark process for comparing the two approaches will be presented. Domain experts were consulted regarding which features they believed to be important to monitor in order to forecast the error in question for this thesis, but there was little knowledge about such features.

The ultimate goal for both approaches is to estimate the current condition on inverters, with respect to their risk of failure. The risk of failure will be for one agreed-upon failure type. Transferring to a predictive maintenance approach for the inverters will reduce the impact of failure on production and reliability in the solar plant. An endpoint of such an algorithm, is the ability to pinpoint the measurement variables which are correlating with the RUL. This will help the plant personnel to interpret the parameters as an indication of which condition the equipment is in. The two methods used for this purpose can be re-used for predictive maintenance in other domains, using time series data.

In a nutshell, the contribution from this thesis is to deliver a comparison between machine learning and statistical methods in predictive maintenance. The results from both methods are considered as a proof of concept for the proposed predictive maintenance frameworks in the solar industry - with actual prediction of RUL. This thesis differs in the work described in Section 1.1.5 in that the objective here is to estimate the RUL of a given type, not detect an occurred failure of an unknown type. This thesis also looks into both statistical and machine learning models trained on more comprehensive data from the inverters, as opposed to the previous two being more focused on strings and modules. The use of the methods used in this thesis could benefit the operation and maintenance in a different way and optimise the way of handling failures of the decided type.

1.4 Information security and infrastructure

IT infrastructure is an important aspect in data science projects. Firstly, sensitive data must be safe - only intended users must have access. The dataset used in this thesis is sensitive, as it gives details about Scatec's operation and power production.

Source .csv files containing raw data from the inverters and weather station was stored in a secure OneDrive location, provided by NMBU. This location is authenticated with MFA (Multi-Factor Authentication), and is to be considered secure. From the OneDrive location, they were imported to a local, password protected SQL (Structured Query Language) database on the author's computer, using Python files. In these Python files, the data was re-sampled to the desired resolution (30 minute intervals). In the SQL database, data from the various sources was merged into views containing only data columns relevant for analysis.

For analysis, a local Jupyter notebook[1] was used, along with packages to import data from the SQL database. Here, data engineering and filtering was performed prior to preprocessing and analysis. No passwords for the SQL database was stored in any Python files, to protect the integrity of the passwords. A flow chart showing the data flow and infrastructure can be seen in Figure 1.4, showing how the data is fetched from the SCADA cloud, stored in a secure OneDrive location, then fetched and stored in a local database. From there, it was fetched for analysis in a jupyter notebook file . The dotted line illustrates which parts of the process were performed on the author's local computer.

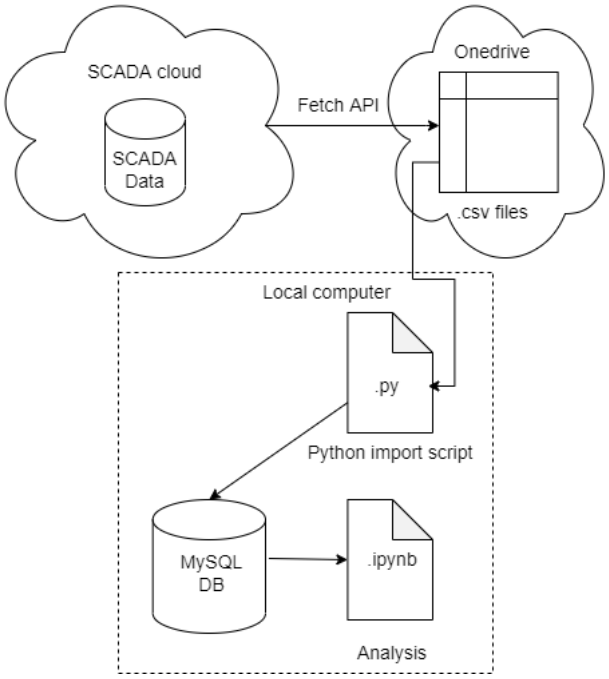

Figure 1.4: Overview on data storage and flows.

[1]The relevant files used for analysis can be found in https://github.com/Goransg/MasterThesis.

Chapter 2

Methods

This chapter presents the methods applied in the thesis. The first Section 2.1, introduces our problem setup mathematically. Afterwards, one section is dedicated to each of the two main modelling pipelines investigated in this work.

2.1 Problem setup

In the following, we assume m maintenance cycles of electrical devices as model input: each maintenance cycle $i \in \{1, \ldots, m\}$ is defined as a particular device tracked over a time window $\{1, \ldots, T_i\}$. The interval starts at $t = 1$, denoting the first time point after the previous maintenance action or downtime, and ends at $t = T_i$, denoting the time of the subsequent failure. The index denotes a regular (discrete) time step of 30 minutes. According to our model setup, we restrict to maintenance cycles, which are terminated by one particular error type. Further, we treat all maintenance cycles as independent and identically distributed, and hence, as random samples.

For each maintenance cycle, we sample a randomised integer $p_i \in \{T_i - T_s, \ldots, T_i\}$, where T_s is a user defined input to determine how close to the end of the maintenance cycle p_i could possibly be. The objective of sampling p_i is to simulate the current point in time, where the prediction should be made. As model input, we observe a multivariate time series of historical sensor measurements up to time p_i, given as $\boldsymbol{x}_t^{(i)} \in \mathbb{R}^n$, where n is the number of sensors, and $t \in \{1, \ldots, p_i\}$. The goal is to predict the residual useful lifetimes, $\text{RUL}_i = T_i - p_i$ given time series $\boldsymbol{x}_t^{(i)}$. We denote our prediction by $\hat{y}_i$, while the actual time until failure (ground truth) is given by $y_i = \text{RUL}_i$. When evaluating predicted RULs, the deviation between the predicted value and the ground truth is calculated with three metrics: RMSE (Root Mean Squared Error), MSE (Mean Squared Error) and R^2. For a test dataset with m observations, the measures are calculated as [53]:

- $\text{MSE} = \frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)$
- $\text{RMSE} = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)}$
- $R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)}{\sum_i (y_i - \bar{y}_i)}$

Since predictive maintenance requires an assessment of the risk of failure rather than an exact RUL estimate, another metric is introduced to benchmark both approaches against

each other. Thus, predictions are grouped into three risk classes g_{1-3}, based on the predicted hidden state for HMM (Hidden Markov Model, presented in Section 2.2.2) and the predicted RUL from the RFR (Random Forest Regressor, presented in Section 2.3.2).
The same split was performed on the ground truth test data, so that the result of both algorithms can be evaluated by the same evaluation metric.
Based on the prediction group, the F1-score were calculated for the test data for each of the methods. The accuracy (micro precision) metric was used for this purpose, as it is suited for multi-class problems. The F1-score is calculated as follows:
$F1 = \frac{\sum_{g=1}^{3} TP_g}{m}$, where TP_g is the number of true positives predicted for the class g [54].

For the analysis in this thesis, the parameters were set to: $T_s = 50$. This means that the maintenance cycles were truncated; cut off at a random point between 50 days prior to failure and one day before failure. Every maintenance cycle is used once, except for maintenance cycles having $m = 1$. All samples prior to p_i are included. All samples after the point p_i are removed. The algorithm can be seen in 1, and an illustration of the intention of the function is shown in Figure 2.1.

Algorithm 1 Implementation of truncating maintenance cycles

Input: X grouped by i, T_s
Output: X grouped by i, cut off at a random point p_i
for Cycle in $\{i_1, i_2, ..., i_m\}$ **do**
$p_i = \text{random.number}(\max(T_i - T_s, 1), T_i - 2)$
$\text{Cycle} = \text{Cycle}[t \leq p_i]$
$y_i = T_i - p_i$
end for

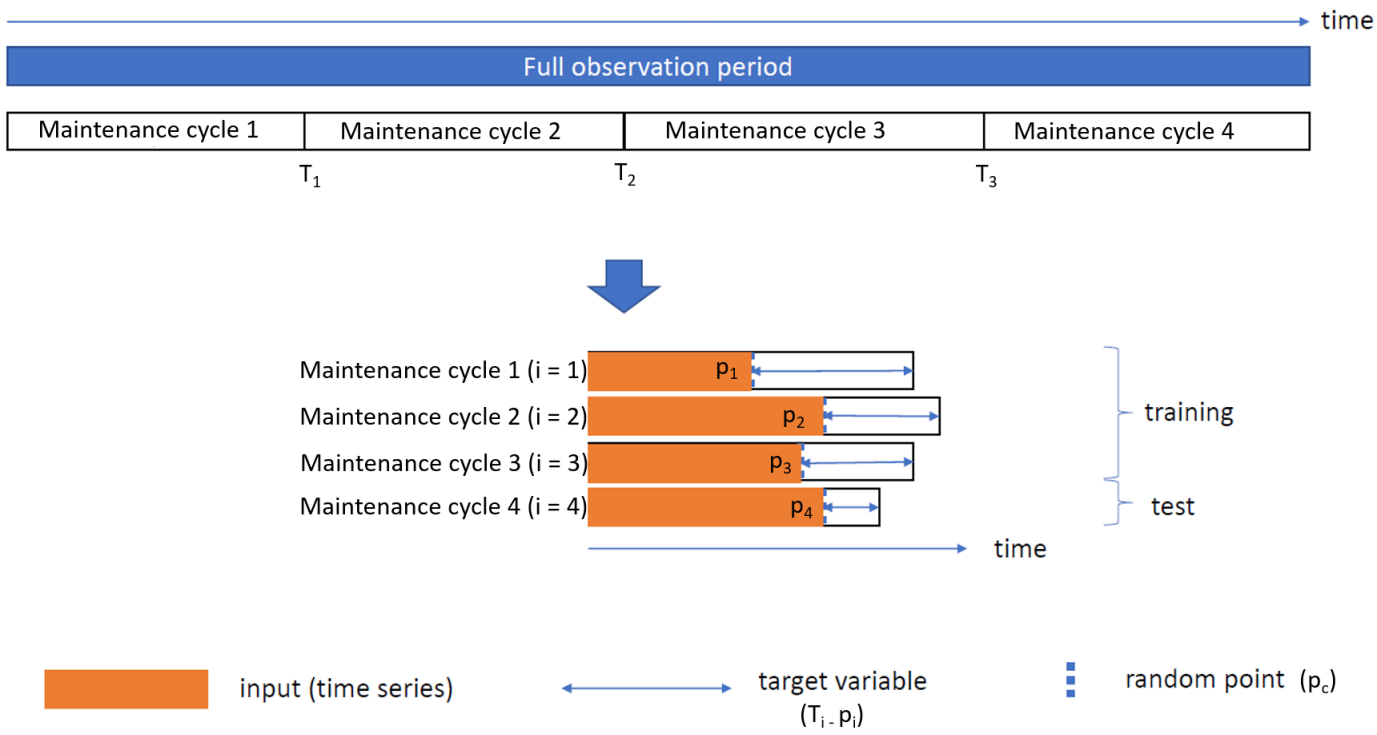

Figure 2.1: The problem setup for predictive maintenance

After truncating the maintenance cycles, data was standardised. This shifts the data so that

the mean of each column is 0, and perform scaling leading to the standard deviation per column being 1. This is done as follows per feature in the dataset:

$$x'_t = \frac{x_t^i - \mu_i}{\sigma_i} \tag{2.1}$$

where x_t is a given measurement variable in the dataset, μ_i mean value of x_i, and σ_i is the standard deviation of x_i.

2.2 Hidden Markov Model with Principal Component Analysis

The first approach examined in this paper is the Hidden Markov Model with the result of a principle component analysis (PCA) as input.

2.2.1 Principal Component Analysis

PCA (Principal Component Analysis) is an unsupervised linear transformation technique used for feature extraction and dimensionality reduction, which creates new features based on an input dataset. The method aims to describe a high share of the variance in the dataset with a low number of features [53]. This is accomplished by identifying correlations between input variables, which is then used to create new features. In dimensionality reduction, these features are obtained by projecting the input to a new subspace, which spans along the directions with highest variance. The new subspace has dimension $Q \times \tilde{n}$; as many as or fewer features than the original dataset $\mathbf{X}$ having dimension $Q \times n$, where $Q = \sum_{i=1}^{m} p_i$. The new features are ordered by their contributions to the total variance in the dataset. The principal components will then be the orthogonal axis to the new subspace.
Summarised, PCA can be conducted in the following steps [53]:

- Perform standardisation on the input dataset, as shown in Chapter 2.1
- Construct the symmetric covariance matrix $\mathbf{\Sigma}$ with dimensions $Q \times n$ by computing the covariance between input variables, pair-wise: $\mathbf{\Sigma} \propto \mathbf{X'X}$, where $\mathbf{X}$ is the $Q \times n$ input matrix with columns $X_{1,\dots,n}$.
- Decompose the covariance matrix into its eigenvectors- and values by satisfying the equation: $\mathbf{\Sigma}\boldsymbol{v} = \lambda\boldsymbol{v}$ where λ is the eigenvalue and $\boldsymbol{v}$ is the eigenvector. Sort eigenvalues in decreasing order.
- Select the first $\tilde{n}$ eigenvectors (user input for the number of features).
- Create a projection matrix $\mathbf{P}$ from the selected eigenvectors.
- Transform the input data set with dimension $Q \times n$ using $\mathbf{P}$, so that the matrix $\mathbf{X}$ reflected on the new feature space is calculated by $\mathbf{XP} = \mathbf{Z}$.

The number of features is defined by the user, being prompted with a trade-off between a low number of features in the new dataset and the amount of explained variance in these new features [53]. The explained variance for a given eigenvalue λ_u can be calculated by: $\text{expvar}_\text{u} = \frac{\lambda_u}{\sum_{u=1}^{Q} \lambda_u}$. A useful tool for this decision is an explained variance-plot, showing the cumulative amount of explained variance $\sum_{u=1}^{n} \text{expvar}_\text{u}$ for each number of features u.

2.2.2 Hidden Markov Model

The Hidden Markov Model (HMM) is a statistical model based on a Markov chain [55]. A Markov chain is defined as a sequence of states, where the probability of a given state in the sequence is only dependent on the state immediately prior to it [56]. This is known as the first-order Markov property, which the states must satisfy in addition to being finite [57] [58]. In an HMM, we assume these states $1, \ldots, N$ are hidden, and not observable. What we *can* observe is the observations x_t produced by the hidden state z_t at a given time. An arbitrary family of probability distributions is applicable to model emission probabilities, however, a common choice is to use the Gaussian distribution, which can be parameterised using a mean vector and a covariance matrix for each given state.

HMM's are a class of probabilistic models used to monitor the time-wise development of a stochastic process [59]. Here, a stochastic process acting like a Markov chain, produces a hidden state z_t at a given time which can only be inferred by a number of new stochastic processes. All observations x_t have a probability distribution for their possible hidden states [60]. In HMMs, transition matrices (A) are used to describe the evolution of a hidden state z_t over time [58]. The transition matrix contains the probability distribution for which state the system will be in at the next time step, based on the current state [61]. A simplified visualisation of the HMM can be seen in Figure 2.2.

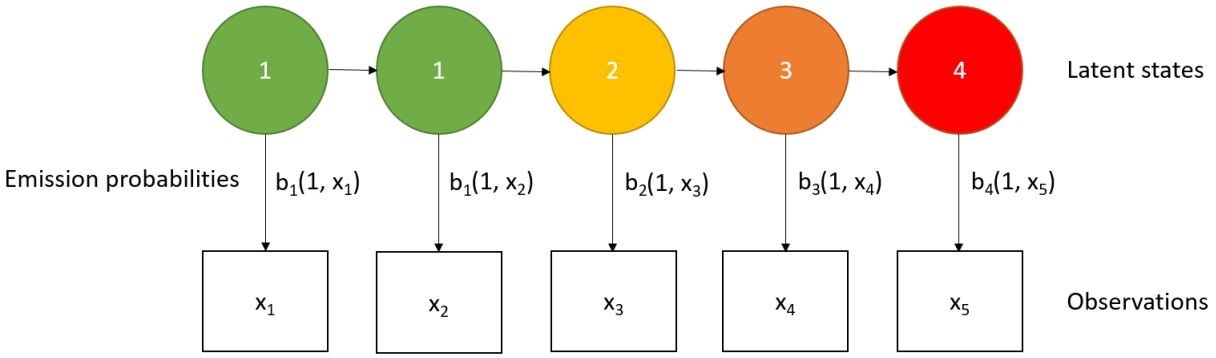

Figure 2.2: An example showing how the connection between the hidden states and the observations in a Hidden Markov Model

In the following, we denote the latent state at time t by $z_t \in \{1, \ldots, N\}$, while the observation at time t is denoted by x_t. The state sequence z_t is assumed to fulfil the Markov property, given by the condition independence

$$P(z_t|z_{t-1}, z_{t-2}, \ldots, z_0) = P(z_t|z_{t-1}),$$

for all time points t. Further, the HMM assumes that the observation x_t depends exclusively on z_t, without any other time-related factors.
In a simplified example with discrete observations $x_t \in \{1, \ldots, k\}$, an HMM is determined by 3 parameters:

- A transition probability matrix $A \in \mathbb{R}^{N \times N}$, denoting the probabilities $P(z_t = h|z_{t-1} = j)$ that the latent state moves from a state $h \in \{1, \ldots, N\}$ (rows) to a state $j \in \{1, \ldots, N\}$ (columns) in one time step,
- An emission probability matrix $B \in \mathbb{R}^{N \times k}$, denoting the probabilities $P(x_t = h|z_t = j)$ to observe value $j \in \{1, \ldots, k\}$ (columns) given the latent state $h \in \{1, \ldots, N\}$ (rows) in the same time step

- An initial state probability vector $\pi \in \mathbb{R}^N$, denoting the discrete probability distribution $P(x_0 = h)$ over all latent states $h \in \{1, \ldots, N\}$ at time $t = 0$.

In this example, the given HMM is determined by the parameter set $\rho = (A, B, \pi)$.
In real-world settings, observations may be continuous, leading to a continuous emission distribution, e.g. a Gaussian distribution $P(x_t|z_t) \sim N(\mu_{z_t}, \sigma^2_{z_t})$. This type of HMM is called Gaussian HMM (GHMM). In this case, the emission probabilities can no longer be represented in a probability matrix B, but are implicitly contained in the model parameters μ_{z_t} and σ_{z_t}, which represent the current latent state $z_t \in \{1, \ldots, N\}$. Thus, the number of model parameters increases to one pair $(\mu_{z_t}, \sigma_{z_t})$ for each of the N latent states, summarised as $\rho = (A, \pi, \mu_1, \ldots, \mu_n, \sigma_1, \ldots, \sigma_n)$.
HMMs may incorporate more than one observation variable, which generalises $x_t \in \mathbb{R}$ to a vector of observations $\boldsymbol{x}_t \in \mathbb{R}^n$. In the framework of GHMMs, the parameter set generalises to mean vectors $\boldsymbol{\mu}_h \in \mathbb{R}^n$ and covariance matrices $\boldsymbol{\Sigma}_h \in \mathbb{R}^{n \times n}$ for each latent state $h \in \{1, \ldots, n\}$. This results in the full parameter set $\rho = (A, \pi, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_n)$.
A possible generalisation of the GHMM is the Gaussian Mixture Model HMM (GMMHMM), which allows each state to define its associated probability via a mixture of Gaussians—however, this leads to a strong increase in the number of model parameters and thus, has a higher risk of overfitting, if the data volume is limited [61].

The Hidden Markov Model can be trained in an unsupervised way that will find the hidden state z_t corresponding to each observation x_t in the dataset, based on its values. It is trained with time series observations X as input, as well as their length.
There are three main tasks related to HMMs [59]:

- Evaluation - finding the likelihood of the observed sequence X given the model ρ and the sequence. This is known as the forward algorithm.
- Applying the Viterbi algorithm to find the most likely state sequence z_t given the model ρ and the sequence.
- Using the Baum-Welch algorithm to find the model that maximises the probability of the observation x_t.

The number of states N in an HMM is a user defined input. Another user defined input is the type of covariance matrix to be created, which determines the shape and number of parameters associated with the estimation of $\boldsymbol{\Sigma}_{\mathbf{z_t}}$. Common options here are:

- Full - The covariance matrix has an unrestricted covariance matrix with covariance between each input feature per state
- Diagonal - The covariance matrix contains a diagonal covariance matrix per state
- Spherical - The covariance matrix contains one variance value per state, applied for all input features
- Tied - One full covariance matrix used for all states

For this use, one strength of the Hidden Markov Model is that the user can determine initial matrices for probabilities, and select which are to be trained by the model [62].
In terms of predictive maintenance, we assume that the hidden state z_t of each time point in the data will be the equipment state of health. For the application in this paper, the evaluation will be the most relevant problem type - finding the most likely sequence of health

states for the equipment. The observations in use are the sensor readings from the SCADA system. Therefore, the HMM will return which hidden state, or condition, being most likely to have produced the observation x_t [57]. The transition matrix will describe the probability distribution for the development of equipment moving to a new risk of failure.. This matrix can in turn be used to determine at what time the system is most likely to be in the final, failed state.

The HMM is a data driven, computationally efficient modelling method, which is has optimal performance with few features [55]. Therefore, the PCA is a good option to decrease the number of features to optimise the input for the Hidden Markov Model.
Hidden Markov models are widely used in life-science. For instance, the authors of [63] use Hidden Markov models for system segmenting uncharacteristic genome DNA sequences into exons, introns and intergenic regions. This paper uses an HMM implementation called VEIL (Viterbi Exon-Intron Locator).
Another use of HMM's is emotion recognition through speech, presented by the authors of [64]. In this work, continuous HMM's are compared with Gaussian Mixture Model HMM's. The result were a higher degree of emotion recognition than for humans, using 7 hidden states.
Analysis of human and animal behaviour is also a common use for HMM's. The authors of [65] presents an HMM that recognises simple human actions, like simple manual operations. The results were a framework to be used for learning human actions from observing them.

2.2.3 Post-hoc analysis

In order to compare the location of the predicted states relative to the actual RUL, a Wilcox signed-rank test is a possible approach, as well as the Student's t-test [66]. The strength of the Wilcox Signed Rank test compared to Student's t-test is that it uses the signs and relative magnitudes of data instead of the actual data [67]. The Wilcox signed-rank test is applicable for non-gaussian distributions, and can be summarised in three steps for comparing two lists of data, as done by the authors of [67]:

- Omit zero-valued differences between the data lists, sort the residual differences in an ascending order according to their value. Assign them rank number 1,2,3 etc.
- Summarise the positive and negative differences separated.
- The null hypothesis is that the positive and negative rank numbers have no difference. A p-value is then found by comparing the sums with regards to a hypothesis assuming that the differences observed are random variations. The p-value combined with a significance threshold will indicate whether the null hypothesis can be rejected or not.

The null hypothesis of the Wilcox signed-rank test is that two groups of data are described with the same mean values, in this case, RUL.

Predicting time until high-risk state with Hidden Markov Model

An intuitive way to use the Hidden Markov Model for predictive maintenance is to define a state threshold, triggering a maintenance action when exceeded. For instance, if under the assumption of $N = 10$ states, a maintenance action can be triggered once the predicted state state exceeds 7.

In addition to this, the output from the trained model can be used in numerous ways to retrieve $\hat{y}$, the predicted RUL.

Estimating time steps to high-risk state using Monte Carlo simulation The Monte Carlo is a statistical simulation used for predicting possible outcomes based on statistics. Random numbers are used together with probabilities to simulate the most likely outcome of an event or sequence [68].
For Hidden Markov Models, we can use the probability distribution from the Hidden Markov Model to simulate the most likely outcomes from each hidden state [69]. For this thesis, we use the Monte Carlo simulation to simulate the number of steps from the system being in state z_t to being in the final, high-risk state z_N.
This simulation will be done by initiating the simulation at state z_t, then use random choices to simulate the transition between states, based on the transition probability matrix A. This operation will be done a given amount of times or until the state reaches z_N. The predicted amount of steps from state z_t to the final state z_N will be the mean value of the steps made in the simulations. The implementation of the Monte Carlo simulation is shown in 2. A similar is used by the authors of [55].

Algorithm 2 Implementation of the Monte Carlo algorithm for predicting time until high-risk states

Input: $\hat{z} \in \{1, 2, \ldots, N\}$
Output: $\hat{T} \in \{\hat{T}_1, \hat{T}_2, \ldots, \hat{T}_(N)\}$
for State in $\{1, 2, \ldots, N\}$ **do**
  steps $= 0$
  $z =$ State
  while $z \neq N$ **do**
    $z =$ random.choice($z \in \{1, 2, \ldots, N\}$, probabilities $= \{a_z\}$)
    steps$+ = 1$
  end while
  $\hat{T}_{State} =$ steps
end for

Calculating the expected hitting time of high-risk state A more theoretic approach to calculating the time until the system reaches high-risk states is to calculate the expected hitting time. In this method, the the values from the transition probability matrix A is used to calculate the time it will take for the system to move from the current state z_t to state z_N. The first hitting time T_i is defined as $T_i = \min\{t \geq 0 : z_t = N\}$ where i is a subset of the state space z [70]. In the case of this study, it will be the final state; the state of failure.
The mean hitting time for z_N is defined as $\bar{T}_{iN} = \mathbb{E}(T_N | z_0 = z)$ and is calculated by:

$$\bar{T}_{iN} = \begin{cases} 0 & \text{for } z_t = N \\ 1 + \sum_{s \neq N} o_{ts} T_{sN} & \text{for } z_t \neq N \end{cases} \tag{2.2}$$

Where o_{ts} is the probability of transition between the current state z_t and any given state z_s. This method takes all possible state-transitions from z_t to z_N, and calculates the mean hitting time in steps [71]. The mean hitting time approach is used by the authors of [61].

Estimating the time until high-risk state based on historical values The most practical and data driven method for estimating the time until high-risk state is to use the historical RUL values that from the data that the algorithm is trained on, for each state. In this method, the RUL for all predictions of a state z_t are taken into account and averaged. In other words, based on the historical data, what has the time until high-risk state been when the system has been in the current state z_t? When this value is calculated, the predicted state is decoded to an expected residual time until high-risk state in a one-to-one relationship.

2.3 Time series feature extraction and Random Forest Regressor

The second examined approach in this thesis is to extract new features from time series data, using the TSFRESH (Time Series Feature Extraction based on Scalable Hypotesis tests) package. The new features are used as input for an RFR (Random Forest Regressor) machine learning algorithm.

2.3.1 Time series feature extraction

TSFRESH is a feature extraction method which is well suited for IoT and Industry 4.0 applications, such as predictive maintenance [72]. The reason for this is that it takes metadata into account and opens to use different types of information per sample. Theses types includes [73]:

- Temporally Invariant Information (i.e. manufacturer of component)
- Temporally variant information with values changing in a volatile manner (i.e. operation status)
- Temporally variant information where values are continuously changing (i.e. temperature)

The values from the selected time window will then be aggregated to one line per object (ID), with new features describing the time series [72]. These features include, among others, mean values, max/min values, median, number of peaks, etc. For instance, the formula for the mean value for a measurement variable x_i in a maintenance cycle m will be $\bar{x}_i = \frac{\sum_{t=1}^{p_i} x_t^{(i)}}{p_i}$.
The Python package also contains a feature selection algorithm, taking the new features from TSFRESH and the target values as input. Features are selected using an automatically configured feature significance hypothesis test, testing whether the feature x_f is relevant for predicting the target value t or not [73]. Where any feature any feature $f \in \{1, \ldots, n\}$ is relevant for predicting the value t only if x_f and t are statistically independent.
From this hypothesis test, a p-value; the probability value for the feature being relevant, is computed. The hypothesis test is:
$H_0^f = \{f \text{ is relevant for predicting } y\}, H_1^f = \{f \text{ is not relevant for predicting } y\}$
Lastly, these p-values are reviewed by comparing them to the false-discovery rate to using the Benjamini-Yekutieli procedure [74]. Here, p-values less or equal to a critical value based on the false discovery rate, are rejected [75]. The feature selector will in this way find the most useful attributes to predict the target value [73]. By performing this feature selection algorithm the advantage is that the removal of redundant and irrelevant features lowers the probability of overfitting in the machine learning model [74].

The TSFRESH approach allows for large scale compression of the number of data points, by increasing the number of features. As seen in Figure 2.3, one time cycle for one ID (object) generates one row containing data with 789 new features per column in the original dataset, generated by a total of 77 different functions. A table showing all the functions used in this package can be seen in Appendix B. The table shows all functions used to generate new features, before any feature selection is conducted. Some of the functions involved will create several new column, with quite similar values. The multiple columns can be seen as redundant, and should be removed if they do not hold important information. Another aspect to consider about these functions is that they only extract information about one input feature at a time, meaning e.g. ratios or correlation between input parameters are not included. In real-world industrial applications, reducing the number of samples as we do with TSFRESH is a big advantage given their tendency of having large numbers of samples [76].

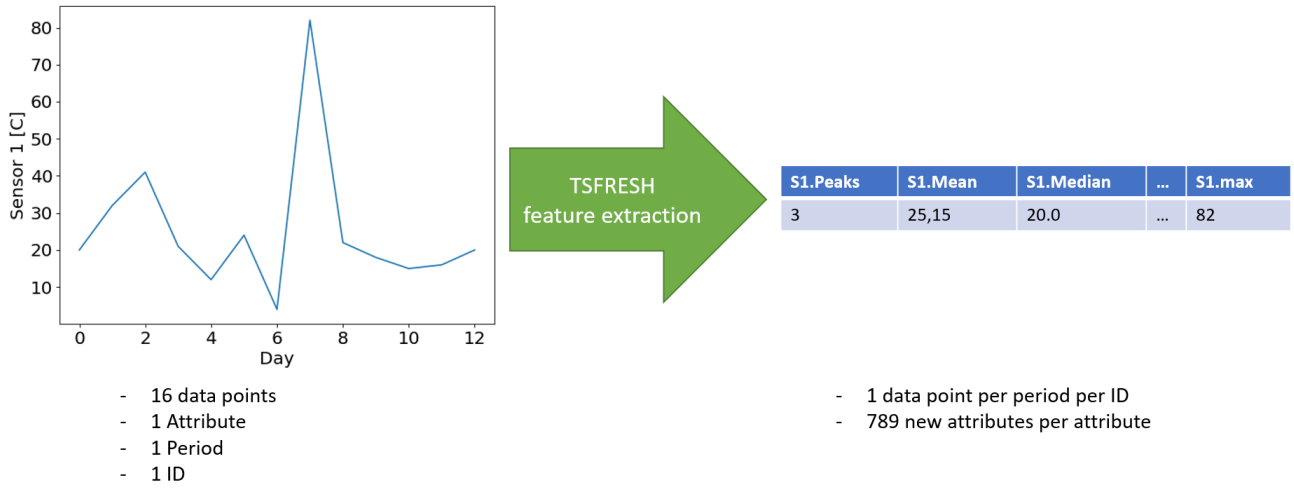

Figure 2.3: How the TSFRESH extracts a multivariate feature matrix from time series input.

In the work done by authors of [73], it was found that the TSFRESH method is more efficient when PCA is conducted on the output from TSFRESH, which is caused by the presence of many correlated attributes in datasets from real-world industrial applications. These could be eliminated by a feature selection algorithm, but exploited by a PCA to give more described variance. Another reason for the advantage of PCA is that machine learning algorithms perform worse on sparsely populated data. Decreasing the number of features and increasing their variance could be a great advantage in machine learning when the number of features is large [53]. However, the PCA will give less insights about what is used for making decisions.

2.3.2 Random Forest Regressor

The RFR (Random forest regressor) is a supervised machine learning model, modelling a continuous output. The principle is to train an ensemble of deep Decision Tree machine learning models whose predictions are combined to obtain a "total" prediction. Individual Decision Trees are trained on different subsets of the training data with n_s features called bootstrap samples, and individually fitted to these subsets [53]. For a dataset with n features, the number of features in each subset is by default: $n_s = \sqrt{n}$. This can, however, be tuned to a higher value for increased stability or decreased value for an increased degree of randomness [53].

Training each tree is done by grouping data and splitting it repetitively. The splitting is performed by selecting a random number of features without replacement. Then, one split is selected by optimising a threshold with respect to the Mean Squared Error between predicted value and ground truth. Splitting is repeated d times, where d is a user input parameter defined as tree depth. Because of these splits in the data, Decision Tree and RFR algorithms are able to handle non-linear relations between input and target.

The criteria and "pipeline" for splitting has the same architecture as a tree. The prediction of each tree is based on which final group, or leaf, the sample ends up in. Figure 2.4 demonstrates the described behaviour. In the RFR, the standard value for maximum depth d is blank, meaning that the splitting will be repeated until every leaf is pure or contain less than a given amount of samples. As a consequence, they often suffer from high variance individually, meaning they are overfitted to the subset of the data they are trained on. To control the degree of overfitting, an adequate selection of d is crucial [53]. In the RFR, results from all decision trees in the ensemble are combined by using the mean value. This makes the model less prone to overfitting, as the different Decision Trees are trained on different subsets of the data, hence will weight the attributes and values differently.

The number of decision trees u and their maximum depth d are the most important hyperparameters to tune for a random forest machine learning model. A high value for u may lead to a better performance, but also affects the computational costs. The maximum depth d can be used to limit the risk of overfitting per decision tree. The user further specifies whether the RFR ensemble shall be optimised with respect to one of the following (most common) metrics: Mean squared error, Poisson deviance or absolute error.

A useful property of a trained RFR is that it is able to quantify the contribution of each feature to the data split in each decision tree. The degree of use is aggregated for all Decision Trees and normalised, and can be displayed using a function for the RFR called feature importance. This returns a table showing how much each feature "contributed" towards training the algorithm towards an optimal objective function.

The RFR algorithm differs from to the traditional machine learning algorithms. While the traditional algorithms often use weights for different features and mathematical predictions, the RFR use grouping of data. This way, it is able to build complex decision boundaries. One of the unique features for this algorithm is that one does not make assumptions of whether the data is linearly separably or not when using the algorithm. This makes it applicable to a wide range of problems. A particular strength is its performance on high-dimensional datasets [77]. Another advantage of the RFR is that it is less sensitive to outlier data, since it the several Decision Trees are trained on different subset of data, the impact of extreme

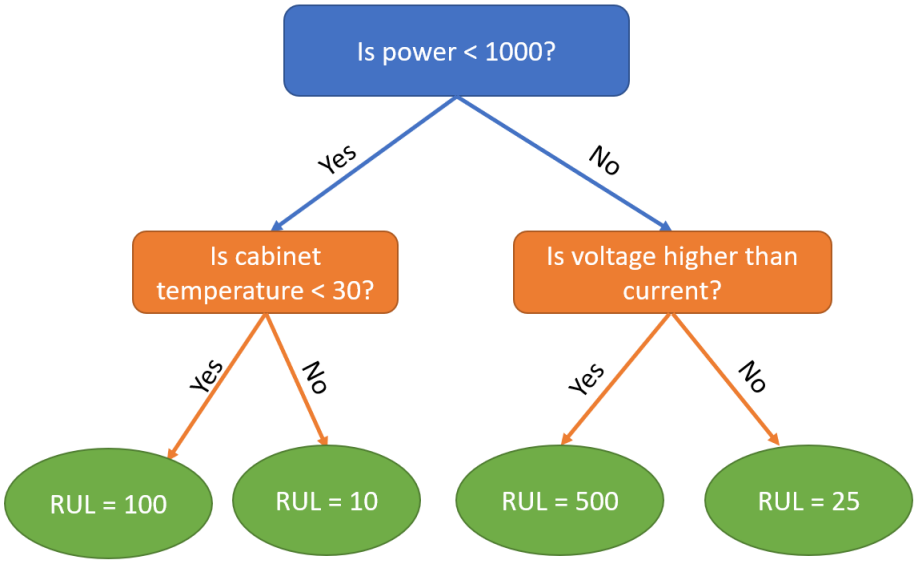

Figure 2.4: A decision tree from the experiment in this thesis with max depth = 3.

values will be lower. Other strengths of the RFR is that it requires little hyperparameter tuning, and does not require scaling of the input data [53]. One drawback for the RFR is that it tends to be a bit unstable due to its randomised behaviour, especially if the number of features exceeds the number of samples.

Chapter 3

Experiments

This chapter describes the experiments conducted in this thesis, as well as their results.

3.1 Experimental setup

The data consists of six sub-plants with 22 inverters each. Each inverter delivers 131 measurement variables, sampled every minute. These measurement variables include e.g. current, voltage and temperature in an inverter. Manual filtering described in Section 1.1.5 is applied to the data, so that only relevant data input is included in the model.

Sensor data is combined with weather data, which is recorded at the same resolution. There are six weather stations per sub-plant. Every inverter has a respective parent weather station, which is used to join the two data sources, based in their time stamp. Therefore, each line contains data from one inverter at one time point, as well as data from the closest weather station at the same time period. Weather and inverter sensor data are both treated equally as input variables for the model.

In order to decrease runtimes in the experiment, data has been re-sampled to 30 minute intervals, using mean value aggregation[1]. This is due to the large amount of data involved, as well as the variable experiences in previous work, presented in chapter 1.2.1. In order to contain the variability in the data while still not making the dataset unbearably large, the 30 minute aggregation was conducted. The fault detection may have been better with a higher resolution, although it could have lead to the predictive models getting confused around local minimums and maximums. Another measurement against high runtimes is the removal of obviously irrelevant input variables, such as manufacturer, being the same for all inverters.

The strength of the mean value aggregation could be that it is better for detecting that the measurement variables are shifting towards higher or lower values - when they are not outliers. This can assist in detecting degradation, and could be the reason for the better performance using the mean value aggregation. The resulting multivariate time series is a view containing 92 measurement variables, and one row per 30 minutes of operation from June 2019 up to the beginning of 2022, for each inverter. Out of these columns, 30 of them contain data indicating distinct error types, leaving 62 columns for analysis after the removal of these. The removal was done after using these to identify T_i for the decided failure type per inverter. In Table 3.1 one can see some example rows from the data from the inverters and the weather stations combined.

[1]Median value aggregation was also tested, but lead to slightly less accurate predictions.

Inverter	Timestamp	s1 DC power	s1 DC voltage	s1 DC current	s1 Phase 1 module temperature	...	Status	Irradiation horizontal	Irradiation incline
01	06.05.2019 08:00	855,19	1193,29	718,75	83,5	...	164	998,4	1037,0
01	06.05.2019 08:30	1031,68	1143,38	910,16	93,1	...	164	1025,4	1037,4
01	06.05.2019 09:00	1226,49	1095,22	1123,15	105,3	...	164	1035,9	1039,4

Table 3.1: A sample of rows and columns in the data used in the experiment, representing the structure and types of data involved

The failure type to focus on was decided together with the plant team, to ensure that a degradation related failure was selected. A separate column for RUL - counting the time until the next failure of the same type, was created by using T_i. The time period between each failure is identified by a unique maintenance cycle ID. Maintenance cycles are treated as independent from each other. For each equipment, the last maintenance cycle are removed, as the failure has yet to occur for these cycles. Figure 3.1 illustrates that the maintenance cycles are of different lengths, and the distribution is skewed towards the left side of the X-axis - indicating that the majority of maintenance cycles are shorter than 50 days. As preprocessing, maintenance cycles shorter than two days were discarded, resulting in a total of 817 samples to use for training and testing.

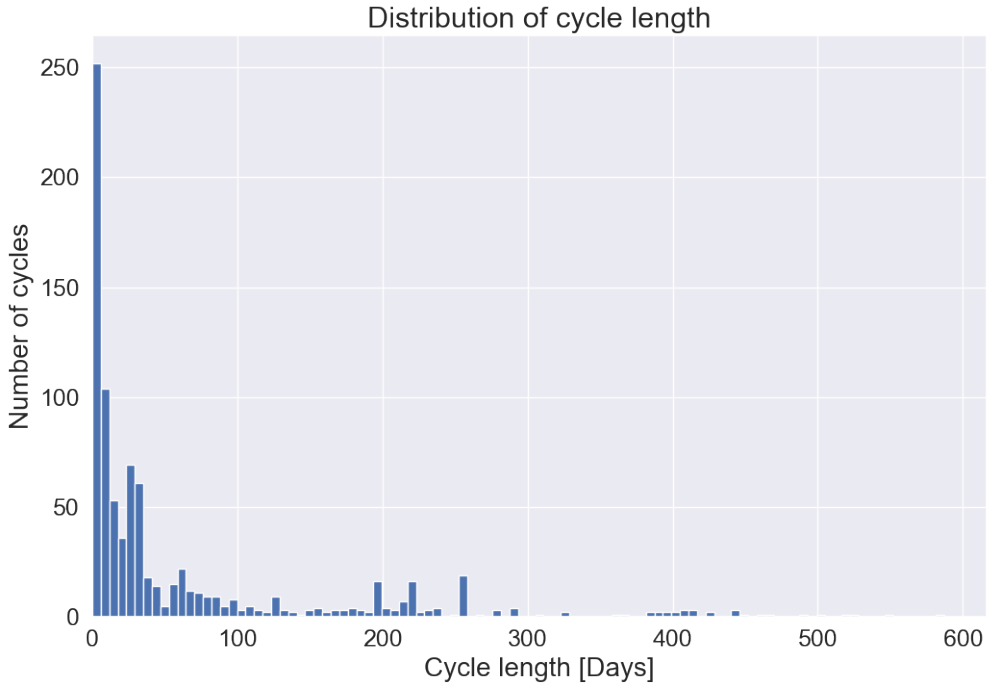

Figure 3.1: The distribution of the lengths of maintenance cycles in the data for analysis. The number of cycles having a length lower than 50 days is the clear majority.

The data was used in two pipelines, which a chart of the data flow in Figure 3.2 shows. As indicated in the flow chart, the first step consists of data preprocessing by outlier removal and standardisation. Irrelevant features are removed, along with data from when there was no production.

The data was split into a training and a test subset, comprising 70% (571 samples) and 30% (246 samples) of the full dataset, respectively. The selection of samples in the two subsets are done randomly.

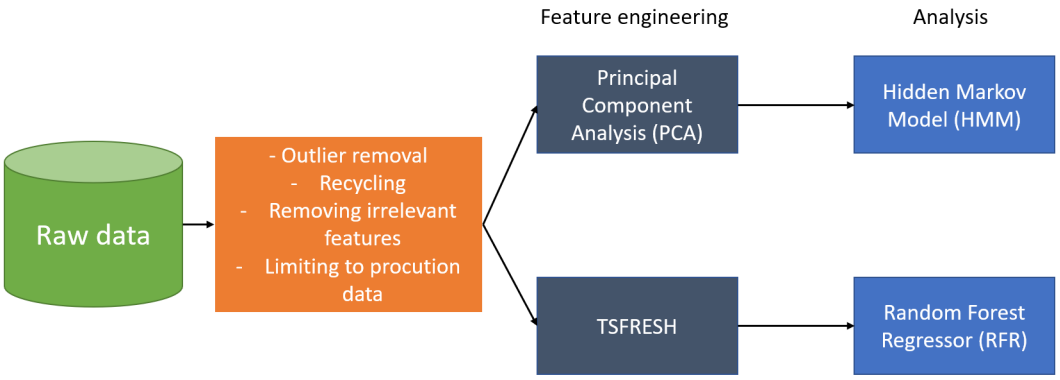

Figure 3.2: The flow of data in the experiment showing the flow from Raw data on the far left side to prediction model on the far right, and what is happening in the different steps of the flow.

3.2 Results

3.2.1 Preprocessing

The data was visualised and examined using a box-and-whiskers-plot, which can be seen in Figure 3.3. This plot shows in which variables there are outliers, for 10 example variables, anonymised. Variable 2 and 9 have a large number of outliers towards the upper quantile. Based on the plot, outlier measurements in respectively the lower and upper quantile were removed, for the variables and quantiles where outlier values were present. This means that individual rows in the maintenance cycles were removed. In order to remove as little data as possible, the data was visualised after every removal of outliers. This way, only the necessary data was removed.
In addition, nonphysical data; samples that have an unrealistically high or low value were removed as described in Figure 1.3. For instance, this included removing extremely high- or low temperature measures, pressure measures exceeding the laws of physics, etc.
After the outlier removal, standard scaling was conducted on the data.

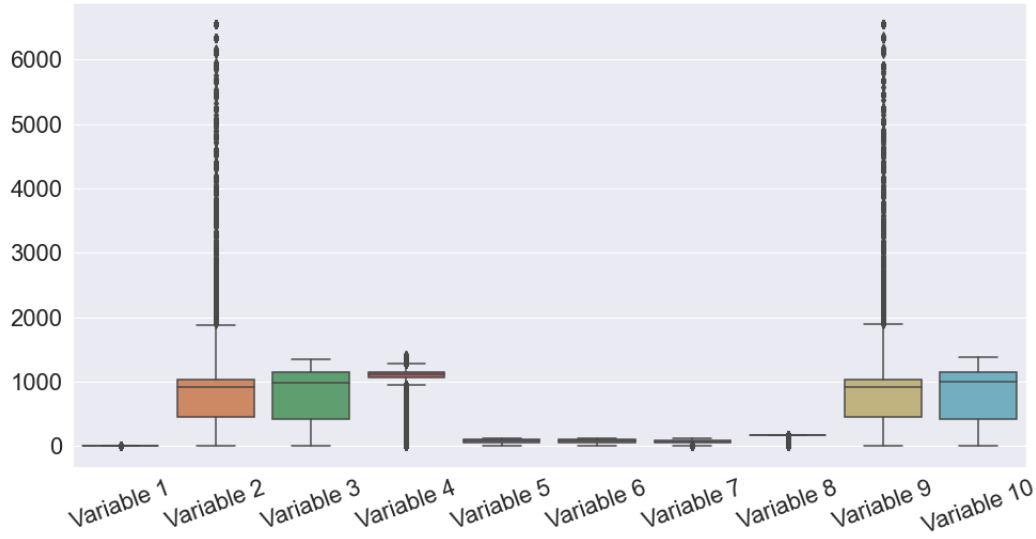

Figure 3.3: A box-and-whiskers-plot showing the distribution of data for 10 measurement variables

3.2.2 Hidden Markov Model with Principal Component Analysis

After the preprocessing, PCA (Principal Component Analysis) was conducted. The number of principal components was determined based on a graph showing the described variance as a function of the number of components extracted in the PCA. The explained variance graph is seen in Figure 3.4, indicating that the PCA has a high explained variance for a low number of features. Multiple numbers of components were tested here, between 3 components with 63,4% explained variance and 8 components with 79,4% explained variance. This was due to the explained variance graph had no obvious optimum. Another aspect considered in the election of components is the fact that HMM's known to perform better for a low number of features in the input data. After testing the algorithm performance based on the number of features in the input data, the choice fell on 3 components due to the results from any higher number of components returning a HMM with a very sparsely populated transition matrix. A sparsely populated transition matrix lead to the maintenance cycles going directly from state 1 to the final, failed, state at seemingly random points in time.

The HMM in use is a full covariance mapping. The number of states tested in the model is 5, 10 and 15. For this experiment, the states correspond to the risk of failure relying on the degree of wear at a given point in time. To achieve robust prediction results, the HMM was trained with 20 different random seeds. The predictions produced by the different random states were combined with mean value aggregation rounded to the closest state. The model was initialised by a start matrix with 100% probability of state 1, in order to reflect the state of freshly installed equipment - this can be seen in Table 3.2.

State	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Start probability	1.0	0	0	0	0	0	0	0	0	0

Table 3.2: The probability distribution of the initial state of a sequence in the Hidden Markov Model. A likelihood of 1.0 for start in in state 1 represents the condition of freshly installed equipment

The transition matrix was initialised as a diagonal matrix, giving zero probability of states transitioning to a lower state number - since the state represents risk of failure, relying on the equipment degradation, which can only increase. Probabilities in the upper triangular matrix

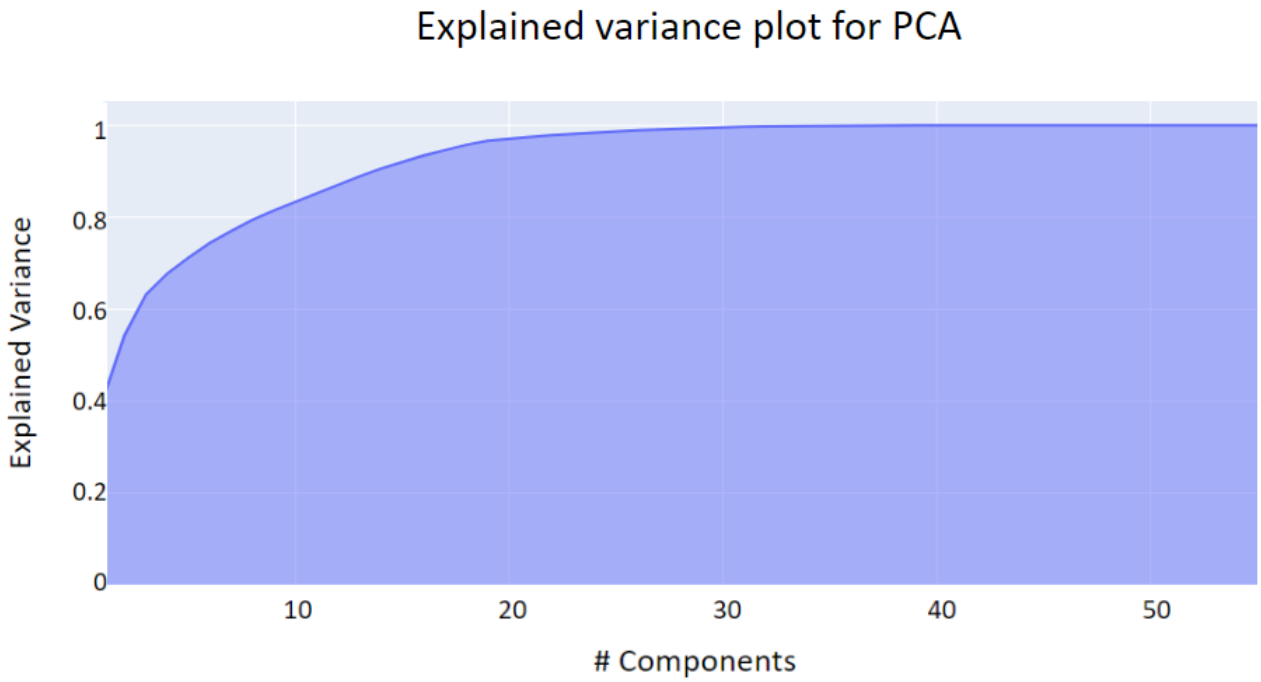

Figure 3.4: The explained variance graph from the PCA, showing the explained variance as a function of number of samples included. The graph shows a steep slope at the beginning and flattens out at around 15-20 states.

were initiated randomly, then trained using Baum-Welch algorithm. An initial transition matrix used for the experiment can be seen in Table 3.3.

From state/To state	1	2	3	4	5	6	7	8	9	10
1	0.5	0.175	0.125	0.075	0.05	0.025	0.125	0.0125	0.0125	0.0125
2	0	0.5	0.2	0.125	0.075	0.05	0.025	0.0125	0.0125	0
3	0	0	0.5	0.2	0.125	0.075	0.05	0.025	0.0125	0.0125
4	0	0	0	0.5	0.2	0.125	0.075	0.05	0.025	0.025
5	0	0	0	0	0.5	0.2	0.125	0.075	0.05	0.05
6	0	0	0	0	0	0.5	0.2	0.125	0.1	0.075
7	0	0	0	0	0	0	0.5	0.275	0.125	0.1
8	0	0	0	0	0	0	0	0.5	0.3	0.2
9	0	0	0	0	0	0	0	0	0.7	0.3
10	0	0	0	0	0	0	0	0	0	1

Table 3.3: The transition matrix from the HMM indicating the probabilities of transition between different states at a given time. The table shows that the lower left diagonal has 0-values, representing that degradation cannot decrease, but only increase over time

The trainable parameters for each hidden state i were: Transition probability vector A_i, the mean matrix $\mu_\mathbf{i}$ and the covariance matrix $\mathbf{\Sigma_i}$.

During training, the full maintenance cycle, ranging from $t = 1$ to $t = T_i$ was used instead of the restricted time period $\{1, \ldots, p_i\}$. To allow a fair comparison, testing was performed on truncated maintenance cycles, $t \in \{1, \ldots, p_i\}$.

Results The prediction by the HMM is a sequence of states showing the evolution of the latent state in each test sample over time. Figure 3.5 shows how the predicted states are transitioning towards higher values as the time of failure approaches, in a model with 10 hidden states. The reason for Maintenance cycles 2 and 4 are ended with about 40 and 20 days of RUL, is the truncation described in Section 2.1. The figure shows in some selected maintenance cycles how the Hidden State evolves over time, as the RUL decreases. An exception here is maintenance cycle 0, having a low RUL, and failing in state 5.

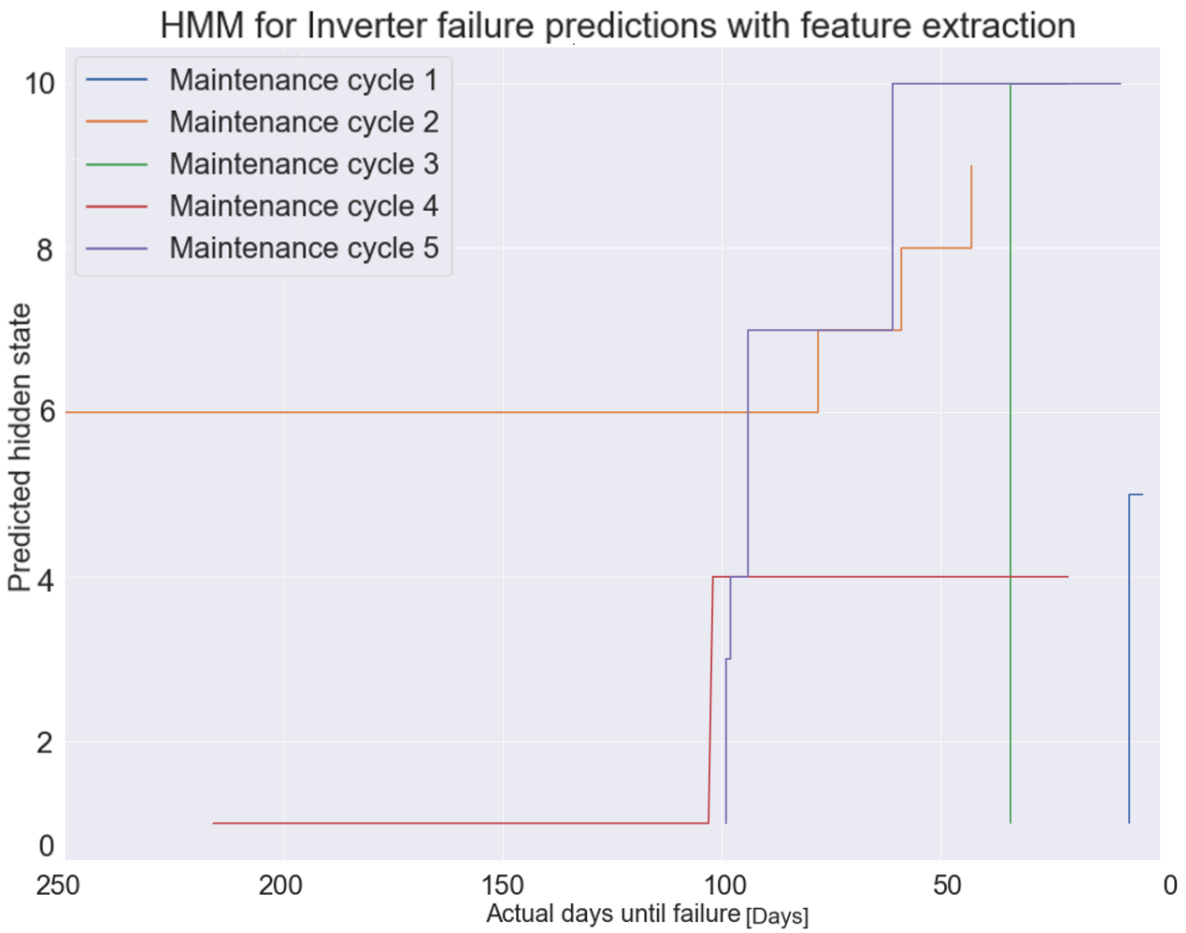

Figure 3.5: The evolution of the hidden state predicted by the HMM for five sample maintenance cycles as the RUL decreases for five maintenance cycles. The lines demonstrate how the hidden state changes relative to the RUL. The first occurrence of the line on the left hand side of the graph indicates the beginning of the maintenance cycle.

The states predicted for the test data represent risk classes related to the RUL of the device. As seen in Figures 3.6, 3.7 and 3.8, the RUL is generally lower in samples where the model has predicted a higher state. These boxplots show that RULs tend to decrease in the earlier states (indicating a failure at an early time in the maintenance cycle) and in the high states, whereas RULs are higher in mediocre states. The decline in RULs over the first states can be explained by the high number of short maintenance cycles in the study, as demonstrated in the overall cycle length distribution in Fig. 3.1 Note that single states, such as state 2 in Fig. 3.5, are sparsely populated and thus, contain mainly outliers, but do not contribute to the overall behaviour of the majority of samples.
For the 5 state HMM, Figure 3.6 shows that a correlation can be observed between the RUL and the predicted hidden state, following an initial life period represented by the first state. However, the figure also shows that the distribution of the actual RUL is quite high in the final states, as the higher quantile ends at about 130 days and there are several outliers from

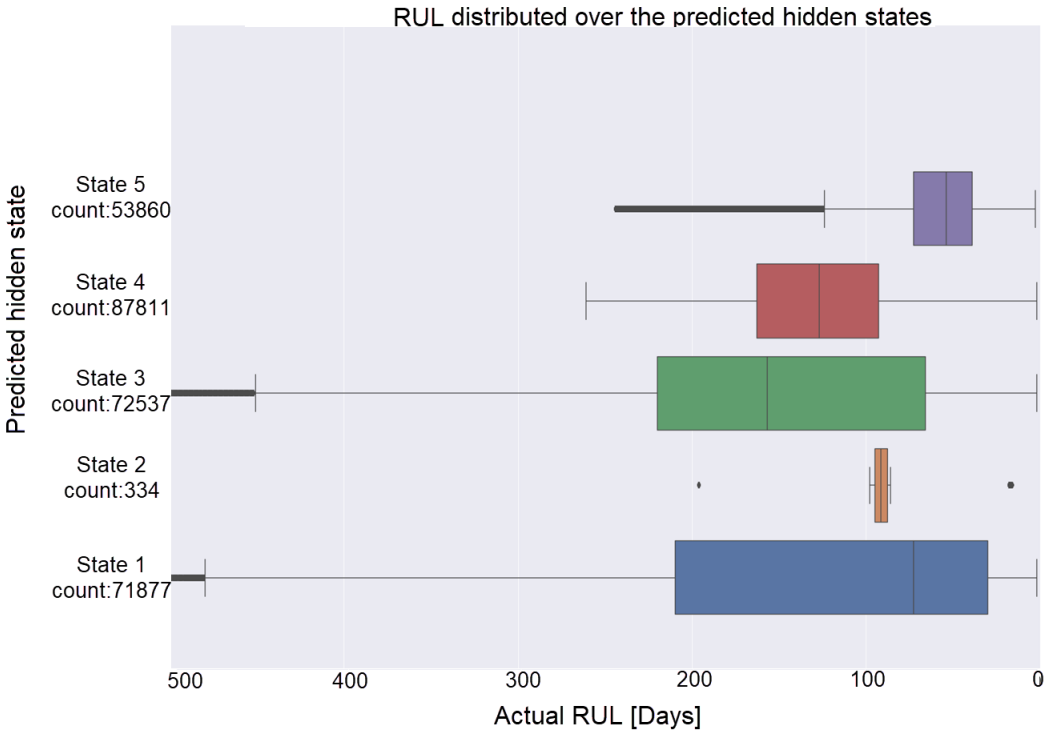

Figure 3.6: The distribution of actual RUL based on the predicted hidden state for the 5 state HMM, to show correlation between predicted hidden state and RUL

there towards RUL = 250 days.
However, the number of samples in the final two states is high.

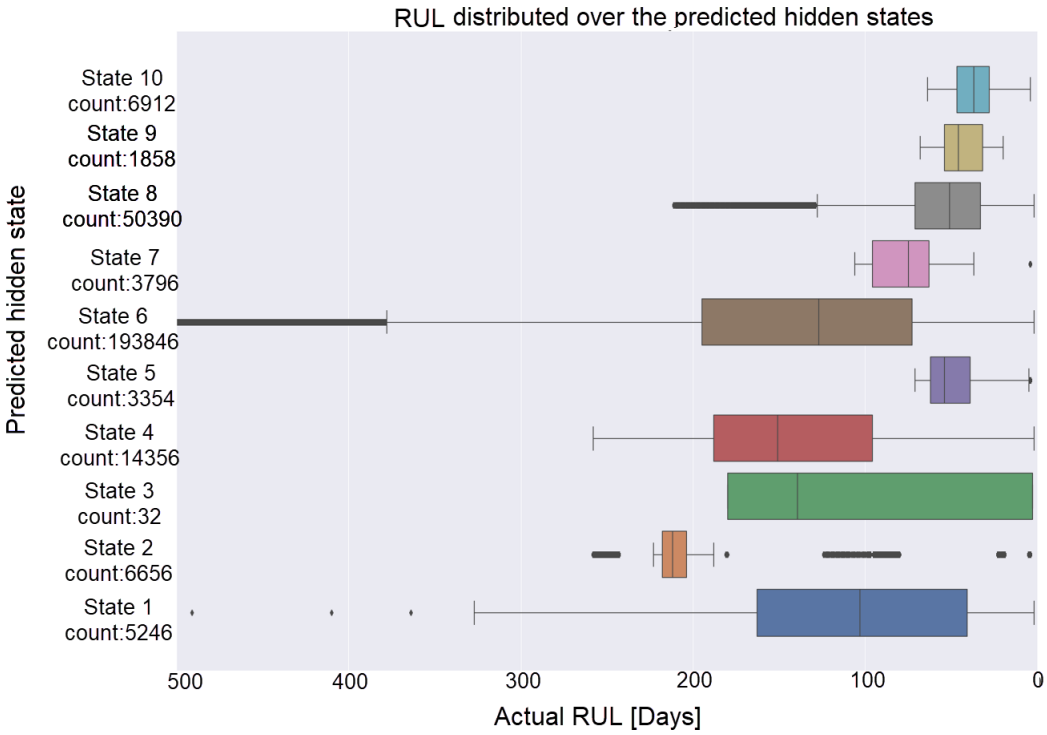

Figure 3.7: The distribution of actual RUL based on the predicted hidden state for the 10 state HMM, to show correlation between predicted hidden state and RUL

The test results for the 10 state HMM is shown in Figure 3.7. This plot also shows a strong correlation between the predicted state and the RUL following an initial time period (state 1) and neglecting sparsely populated states (e.g., state 3). In addition, large amounts of samples have been predicted for the final states 8-10. Some noteworthy results in this plot is the low lifetime distribution in state 5 and the low amount of samples in state 3.

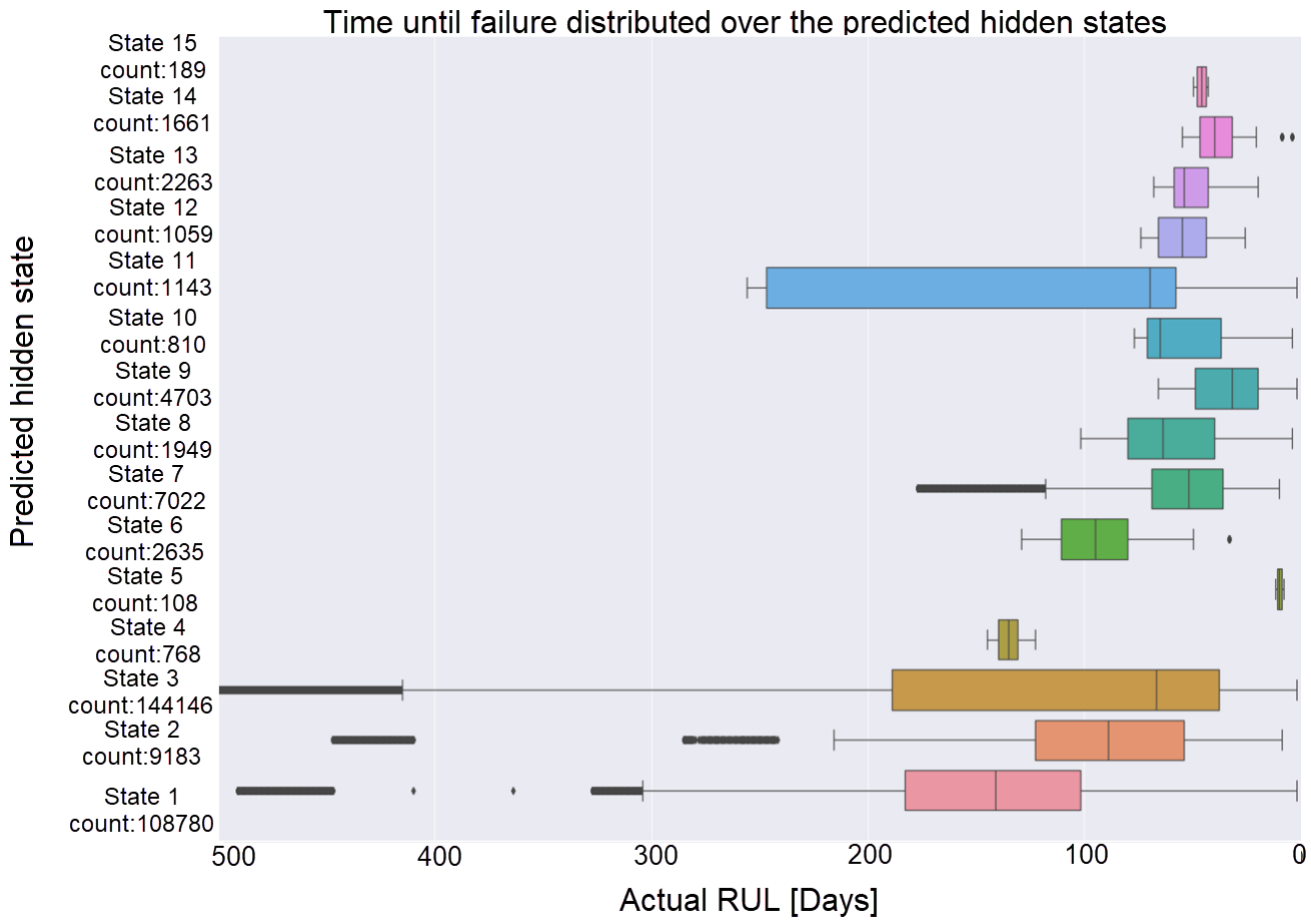

Figure 3.8: The distribution of actual RUL based on the predicted hidden state for the 15 state HMM, to show correlation between predicted hidden state and RUL

As for the 15 state HMM displayed in Figure 3.8, the model has a lower number of samples past state 3. This indicates that the transition probabilities towards the higher states are low, hence the maintenance cycles seem to get "stuck" in state 1 and 3.

Due to the large span in lifetimes in the final states of the 5-state model and the low distribution among states in the 15-state model, the 10 state model was the chosen model for further analysis. Seemingly, this model has a good trade-off between the distribution among states and correlation between the model complexity, steered by the number of states, and its explanatory power.

The correlation between the actual RUL and the predicted state seems to be present in several cases. In Figure 3.7 one can see some sample test results where the transitions towards higher states clearly has a high degree of correlation with degradation and RUL.

An interesting observation is that state 3 has only 34 samples. The reason for this is likely to be outlier data. Nevertheless, the low number of samples in this state makes it neglectably

small - as moving all the samples from this state to any of the other state would hardly make a difference in its distribution of RUL.

RUL estimation based on the predicted states from HMM was done, by decoding each state into a RUL value. The RUL value was obtained by calculating the mean RUL value for each state predicted on the training data. The prediction from this method on the test data achieved a RMSE of 133.5 and an R^2 of -0.45 compared to the actual RUL.

Post-hoc analysis The visualisation of the dynamic of state transitions over time is provided in Figure 3.9, which takes all maintenance cycles from the test data into consideration. The intention of the plot is to display the density of transitions between the predicted hidden states. The vertical blue lines indicate the hidden states, and the size of the blue lines indicate the number of samples in the hidden state they are representing. The black and grey lines represents the predicted hidden state for a maintenance cycle moving from one state to anther. In a nutshell, the graph describes how the predicted hidden state moves from one state to another as the time passes. The plot shows between which states the highest number of transitions occur within the test data. Note that transition are restricted to the increasing order of the states; however, a jump over more than one state in a single time step (e.g., from state 4 to state 6), is possible. happen for the test data. The figure shows high number of direct transitions between state 1 and 6, namely 166 out of 179 maintenance cycles leaving state one directly or indirectly end up in state 6. The majority of the transitions to state 6 come directly from state 1. From state 6, only 63 values moves onto the states above. However, the percentage-wise most absorptive state is state 8, where only 3 out of 55 maintenance cycles move on to the final two states, hosting respectively 7 and 12 maintenance cycles.

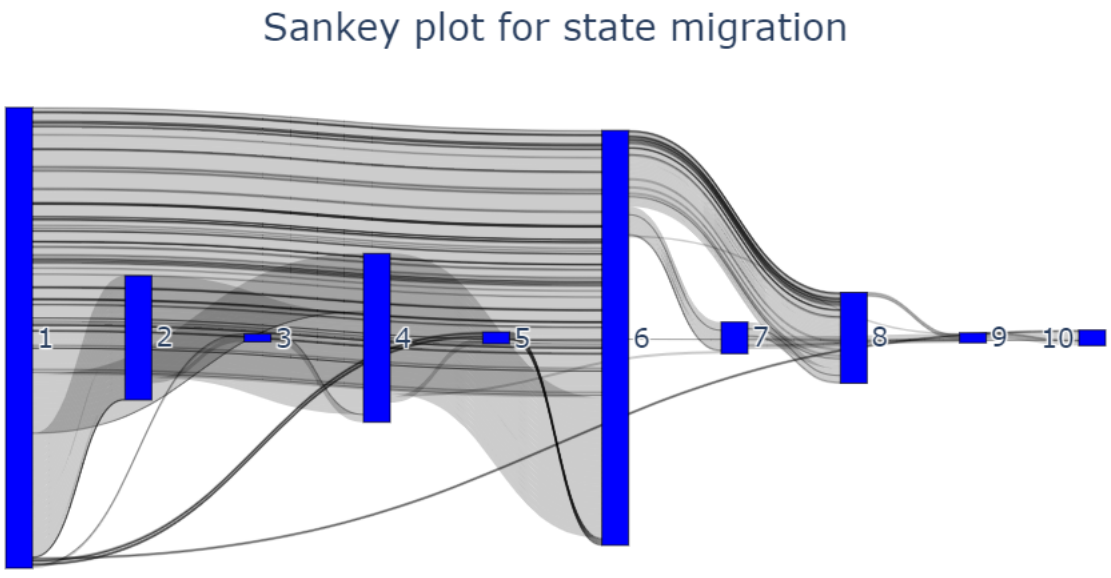

Figure 3.9: A Sankey plot displaying how the maintenance cycles transition between states in the test data. The figure shows a lot of cycles migrating to state 6, then fewer moving to and ending at state 8.

To underline the observations about the trained states numerically, predicted by the HMM, a statistical test was conducted on the RULs for each state predicted, using the pair-wise Wilcoxon test. This test compared the actual RULs depicted in Fig. 3.6-3.8, in different

states predicted by the HMM on the test data. Table 3.4 shows the p-values for the different states, testing whether their expected RULs are equal or not. Any below a significance level $\alpha = 0.01$ means that the null hypothesis should be rejected, i.e. a significant difference in the mean values exists.

State/State	1	2	3	4	5	6	7	8	9	10
1	1	0	0	0.114	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0.18	0	0	0	0
4	0.114	0	0	1	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0.3	0
6	0	0	0	0.18	0	1	0	0	0	0
7	0	0	0	0	0	0	1	0	0	0
8	0	0	0	0	0	0	0	1	0.755	0
9	0	0	0	0	0.3	0	0	0.755	1	0
10	0	0	0	0	0	0	0	0	0	1

Table 3.4: The p-values for the null hypothesis from the pairwise Wilcoxon test states compares distributions of RULs given the predicted hidden states. Thresholded at 0.01.

In summary, the pairwise test suggests that almost all pairs of states differ in terms of actual RULs. Only 4 pairs of states cannot be considered to be different according to the given data: states 1 and 4, states 4 and 6, states 5 and 9, as well as states 8 and 9. While 1, 4, and 6 describe the mainstream pathway at a level of high RULs, states 5, 8, and 9 contain samples, which are close to an error.

In order to get a pinpoint of what the RUL is in a given hidden state predicted by the HMM, implementations of all three approaches presented in Section 2.2.3 were tested. An example of this can be seen in Figure 3.10, where the prediction of RUL is done by running a Monte Carlo simulation based on the transition matrix A from the HMM. Despite the predicted RUL from the simulation not being particularly accurate for individual samples, there is clearly some general resemblance between the graphs in Figures 3.7 and 3.10. The distribution of predicted RUL moves to the right side of the x-axis of Figure 3.10 as the hidden state increases, indicating that the output could be useful to give an estimation of RUL. Out of the three methods discussed in chapter 2.2.3, the Monte Carlo simulation approach was the method performing best on the test data with regards to the correlation between predicted hidden state and actual RUL.

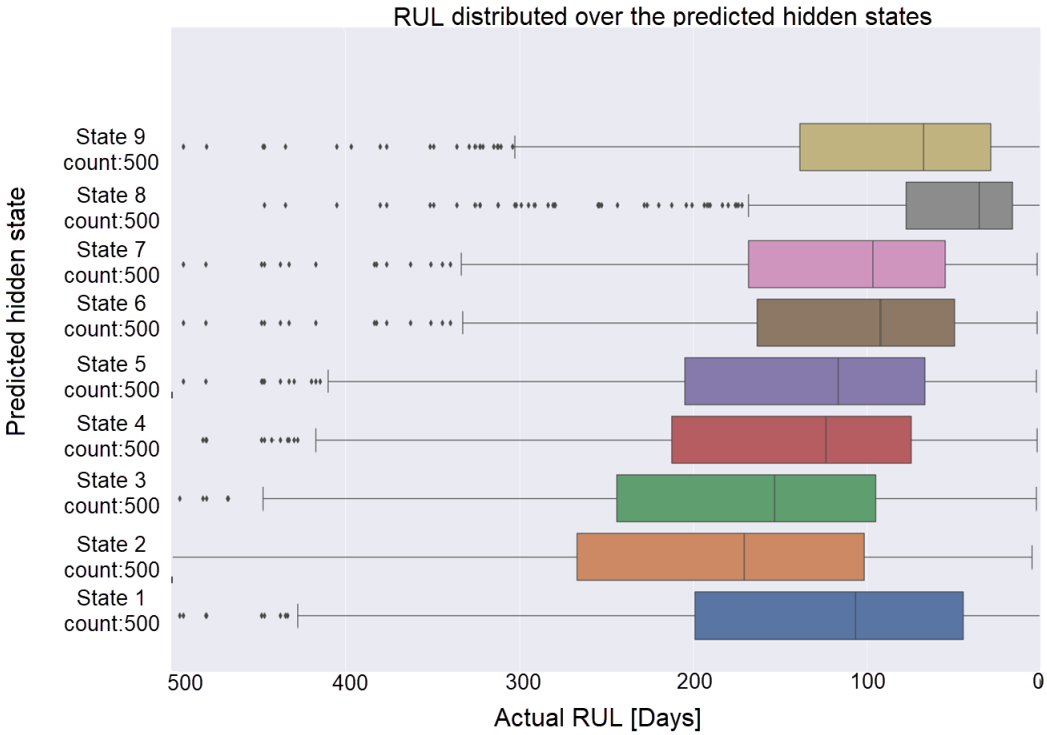

Figure 3.10: The prediction of time to high-risk state from Monte Carlo simulation based on the transition matrix from the Hidden Markov Model, showing the distribution of the simulated time to high-risk state for the different states.

3.2.3 Time series feature extraction with Random Forest Regression

With the preprocessed data as input, new features were extracted from the dataset using the TSFRESH package. No limitations in features was made here, so the extraction takes all measurement variables in the data from SCADA into account. Based on 55 variables in the original dataset, the number of extracted features was 41552 after excluding columns containing information about RUL and inverter ID. After the TSFRESH feature extraction some of the returned columns contained numerous NaN values. Columns containing exclusively NaN values were removed along with other columns containing more than 400 (50%) NaN values, leading to 285 columns being removed. The rest of the columns were imputed using a built-in function for TSFRESH, using the median value of the columns to fill the NaN values. As a next step, the built-in TSFRESH feature selection was performed, which calculates feature significance p-value, followed by a hypothesis test using the Benjamini-Yekutieli method. The feature selection reduced the number of columns in the dataset to 10970.

The RFR algorithm was trained on the training subset using several parameter setups, and tested for several parameter setups. The target function to optimise is the Mean Squared Error between the predicted and the ground truth RULs.

Results On the test data, he machine-learning based pipeline involving TSFRESH feature extraction and RFR machine learning algorithm achieved the RMSE and R2 scores displayed in Tables 3.5 and 3.6, respectively.
As a consequence of the results displayed in these tables, the final prediction was made with a model with $u = 100$ estimators and a max depth $d = 10$. In general, the Tables show that the procedure is rather robust under distinct parameter settings, depicting only minor

Estimators\Max depth	**5**	**10**	**15**	**20**	**25**
100	20.26	20.14	20.20	20.19	20.19
200	20.27	20.18	20.21	20.21	20.21
300	20.26	20.16	20.18	20.17	20.17
400	20.26	20.16	20.18	20.18	20.17

Table 3.5: The RMSE scores produced on the test data by the different parameters for the Random Forest Regressor

Estimators\Max depth	**5**	**10**	**15**	**20**	**25**
100	0.235	0.245	0.240	0.241	0.241
200	0.235	0.242	0.239	0.239	0.239
300	0.235	0.243	0.241	0.242	0.242
400	0.235	0.243	0.241	0.242	0.242

Table 3.6: The r2 scores produced on the test data by the different parameters for the Random Forest Regressor

changes in both, RMSE and R2.

Figure 3.11 shows the predicted RUL compared to the ground truth, both in mean values and per sample. The pattern shows that the accuracy in the prediction increases as the time approaches the failure event. Although there are minor deviations, the general trend of the plot is suggesting a positive (although not linear) correlation to the ground truth. Even though outliers exist in the predictions, the correlation between predicted RUL and the ground truth is obvious.

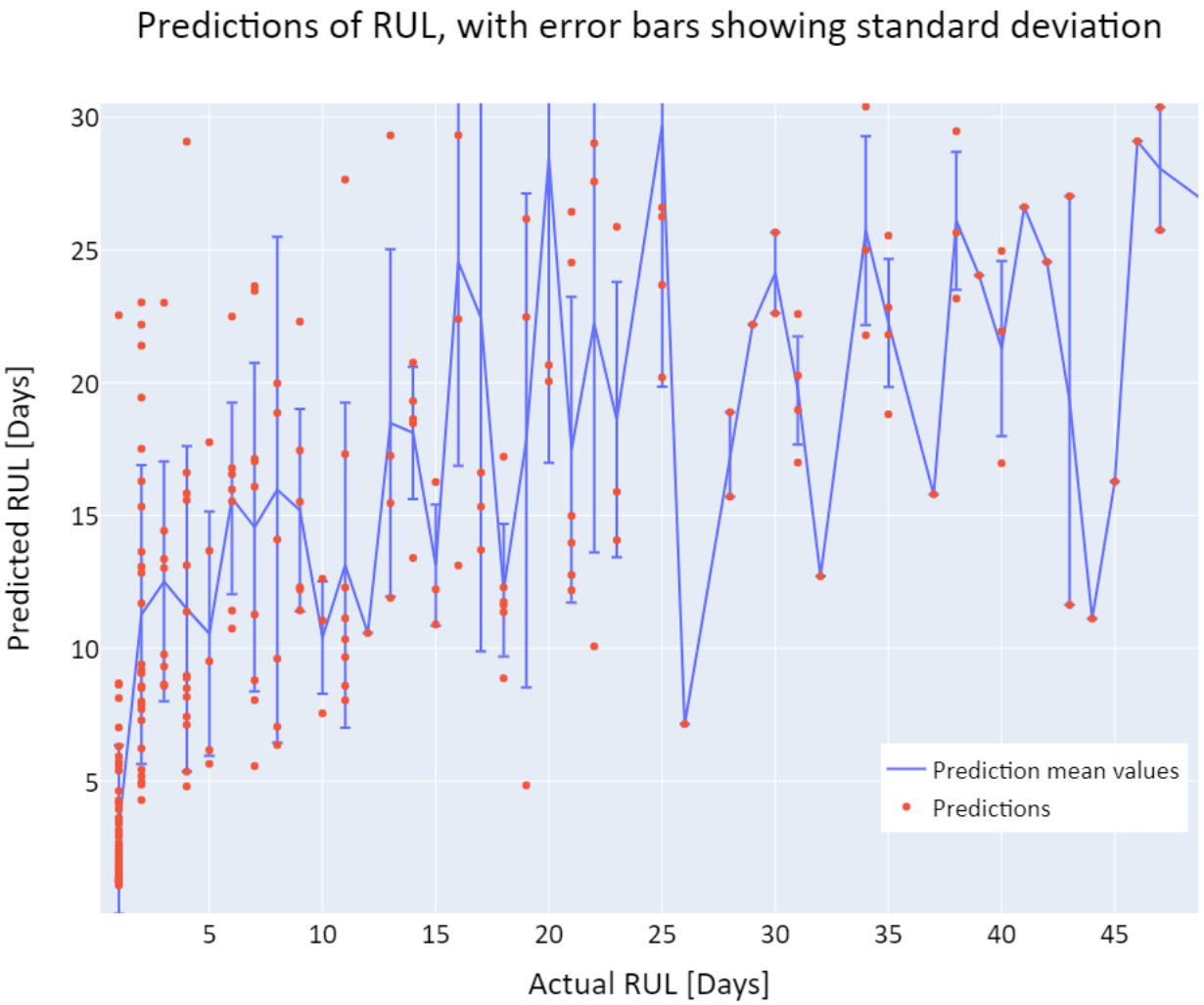

Figure 3.11: The predicted RUL compared to the actual RUL. Error bars indicate the standard deviation of the predicted values

Post-hoc analysis The eight most important features for the prediction, returned by the RFR can be seen in Table 3.7. The clearly most important feature returned is the symmetry of the measured variable Total Capacitive Reactive Energy in the inverter. These are features extracted by the TSFRESH algorithm.

Feature name	Importance
Total_Capacitive_Reactive_Energy_in_the_inverter_symmetry_looking_r_0.3	0.37
Section_2_Status_symmetry_looking_r_0.1	0.14
Liquid_Cooling_flow_variance_larger_than_standard_deviation	0.13
Section_2_DC_Power_Measurement_has_duplicate	0.08
Section_1_Phase_3_power_module_temperature_has_duplicate	0.07
Line_Voltage_Measurement_of_Phases_2_and_3_large_standard_deviation_r_0.1	0.07
Section_2_Phase_2_power_module_temperature_large_standard_deviation_r_0.25	0.06

Table 3.7: A list of the most important features as well as their importance returned by the random forest regressor

For the RFR approach, PCA was also tested on the output of both the raw feature extraction and the feature selection, reducing the runtime of the RFR significantly. According to the explained variance plot, a low number (6) of features were required to describe the full variance

of the dataset. The results, however, demonstrated a worse performance than the variant without PCA. Hence, PCA was not used in the final prediction.

3.3 Benchmarking the approaches

In order to establish a fair comparison of the performances achieved with the two approaches presented in the thesis, and to compare their results, a benchmarking experiment is conducted. In particular, the experiment translates predictions of RUL from the RFR, and hidden states from the HMM into risk classes. These classes are used to indicate the risk of failure predicted by the algorithms. There are three risk classes, as described in Section 2.1. In contrast to RULs, discrete risk classes represent a potential decision taken for the purpose of predictive maintenance, e.g.

- Ordinary operation, low risk of failure (risk 0)
- Warning state, medium risk of failure (risk 1)
- Maintenance required, high risk of failure (risk 2)

In terms of RULs and hidden states, the three classes were defined as:

Prediction	**RFR**	**HMM**
Risk 1	$[0, 5]$	$[8, 10]$
Risk 2	$(5, 20]$	$[5, 7]$
Risk 3	$(20, \infty)$	$[1, 4]$

3.3.1 Results

The micro F1 scores produced by the two models in their benchmark for the test data were as follows:

- PCA and HMM: 0.42
- TSFRESH and RFR: 0.61

Confusion matrices for both approaches are shown in Fig 3.13 and 3.12. The figures show that the RFR manages to distinguish class 2 and 3 fairly well, but has a lot of confusion in true class 1, where it predicts class 2 almost as many times as class 1. Class 1 is only predicted once when the ground truth is class 2 and 3, and class 3 is rarely predicted when class 1 is the ground truth. As for the HMM, it has several false positives and false negatives where the true class is class 1. This could be due to the occurrences of low RULs in states 1,3,5 and 6, combined with the upper quantile of RULs in states 8-10, as seen in Figure 3.7. However, Class 2 and 3 are distinguished well.
Figure 3.12 shows that when the actual RUL in the training data is below 20 days (Risk class 1 and 2), the RFR will predict a RUL below 20 in 164 out of 187 times. Overall, when the model predicts risk class 1 and 2, the actual risk class is the same for 164 out of 183 predictions. This shows a high degree of reliability for this model. Meanwhile, Figure 3.13 shows that the HMM predicts class 1 or 2 for 162 out of 187 actual samples in these classes. However, it predicts class 1 and 2 incorrectly 32 times.

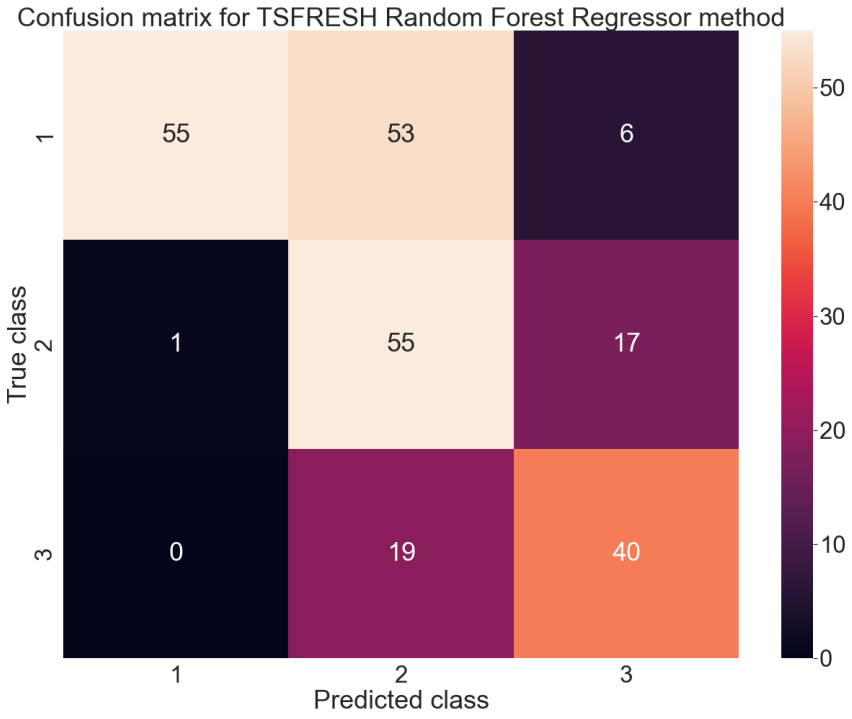

Figure 3.12: A confusion matrix showing the predicted classes from the RFR compared to the true classes for the test samples

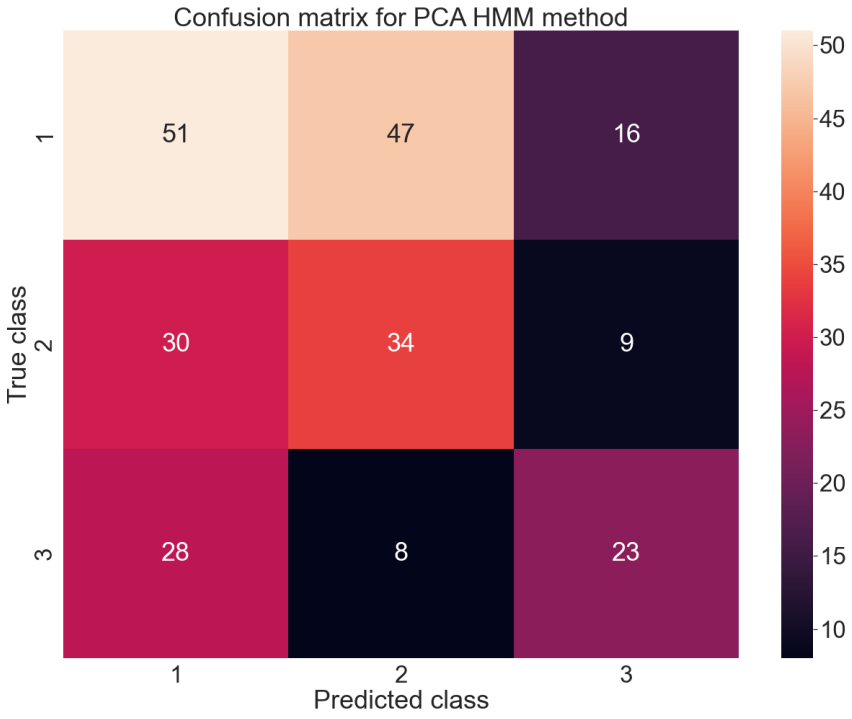

Figure 3.13: A confusion matrix showing the predicted classes from the HMM compared to the true classes for the test samples

3.4 Hardware and runtimes

3.4.1 Hardware

The computer in use is a Dell Latitude 7400, with a Intel(R) Core(TM) i78665U CPU @ 1.90 GHz processor with a base speed of 2.11 GHz, x64-based. The cache sizes are as follows:

- L1 Cache - 256 kB
- L2 Cache - 1,0 MB
- L3 Cache - 8,0 MB

The RAM size of the computer is 16 GB, DDR4. The hard drive of the computer is a 512 GB Micron 2200S NVMe512GB. Running time on the various approaches and steps can be seen in Table 3.8, and shows that the TSFRESH feature extraction took a substantial amount of time for the full dataset.

Runtimes in seconds [s]	**TSFRESH + RFR**	**PCA + HMM**
Feature extraction	313274	637
Model training	241	7456

Table 3.8: The runtimes for feature extraction and model training

Clearly, TSFRESH is the major factor impacting runtime in the machine-learning based pipeline, which is significantly slower than the PCA-based feature extraction in the statistical approach. However, HMM training requires a longer time than training the RFR, given the presented parameter settings.

Chapter 4

Discussion

This chapter presents an argumentation of the various aspects related to the methods used in this thesis work, as well as the alternatives. Results will also be interpreted and discussed in this chapter.

4.1 Data preprocessing and analysis setup

The outlier removal was done in the analysis step instead of the data importation step due to the importance of data visualisation in outlier removal. The downside to outlier removal is the potential removal of important data. Therefore, the approach of removing outliers based on box and whiskers plots are a method that minimises the amount of important data removed. It was shown through the improvement in predictive performance after the introduction of outlier removal that the outlier removal helped the analysis significantly. The amount of removed data was optimised with regard to the predictive performance.

An advantage when training machine learning algorithms and models is to ensure that the dataset is balanced. For this thesis, the distribution of maintenance cycle lengths is heavily biased towards the lower end of the scale, as shown in Figure 3.1. The large number of short maintenance cycles has a potential impact on the predictive performance of the methods presented, i.e. their reliability could be low when the RUL is large. On the other hand, this distribution could increase the prediction accuracy when the RUL is low, since there are more training data in this part of the maintenance cycle.

The holdout method used in this thesis was a rather simple one, only including training and test data, with a size ratio of 0.3. Splitting the data in three was considered, to be able to have an unseen validation data validation set in addition to the two mentioned above. This was however discarded, as the amount of data is limited in this thesis; An additional split would potentially affect the quality of model training. At the same time, the number of hyperparameters to tune in this thesis is arguably low, compared to other existing models such as ANN (Artificial Neural Networks) etc. Nevertheless, a cross-validation could have been possible, but was not used due to the stable performance for the models across several hyperparameter setups.

A key assumption in this thesis is that two maintenance cycles are independent of each other. This means that different maintenance cycles for one equipment are treated in the same way as maintenance cycles from distinct inverters. This could lead to important information not being taken into account for in the RUL prediction:

- The inverter might have some ageing effect, shortening the maintenance cycles when

the inverter ages

- Different inverters might have individual differences. This could include service history, production variations, etc.

For instance, if an inverter is located in a place with heavy wind or sand accumulation, hence has a quicker degradation, the algorithms are not able to take this information into account. The age and history of the inverters are also not taken into account as a consequence of this assumption. However, parts of this information is indirectly provided by the model input data: the number of days the inverter has been on and the number of days the inverter has been feeding power into the grid are two examples. This provides some information about the age of the inverter.

An essential decision was to determine that the highest ending point for a maintenance cycle is 50 days before failure, when truncating the maintenance cycles. This results in a maximum number ground truth RUL of 50 days. The reason for this decision was to be able to capture the degradation better in long maintenance cycles, as these cycles would show how the equipment degrades from ageing. Another aspect is that degradation is most likely less visible for larger RUL values. The downside to this is the lack of testing predictions for RUL$>$ 50, especially for the TSFRESH - RFR method. It also means that the predictions from the algorithm built in this thesis could be unreliable when RUL $>$ 50.

4.2 PCA and Hidden Markov Model

One crucial assumption in the HMM is that the current state is only dependent on the previous state in the sequence. This is known as the first Markov property, as presented in Chapter 2.2.2 it is assumed once the HMM is used. In a degradation process, this assumption might be dependent on the error mechanism. If a component is slowly degrading and ageing, it is fair to assume that the degree of degradation only will increase over time. This will also be the case if the degradation happens at a uniform rate. For these cases, the only relevant state of degradation is the current state. For an error event occurring in a volatile manner, the first Markov property may not be fulfilled, as the speed of the degradation prior to the current time could be relevant to detect a trend in the degradation. For the error type chosen in this thesis, instances of both cases seem to be present in the data - as the length of maintenance cycles ranges between 1 and 600 days. This taken into account, the HMM may not be able to give an accurate state representing the probability of failure for maintenance cycles where the degradation is volatile. This could be part of the reason why there are some failures occurring while the model is predicting a low state number.

An important aspect to the HMM is the start probability matrix, which is specified as user input and not trained by the model. The probability of starting in state 1 was set to 100% by the author, but this may not always be the case in real-life applications. For any installed part, there are scenarios where the initiation in state 1 may not be accurate. The first scenario is one where the part is faulty from the manufacturer due to production errors, or damage in transportation and storage. The probability of such events are low, but present. The second scenario is if the part is installed in a faulty manner. This is a type of error that is always present when humans are involved [19].

As seen in chapter 3.2.2, the HMM does not deliver accurate RUL predictions using the methods presented in this thesis. Even though a correlation between the predicted RUL and

the ground truth can be identified as shown in Figure 3.10, there are multiple of predictions of RUL= 0 when actual RUL is high, and vice versa. However, an interesting aspect shown in Figures 3.5 and 3.7 is the correlation between the predicted states and the actual RUL.

HMM being a statistical model, deep insights can be gained from the trends and indicators leading up until the time of failure. A clear tendency is that the predicted state is increasing while the RUL is decreasing. The final three states 8-10 have a low RUL. These could be used for indicating that there is an increased risk for failure in the inverter, although not covering all single errors exhaustively.

An interesting observation is that single, lower states, such as state 5 in Figure 3.7 also has a distribution of low RUL. State 5 could be interpreted to represent the increased risk of failure in the earlier stages of component lifetime, as seen in Figure 1.1. However, it is questionable whether the Weibull bathtub curve actually is applicable to the case at hand, as only a part of the inverter is changed on failure, and not the full inverter. Although the bathtub curve could be applicable to the part installed, every maintenance cycle will not have a full bathtub curve for the inverter itself. Nevertheless, the risk of failure will always be present. The Sankey plot in Figure 3.9 indicates that few maintenance cycles actually end in state 5 for the test data, although it is clear from the low mean RUL in state 5 that a failure is imminent in this state. Together with Maintenance cycle 1 in Figure 3.5 failing at an early point in time while in state 5, this strengthens the theory about state 5 indicating a risk of early failure. One can also see in Figure 3.5 that there exists some low RUL values in the lower state numbers. This could be owed to the fact that the risk of failure is always present, and some failures will happen unexpectedly. Some of the low RUL values shown in low states could also be caused by the imbalanced dataset in use.

The strong correlation between predicted hidden state and actual RUL indicates that the predicted hidden state could be used to indicate risk of failure - indicating for the plant personnel that the risk of failure is high and the equipment should be inspected when the predicted state is for instance 5 or 8-10. Therefore, the HMM could be useful for predictive maintenance, despite not providing a specific prediction on RUL. A downside to the approach of PCA and HMM is that it is not possible to trace back the model prediction to a particular feature due to the PCA transformation. The new representation of features from the PCA together with the complicated interpretation of a HMM gives the user little to no information about the key factors leading to the decision in the process. This is not optimal, since the user benefits from insights into feature relevance in the model.

4.3 Time series feature extraction combined with Random Forest Regression

The pipeline using TSFRESH combined with RFR (Random Forest Regressor) predicted directly the RUL on a random time in the maintenance cycles. For the TSFRESH feature extraction, all functions were implemented on all sensors, leading to an immense amount of features extracted. Even though possibilities existed to select which functions to use on which sensor, these were not exploited since it could potentially lead to important information not being extracted. However, a reduced number of features would have reduced the computational burden associated with the method.

Considering the decision of hyperparameters for the RFR based on performance on the test data, as shown in Chapter 3.2.3, there was indeed a risk of the model overfitting to the test data. However, the difference in performance were not significant for the different hyperparameters tuned, as seen in Tables 3.5 and, 3.6. Therefore, the degree overfitting towards the test data is to be considered neglectable for the RFR model used here.

In comparison with other regression algorithms, the performance of the RFR is higher than any other regression algorithm tested for the prediction of RUL, arguably because of the nonlinear nature of the RFR. Preliminary tests were conducted using: SVR (Support Vector Regressor), Linear Regression and RNN (Recurrent Neural Networks). Despite the clear correlation between the ground truth and predicted RUL, there are some inaccurate predictions. Prediction errors are obviously correlated with the RUL: the longer the RUL, the higher the number of outlier predictions. The inaccuracy for higher RULs could be caused by the algorithm not being able to detect degradation due to its lack of presence at early points in the maintenance cycle. Alternatively, it could also be due to inadequate training data caused by the unbalanced distribution of cycle lengths. The imbalanced distribution could also be the reason for the strong correlation between prediction and ground truth towards RUL $= 0$, as the majority of maintenance cycles range in this period. The problem of imbalanced distribution could have been resolved by re-using long maintenance cycles multiple times; ending a maintenance cycle at several random points p_i and treating the new maintenance cycles as independent ones. The downside to this would be that the algorithm could have overfitted towards unique trends in individual maintenance cycles, making the generalisation poor. In addition, the assumption that maintenance cycles are independent would be breached. Synthetic measurements were also considered for balancing the dataset, but this was not pursued due to the risk of unrealistic samples being generated.

A positive side to the RFR and TSFRESH method is that it offers a clear indication of feature importance. The TSFRESH algorithm pursues a consistent naming scheme based on features from the original input data, making it possible to identify both the important measurement variables to monitor, and which events are acting as degrading for the object. For instance, if the number of peaks in current is an important feature to predict the RUL, it is an indication that peaks in current are events that are degrading the equipment and eventually contribute to the failure occurring.

Regarding the feature importance returned by the RFR, a major point of criticism is that they can be affected by random correlation [78]. This is logical, as the selection of features for any given Decision Tree in the Random Forest based on a random initialisation [53]. A suggestion of making this more reliable comes from the authors of [78], saying that a "drop-

out" procedure for columns should be followed when training a Random Forest algorithm. This is done by training one Random Forest model while leaving out one column at a time, resulting in training as many RFR models to train as there is features in the input dataset. The suggested approach was considered for this thesis, but could not be pursued due to the vast amount of features put into the RFR (10970), which would lead to an extreme time consumption for the algorithm, which makes it less feasible. One can indeed see that there is likely to be some spurious correlations in Table 3.7, confirming the theory. In line 4 and 5 from the top, there are features measuring the number of duplicate values in a feature. While this could be an indication of how long the equipment has been running, given that the occurrence of a duplicate value is random, this would lead to the amount of duplicates increasing over time. From another viewpoint, duplicate measurements in module temperature and power measurement are not a choice of features to indicate equipment degradation.

4.4 Method comparison

Summarised, both methods tested in this thesis have their strengths and drawbacks. The HMM gives a good understanding of the dynamic of the risk of failure along the time axis. Based on the predicted state in a given point in time, one can make assumptions about the risk of failure. Meanwhile, the weaknesses of the HMM method is the strength of the RFR method; the RFR have transparency when it comes to the factors contributing to the predicted results, and it predicts the estimated RUL in days. It has a strong predictive performance when the RUL approaches 0, although there are inaccuracies when the RUL is higher. In fact, the RFR approach has overall a considerably better performance than the HMM approach according to the benchmark results, despite some less confusion between class 2 and 3. When considering the benchmark results for the methods presented in Section

3.3.1, it is worth bearing in mind that the interpretation is more rigid when evaluating results of classification. Whereas the evaluation of regression algorithms takes into account *how* wrong the prediction of the model is, while a classification evaluation returns either right or wrong. Therefore, the results presented for the common benchmark method may be slightly conservative, and highly affected by the class limits set.

Chapter 5

Summary and conclusion

In this thesis, two different methods for estimating risk of failure for predictive maintenance in solar plant inverters are presented, evaluated and compared.
Both methods had the same data input; sensor data aggregated to 30 minute periods, split into maintenance cycles based on occurrence of a certain failure type. Irrelevant features and outlier data rows were removed, as well as samples from periods with no power production. The maintenance cycles were truncated to end at random points, to represent the actual use of the methods; estimating RUL (Residual Useful Lifetime) or risk of failure at a given point prior to failure. The first method includes a PCA (Principal Component Analysis) providing input to a HMM (Hidden Markov Model). The PCA reduces the number of input features for the HMM, which has optimal performance on a low number of input features. This model returns 10 different states, representing the risk of failure at a given point in time. The second method consist of a time series feature extraction function called TSFRESH (Time Series Feature Extraction based on Scalable Hypotesis tests), summarising a maintenance cycle with several rows of time steps into one row with an increased amount of columns compared to the original. The output from TSFRESH was used in a RFR (Random Forest Regressor), estimating the RUL for the equipment.

The results from the methods tested this thesis shows that there exists possibilities in implementing methods for predictive maintenance in solar plants. It also indicates that the semantic and pragmatic quality of the data being samples on site today is sufficient for estimating the time until a certain failure occurs. Therefore, the cost of installing new sensors can be saved.

For the case and failure type studied in this thesis, the TSFRESH combined with RFR could be used for estimating the RUL for the determined failure type. Despite some outliers in the predicted value, the prediction as the RUL approaches 0 increases in quality. This indicates that there exists information about degradation in the data.

The HMM can be used for indicating the risk of failure at a given time in the maintenance cycle. A maintenance cycle entering state 5, 8, 9 and 10 indicates that there is increased risk of failure, and that the equipment should be monitored closely.

Individually or combined, the experiments in this thesis has shown that the methods presented are able to provide useful information about the condition of the component, and deliver early warnings of impending failures. Such warnings could lead to increased preparedness for impending failures among the plant personnel, potential for early intervention to avoid failure, or indication that a risk-based inspection should be performed on the equipment. In addition, the feature importance indicates which features are critical to monitor in order

to indicate the equipment health with regards to the determined error type. Therefore, the conclusion is that both methods serves as proofs of concept, and could be applied for predictive maintenance in the solar industry. Nevertheless, there will always occur failures that are unexpected, but the methods presented in this thesis could help in giving an early warning for some occurrences.

5.1 Future work

The proofs of concept shown in this thesis could be a first step towards implementing predictive maintenance for degradation related failures in solar plants. Together with domain experts, failure types relevant for predictive maintenance could be identified and pursued. Failure events should also be validated against the downtime registry per inverter.

The methods in this thesis could be tested on data from various locations individually, and evaluated. If the results are promising, there could be potential for implementing the presented methods in e.g. the SCADA system, and from there indicate to the CMC or the plant personnel which devices are at risk of failure.

The methods of this thesis could also be refined and fine-tuned; for instance, several approaches for the re-sampling of maintenance cycles could be tested, as well as up-sampling of maintenance cycles with high duration.

Features identified as important from the Random Forest could be advantageously analysed, and methods suggested by [78] applied to find the reliability of the feature importance found in this thesis.

The input data for analysis in this thesis could be combined with data from the EAM (Enterprise Asset Management) system to train models based on what work has been conducted to correct failures. In turn, the resulting models could be used to provide further information than provided in this thesis; namely what is the action needed to prevent the impending failure, which is known as Prescriptive maintenance.

Bibliography

[1] A. F. Skomedal, "Data-based approaches to efficient operation and maintenance of pv systems." University of Oslo, Department of Technology systems, Philosophiae Doctor Thesis, 2021.

[2] United Nations, "The Paris Agreement," WEB, (Last read: 14.01.2022). [Online]. Available: https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement

[3] International Energy agency, "World Energy Outlook," Report, 2012.

[4] S. Berlijn, "Digital Electric power systems," Lecture, (NMBU, 05.09.2019).

[5] M. Jaganmohan, "Cumulative installed solar PV capacity worldwide from 2000 to 2020," Statista.com, 2021.

[6] Scatec ASA, "Scatec ASA," Online, last read 14.01.2022. [Online]. Available: Scatec.com

[7] A. Von Meier, *Electric power systems: a conceptual introduction.* John Wiley & Sons, 2006.

[8] M. Baptista, S. Sankararaman, I. P. de Medeiros, C. Nascimento, H. Prendinger, and E. M. Henriques, "Forecasting fault events for predictive maintenance using data-driven techniques and arma modeling," *Computers Industrial Engineering*, vol. 115, pp. 41–53, 2018.

[9] P.I.Bye, *Vedlikehold og driftssikkerhet*, 2009.

[10] V. Devabhaktuni, M. Alam, S. Shekara Sreenadh Reddy Depuru, R. C. Green, D. Nims, and C. Near, "Solar energy: Trends and enabling technologies," *Renewable and Sustainable Energy Reviews*, vol. 19, pp. 555–564, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032112006363

[11] M. H. Tesfazion, "A data driven approach for power loss detection in utility scale solar power plants," *Master's Thesis 2019, Faculty of Science and Technology, NMBU.*

[12] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 812–820, 2015.

[13] S. Voronov, *Machine Learning Models for Predictive Maintenance*, ser. Linköping Studies in Science and Technology. Dissertations. Linköping, 2020.

[14] J.Nicholas, *Lean production for competitive advantage.* CRC press, 2011.

[15] "Praktisk prosjektledelse : fra idé til gevinst," Bergen, 2014.

[16] G. Susto, J. Wan, S. Pampuri, M. Zanon, A. B. Johnston, P. O'Hara, and S. McLoone, "An adaptive machine learning decision system for flexible predictive maintenance," in *IEEE International Conference on Automation Science and Engineering (CASE)*, 2014, pp. 806–811.

[17] G. A. Susto, S. McLoone, D. Pagano, A. Schirru, S. Pampuri, and A. Beghi, "Prediction of integral type failures in semiconductor manufacturing through classification methods," in *IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA)*, 2013, pp. 1–4.

[18] E. P. Management, "Reducing operations & maintenance costs," White paper, 2003, (Last read: 05.10.2021).

[19] J. Reason, *Human error*. Cambridge university press, 1990.

[20] A.-J. Selstad and G. S. Gedde-Dahl, "The application of parameters in predictive maintenance and operational optimalization." NTNU - Department of Mechanical and Industrial Engineering, bachelor thesis, 2019.

[21] L. B. Bosman, W. Leon-Salas, W. Hutzel, and E. A. Soto, "Pv system predictive maintenance: Challenges, current approaches, and opportunities." Energies, 2019.

[22] D. GL, "A dnv gl group technology research and greenpowermonitor position paper 2021: Predictive maintenance of solar pv plants: The time is now." DNV, 2021.

[23] L. Ren, Y. Sun, J. Cui, and L. Zhang, "Bearing remaining useful life prediction based on deep autoencoder and deep neural networks," *Journal of Manufacturing Systems*, vol. 48, pp. 71–77, 2018, special Issue on Smart Manufacturing.

[24] T. Tinga and R. Loendersloot, *Physical Model-Based Prognostics and Health Monitoring to Enable Predictive Maintenance*. Cham: Springer International Publishing, 2019, pp. 313–353. [Online]. Available: https://doi.org/10.1007/978-3-030-05645-2_11

[25] L. P. Silvestrin, M. Hoogendoorn, and G. Koole, "A comparative study of state-of-the-art machine learning algorithms for predictive maintenance." in *SSCI*, 2019, pp. 760–767.

[26] T. P. Carvalho, F. A. A. M. N. Soares, R. Vita, R. da P. Francisco, J. P. Basto, and S. G. S. Alcalá, "A systematic literature review of machine learning methods applied to predictive maintenance," *Computers Industrial Engineering*, vol. 137, p. 106024, 2019.

[27] S. Butte, A. Prashanth, and S. Patil, "Machine learning based predictive maintenance strategy: A super learning approach with deep neural networks," in *IEEE Workshop on Microelectronics and Electron Devices (WMED)*, 2018, pp. 1–5.

[28] T. Huuhtanen and A. Jung, "Predictive maintenance of photovoltaic panels via deep learning," in *IEEE Data Science Workshop (DSW)*, 2018, pp. 66–70.

[29] N. Kolokas, T. Vafeiadis, D. Ioannidis, and D. Tzovaras, "Forecasting faults of industrial equipment using machine learning classifiers," in *Innovations in Intelligent Systems and Applications (INISTA)*, 2018, pp. 1–6.

[30] G. K. Durbhaka and B. Selvaraj, "Predictive maintenance for wind turbine diagnostics using vibration signal analysis based on collaborative recommendation approach," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016, pp. 1839–1842.

[31] T. Praveenkumar, M. Saimurugan, P. Krishnakumar, and K. Ramachandran, "Fault diagnosis of automobile gearbox based on machine learning techniques," *Procedia Engineering*, vol. 97, pp. 2092–2098, 2014, "12th Global Congress on Manufacturing and Management" GCMM - 2014.

[32] I. Amihai, M. Chioua, R. Gitzel, A. M. Kotriwala, D. Pareschi, G. Sosale, and S. Subbiah, "Modeling machine health using gated recurrent units with entity embeddings and k-means clustering," in *16th International Conference on Industrial Informatics (INDIN)*, 2018, pp. 212–217.

[33] V. Mathew, T. Toby, V. Singh, B. M. Rao, and M. G. Kumar, "Prediction of remaining useful lifetime (rul) of turbofan engine using machine learning," in *IEEE International Conference on Circuits and Systems (ICCS)*, 2017, pp. 306–311.

[34] J. Zenisek, F. Holzinger, and M. Affenzeller, "Machine learning based concept drift detection for predictive maintenance," *Computers Industrial Engineering*, vol. 137, p. 106031, 2019.

[35] J. Deutsch and D. He, "Using deep learning-based approach to predict remaining useful life of rotating components," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 1, pp. 11–20, 2018.

[36] D. Wu, C.Jennings, J. Terpenny, R. X. Gao, and S. Kumara, "A Comparative Study on Machine Learning Algorithms for Smart Manufacturing: Tool Wear Prediction Using Random Forests," *Journal of Manufacturing Science and Engineering*, vol. 139, no. 7, 04 2017, 071018. [Online]. Available: https://doi.org/10.1115/1.4036350

[37] A. Giantomassi, F. Ferracuti, A. Benini, S. Longhi, G. Ippoliti, and A. Petrucci, "Hidden markov model for health estimation and prognosis of turbofan engines," vol. 3, 01 2011, pp. 1–6.

[38] P. Bulanyi and R. Zhang, "Shading analysis & improvement for distributed residential grid-connected photovoltaics systems," in *The 52nd Annual Conference of the Australian Solar Council*, 2014.

[39] B. Figgis, A. Ennaoui, S. Ahzi, and Y. Rémond, "Review of pv soiling measurement methods," in *2016 International Renewable and Sustainable Energy Conference (IRSEC)*. IEEE, 2016, pp. 176–180.

[40] ——, "Review of pv soiling measurement methods," in *2016 International Renewable and Sustainable Energy Conference (IRSEC)*. IEEE, 2016, pp. 176–180.

[41] M. Villarini, V. Cesarotti, L. Alfonsi, and V. Introna, "Optimization of photovoltaic maintenance plan by means of a fmea approach based on real data," *Energy Conversion and Management*, vol. 152, pp. 1–12, 2017.

[42] D. Lazos, A. B. Sproul, and M. Kay, "Development of hybrid numerical and statistical short term horizon weather prediction models for building energy management optimisation," *Building and Environment*, vol. 90, pp. 82–95, 2015.

[43] A. Möller and J. Groß, "Probabilistic temperature forecasting based on an ensemble autoregressive modification," *Quarterly Journal of the Royal Meteorological Society*, vol. 142, no. 696, pp. 1385–1394, 2016.

[44] R. Anand, R. K. Pachauri, A. Gupta, and Y. K. Chauhan, "Design and analysis of a low cost pv analyzer using arduino uno," in *2016 IEEE 1st international conference on power electronics, intelligent control and energy systems (ICPEICES)*. IEEE, 2016, pp. 1–4.

[45] P. Guerriero, F. Di Napoli, G. Vallone, V. d'Alessandro, and S. Daliento, "Monitoring and diagnostics of pv plants by a wireless self-powered sensor for individual panels," *IEEE Journal of Photovoltaics*, vol. 6, no. 1, pp. 286–294, 2015.

[46] S. Adhya, D. Saha, A. Das, J. Jana, and H. Saha, "An iot based smart solar photovoltaic remote monitoring and control unit," in *2016 2nd international conference on control, instrumentation, energy & communication (CIEC)*. IEEE, 2016, pp. 432–436.

[47] S. Silvestre, A. Chouder, and E. Karatepe, "Automatic fault detection in grid connected pv systems," *Solar Energy*, vol. 94, pp. 119–127, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0038092X13001849

[48] A. Rivai, N. Abd Rahim, M. F. Mohamad Elias, and J. Jamaludin, "Analysis of photovoltaic string failure and health monitoring with module fault identification," *Energies*, vol. 13, no. 1, 2020. [Online]. Available: https://www.mdpi.com/1996-1073/13/1/100

[49] M. De Benedetti, F. Leonardi, F. Messina, C. Santoro, and A. Vasilakos, "Anomaly detection and predictive maintenance for photovoltaic systems," *Neurocomputing*, vol. 310, pp. 59–68, 2018.

[50] S. R. Madeti and S. Singh, "Monitoring system for photovoltaic plants: A review," *Renewable and Sustainable Energy Reviews*, vol. 67, pp. 1180–1207, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032116305792

[51] R. Platon, J. Martel, N. Woodruff, and T. Y. Chau, "Online fault detection in pv systems," *IEEE Transactions on Sustainable Energy*, vol. 6, no. 4, pp. 1200–1207, 2015.

[52] T. Westerkamp, *10 Steps to Setting Up Your Own Predictive Maintenance Program*. BNI Building News, 2013, vol. 5.

[53] V. M. S. Raschka, *Python Machine Learning*. Packt Publishing - ebooks Account, 2018.

[54] M. Grandini, E. Bagli, and G. Visani, "Metrics for multi-class classification: an overview," 2020. [Online]. Available: https://arxiv.org/abs/2008.05756

[55] A. Giantomassi, F. Ferracuti, A. Benini, S. Longhi, G. Ippoliti, and A. Petrucci, "Hidden markov model for health estimation and prognosis of turbofan engines," vol. 3, 01 2011, pp. 1–6.

[56] Katrine Frey Frøslie, "Markov-prosess i store norske leksikon," https://snl.no/Markov-prosess, (Last read: 24.04.2022).

[57] P. Hofmann and Z. Tashman, "Hidden markov models and their application for predicting failure events," in *International Conference on Computational Science*. Springer, 2020, pp. 464–477.

[58] A. Simões, J. M. Viegas, J. T. Farinha, and I. Fonseca, "The state of the art of hidden markov models for predictive maintenance of diesel engines," *Quality and reliability engineering international*, vol. 33, no. 8, pp. 2765–2779, 2017.

[59] A. Marjanović, G. Kvaščev, P. Tadić, and Ž. Đurović, "Applications of predictive maintenance techniques in industrial systems," *Serbian Journal of Electrical Engineering*, vol. 8, no. 3, pp. 263–279, 2011.

[60] A. Martins, I. Fonseca, J. T. Farinha, J. Reis, and A. J. M. Cardoso, "Maintenance prediction through sensing using hidden markov models—a case study," *Applied Sciences*, vol. 11, no. 16, p. 7685, 2021.

[61] F. Cartella, J. Lemeire, L. Dimiccoli, and H. Sahli, "Hidden semi-markov models for predictive maintenance," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[62] M. Tadayon and G. Pottie, "Comparative analysis of the hidden markov model and lstm: A simulative approach," *arXiv preprint arXiv:2008.03825*, 2020.

[63] J. Henderson, S. Salzberg, and K. H. Fasman, "Finding genes in dna with a hidden markov model," *Journal of Computational Biology*, vol. 4, no. 2, pp. 127–141, 1997.

[64] B. Schuller, G. Rigoll, and M. Lang, "Hidden markov model-based speech emotion recognition," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, vol. 2, 2003, pp. II–1.

[65] J. Yang, Y. Xu, and C. Chen, "Human action learning via hidden markov model," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 27, no. 1, pp. 34–44, 1997.

[66] T. Harris and J. W. Hardin, "Exact wilcoxon signed-rank and wilcoxon mann–whitney ranksum tests," *The Stata Journal*, vol. 13, no. 2, pp. 337–343, 2013. [Online]. Available: https://doi.org/10.1177/1536867X1301300208

[67] T. J. Cleophas, "Statistics applied to clinical studies," Dordrecht, 2012.

[68] Katrine Frey Frøslie, "Monte carlo-metode i store norske leksikon," https://snl.no/Monte_Carlo-metode, (Last read: 30.04.2022).

[69] J. Norris, "Markov chain monte carlo," *Statistical Laboratory*, 2004.

[70] J. Zheng, J. Huang, and C. Tong, "The order estimation for hidden markov models," *Physica A: Statistical Mechanics and its Applications*, vol. 527, p. 121462, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378437119308519

[71] I. Votsi, N. Limnios, G. Tsaklidis, and E. Papadimitriou, "Hidden semi-markov modeling for the estimation of earthquake occurrence rates," *Communications in Statistics - Theory and Methods*, vol. 43, no. 7, pp. 1484–1502, 2014. [Online]. Available: https://doi.org/10.1080/03610926.2013.857414

[72] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package)," *Neurocomputing*, vol. 307, pp. 72–77, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218304843

[73] M. Christ, A. W. Kempa-Liehr, and M. Feindt, "Distributed and parallel time series feature extraction for industrial big data applications," *arXiv preprint arXiv:1610.07717*, 2016.

[74] H. Y. Teh, K. I.-K. Wang, and A. W. Kempa-Liehr, "Expect the unexpected: Unsupervised feature selection for automated sensor anomaly detection," *IEEE Sensors Journal*, vol. 21, no. 16, pp. 18 033–18 046, 2021.

[75] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency," *The Annals of Statistics*, vol. 29, no. 4, pp. 1165–1188, 2001. [Online]. Available: http://www.jstor.org/stable/2674075

[76] M. Christ, F. Kienle, and A. Kempa-Liehr, "Time series analysis in industrial applications," in *Workshop on Extreme Value and Time Series Analysis*, 2016.

[77] G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, no. 2, pp. 197–227, 2016.

[78] T. Parr, J. D. Wilson, and J. Hamrick, "Nonparametric feature impact and importance," *arXiv preprint arXiv:2006.04750*, 2020.

[79] M. Christ, et al., "TSFRESH: Overview on extracted features," Python Documentation, (Last read: 23.04.2022). [Online]. Available: https://tsfresh.readthedocs.io/en/latest/text/list_of_features.html

Appendix A

Glossary

FMEA	Failure Mode and Effect Analysis
RUL	Residual Useful Lifetime
EAM	Enterprise Asset Management system
SCADA	Supervisory Control And Data Acquisition
PV plants	Solar plants using PhotoVoltaic solar cell technology
SOH	State Of Health
HSE	Health, Environment and Safety
CMC	Central Monitoring Centre
The Grid	The power supply in a region or nation
Soiling	The accumulation of dust and sand on solar modules
TSFRESH	Time Series FeatuRe Extraction based on Scalable Hypothesis tests
HMM	Hidden Markov Model
RFR	Random Forest Regressor
MFA	Multi-Factor Authentication
SQL	Structured Query Language

Appendix B

Exhaustive list of attributes generated by TSFRESH

Table B.1 contains all features generated by the TSFRESH algorithm. The list is fetched from [79].

Attribute function	**Description**
abs_energy(x)	Returns absolute energy of the time series feature, the sum over the squared values
absolute_maximum(x)	Returns the highest absolute value of the time series feature
absolute_sum_of_changes(x)	Returns the sum over the absolute value of consecutive changes in the time series feature
agg_autocorrelation(x, param)	Descriptive statistics on the autocorrelation of the time series feature
agg_linear_trend(x, param)	Calculates a linear least-squares regression for values of the time series feature
approximate_entropy(x, m, r)	Vectorized Approximate entropy
ar_coefficient(x, param)	The unconditional maximum likelihood of an autoregressive AR(k) process.
augmented_dickey_fuller(x, param)	Does the time series feature have a unit root?
autocorrelation(x, lag)	The autocorrelation of the specified lag for the time series feature
benford_correlation(x)	Useful for anomaly detection applications
binned_entropy(x, max_bins)	First bins the values of the time series feature into max_bins equidistant bins
c3(x, lag)	Uses c3 statistics to measure non linearity in the time series feature
change_quantiles(x, ql, qh, isabs, f_agg)	First fixes a corridor given by the quantiles ql and qh of the distribution of x.
cid_ce(x, normalize)	An estimate for complexity in the time series feature
count_above(x, t)	The percentage of values in the time series feature that are higher than a threshold t
count_above_mean(x)	The number of values in the time series feature that are higher than its mean
count_below(x, t)	The percentage of values in the time series feature that are lower than s threshold t

count_below_mean(x)	The number of values in the time series feature that are lower than the mean of the time series
cwt_coefficients(x, param)	A Continuous wavelet transform for the Ricker wavelet.
energy_ratio_by_chunks(x, param)	Calculates the sum of squares a data group and express it as the sum of squares over the whole time series feature
fft_aggregated(x, param)	The spectral centroid (mean), variance, skew, and kurtosis of the absolute Fourier transform spectrum
fft_coefficient(x, param)	The Fourier coefficients of the one-dimensional discrete Fourier Transform
first_location_of_maximum(x)	The first location of the time series feature's maximum value
first_location_of_minimum(x)	The first location of the time series feature's minimum value
fourier_entropy(x, bins)	The binned entropy of the power spectral density of the time series feature, using the welch method
friedrich_coefficients(x, param)	Coefficients of a polynomial equation fitted over the time series feature
has_duplicate(x)	If there are duplicates in the time series feature
has_duplicate_max(x)	If there are duplicates for the max value in the time series feature
has_duplicate_min(x)	If there are duplicates for the max value in the time series feature
index_mass_quantile(x, param)	Indicates which quantile the majority of data mass for the time series feature exsist
kurtosis(x)	The degree of kurtosis of the time series feature
large_standard_deviation(x, r)	Whether the time series feature has large standard deviation
last_location_of_maximum(x)	The last location of the maximum value for the time series feature
last_location_of_minimum(x)	The last location of the minimum value for the time series feature
lempel_ziv_complexity(x, bins)	A complexity estimate for the time series feature
length(x)	Returns the length of the time series feature
linear_trend(x, param)	A linear least-squares regression for the values of the time series feature versus the sequence from 0 to length of the time series feature minus one
linear_trend_timewise(x, param)	A linear least-squares regression for the values of the the time series feature versus the sequence from 0 to length of the the time series feature minus one
longest_strike_above_mean(x)	The length of the longest data sequence in the time series feature that is bigger than the mean value of the time series feature
longest_strike_below_mean(x)	The length of the longest data sequence in the time series feature that is smaller than the mean value of the time series feature
matrix_profile(x, param)	The matrix profile mean plus the Tukey's five number set

max_langevin_fixed_point(x, r, m)	Largest fixed point of dynamics in the time series feature estimated from a polynomial describing the time series feature
maximum(x)	The maximum value of the time series feature
mean(x)	The mean value of the time series feature
mean_abs_change(x)	Average over first differences in the time series feature
mean_change(x)	Average over differences the time series feature
mean_n_absolute_max (x, number_of_maxima)	The arithmetic mean of a number of absolute maximum values of the time series feature
mean_second_derivative_central(x)	The mean value of a central approximation of the second derivative
median(x)	The median value of the time series feature
minimum(x)	The minimum value of the time series feature
number_crossing_m(x, m)	The number of times the time series feature has crossed the value m
number_cwt_peaks(x, n)	Number of different peaks in the time series feature
number_peaks(x, n)	Number of peaks in the time series feature
partial_autocorrelation(x, param)	The value of partial autocorrelation function at a given lag
percentage_of_reoccurring_ datapoints_to_all_datapoints(x)	The percentage of recurring data points in the time series feature
percentage_of_reoccurring_ values_to_all_values(x)	Percentage of values that are redundant in the time series feature
permutation_entropy(x, tau, dimension)	Permutation entropy of the time series feature
quantile(x, q)	Calculates the q quantile of the time series feature
query_similarity_count(x, param)	A similarity measure for the time series feature
range_count(x, min, max)	Number of observed numbers in the interval (min, max)
ratio_beyond_r_sigma(x, r)	Ratio of values more than r times the mean of the time series feature
ratio_value_number _to_time_series_length(x)	An indicator of how often recurrent errors occur
root_mean_square(x)	The root mean square of the time series feature
sample_entropy(x)	The sample entropy of the time series feature
set_property(key, value)	A decorator that sets the property key of the function to value
skewness(x)	The skewness of the time series feature
spkt_welch_density(x, param)	The cross power spectral density of the time series feature at different frequencies
standard_deviation(x)	The standard deviation of the time series feature
sum_of_reoccurring_data_points(x)	Sum of all recurrent datapoints in the time series feature
sum_of_reoccurring_values(x)	Sum of all recurrent values in the time series feature
sum_values(x)	The sum of the time series feature

symmetry_looking(x, param)	Boolean telling whether the distribution of the time series feature looks symmetric
time_reversal_asymmetry _statistic(x, lag)	Returns the time reversal asymmetry statistic of the time series feature
value_count(x, value)	Count occurrences of value in the time series feature
variance(x)	Returns the variance of the time series feature
variance_larger_than _standard_deviation(x)	Is variance higher than the standard deviation?
variation_coefficient(x)	Returns the variation coefficient of the time series feature

Table B.1: An exhaustive table showing all the feature extraction functions in the TSFRESH package. Several of the functions return several columns.

Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway