Condition monitoring and failure diagnosis for wind turbines in the context of rare failing and slowly progressing damage

Chesterman, Xavier

Publication date:
2024

VRIJE
UNIVERSITEIT
BRUSSEL

Faculty of Science and
Bio-Engineering Sciences
Department of Computer Science

Condition monitoring and failure diagnosis for wind turbines in the context of rare failing and slowly progressing damage

Dissertation submitted in fulfilment of the requirements for the degree of Doctor of Science: Computer science

Xavier Chesterman

Brussels, September 2024

Promotors: Prof. Dr. Ann Nowé
Prof. Dr. Jan Helsen

List of Jury Members

- Prof. Dr. Ann Nowé (ann.nowe@vub.be), Pleinlaan 9, 1050 Brussels, Belgium
- Prof. Dr. Jan Helsen (jan.helsen@vub.be), Pleinlaan 2, 1050 Brussels, Belgium
- Prof. Dr. Pieter Libin (pieter.libin@vub.be), Pleinlaan 9, 1050 Brussels, Belgium
- Prof. Dr. Coen De Roover (coen.de.roover@vub.be), Pleinlaan 2, 1050 Brussels, Belgium
- Prof. Dr. Bart Goethals (bart.goethals@uantwerpen.be), Middelheimlaan 1, 2020 Antwerp, Belgium
- Prof. Dr. Donatella Zappalá (D.Zappala@tudelft.nl), Postbus 5, 2600 AA Delft, The Netherlands
- Prof. Dr. Amir Nejad (amir.nejad@ntnu.no), Jonsvannsveien 82, B429 Trondheim, Norwey
- Dr. Cédric Peeters (cedric.peeters@vub.be), Pleinlaan 2, 1050 Brussels, Belgium

Summary

It is a well-known fact that the current generation of operational wind turbines suffers from premature component failures. These result often in long downtimes. If it were possible to identify these failures sufficiently in advance, it would be possible to replace or repair the failing components during regular maintenance, which would result in significantly lower maintenance costs and shorter downtimes.

Failure prediction and diagnosis for wind turbines is currently an unsolved question. A useful system should be able to detect many different failure types well in advance. This means it should not just be able to identify the moment a component starts behaving out of the ordinary, but it should also be able to interpret the abnormal behavior.

The improved availability of data has been a blessing and a curse. On the one hand, it has made it possible to analyze the behavior of the wind turbines thoroughly. On the other hand, the large quantity of data has made the task of analyzing and understanding it challenging for experts. An automated approach would solve this problem. Developing such a methodology is currently the topic of state-of-the-art research.

The main goal of the research presented in this thesis is the development of an automated failure detection and diagnosis framework for wind turbine drivetrains. This is done using data that is readily available from wind farms, e.g. 10-minute Supervisory Control And Data Acquisition (SCADA) and status log data. The framework must be able to predict failures of wind turbine drivetrains, e.g. gearbox, generator, ..., well in advance by analyzing component temperatures. It must also be able to determine the failure mode by analyzing the patterns in the identified abnormal behavior.

The framework is a 4-step pipeline consisting of several artificial intelligence (AI) techniques that are used sequentially. The first step uses machine learning (ML) techniques to model the normal component temperature behavior of wind turbines given their

operational state. In the second step, statistical models, i.e. CUSUM, MAD-IOD are used to identify anomalies in the difference between the observed and predicted normal component temperatures. This information is combined in the third step with information from the status logs. Pattern and association rule mining techniques are used to identify patterns or rules that can be associated with drivetrain errors or failures. The patterns or rules are after postprocessing used for failure diagnosis.

The validation, which is done on data from three operational wind farms, shows that the best-performing failure prediction methodology can detect failures accurately and well in advance. It is now used periodically to monitor an operational wind farm at the specific request of the wind turbine operator. The best failure diagnosis methodology succeeds in identifying patterns that are related to certain failure modes. These patterns can be used to make an accurate failure diagnosis.

The thesis is structured as follows. It starts with an introduction that describes the general problem, the problem statement, and research questions. The first chapter of the thesis gives a description of the data characteristics, the failure properties, and two wind turbine ontologies. The second chapter discusses the failure prediction methodology. Several different methodologies are presented and compared. The third chapter focuses on the failure diagnosis. Two methodologies are discussed. The thesis ends with a general conclusion.

Samenvatting

De huidige en vorige windturbinegeneraties worden gekenmerkt door veelvuldig vroegtijdig falen van bepaalde componenten. Dit resulteert vaak in lange stilstanden van de windturbines. Indien het mogelijk zou zijn om dit falen voldoende op voorhand te voorspellen, dan zouden de componenten tijdens het normale onderhoud kunnen vervangen worden. Dit zou de onderhoudskosten en de stilstanden significant kunnen verlagen.

Het voorspellen en diagnosticeren van windturbinestoringen is momenteel een probleem dat nog niet afdoend opgelost is. Een nuttige methodologie moet in staat zijn om meerdere verschillende storingstypes te kunnen detecteren voordat ze effectief optreden. Dit betekent dat het niet alleen in staat moet zijn om het ogenblik te identificeren dat een component vreemd begint te gedragen, maar ook om patronen in het afwijkend gedrag te interpreteren.

De verbeterde toegankelijkheid van data heeft voor- en nadelen. Enerzijds maakt het mogelijk om het gedrag van windturbines in detail te bestuderen. Anderzijds, maakt de omvang van de data het analyzeren en interpreteren van patronen moeilijker voor experten. Dit noopt tot automatisatie. De ontwikkeling van een dergelijk systeem is momenteel de focus van onderzoek.

Het hoofddoel van het onderzoek dat in deze thesis voorgesteld wordt is de ontwikkeling van een automatisch storingsdetectie/predictie- en storingsdiagnosesysteem voor de aandrijflijn van windturbines. Hiervoor wordt data die standaard beschikbaar is van windturbines gebruikt, nl. 10-minuten Supervisory Control And Data Acquisition (SCADA) en statuslogboekdata. Dit hoofddoel kan opgedeeld worden in twee sub-doelen. Ten eerste moet het systeem in staat zijn om storingen in de windturbineaandrijflijn op voorhand te voorspellen via de analyse van de temperatuursignalen van verschillende componenten. Ten tweede moet het ook in staat zijn om het storingstype te bepalen op basis van de patronen in het (abnormaal) gedrag van de windturbine.

Het systeem bestaat uit 4 sequentiële stappen die gebruik maken van artificiële intelligentie (AI). De eerste stap maakt gebruik van machinaal leren (ML) om het normale temperatuursgedrag van de componenten te modeleren. De tweede stap gebruikt statistische data-analyse technieken om anomalieën te detecteren in het verschil tussen het geobserveerde temperatuursgedrag en het voorspelde, normale, gedrag. Deze resultaten worden in de derde stap gecombineerd met informatie uit het statuslogboek. *Pattern mining*- en *association rule mining*- technieken worden gebruikt om patronen of regels te identificeren die gerelateerd zijn aan verschillende aandrijflijnstoringen. In de vierde stap worden de patronen of regels verder verwerkt om zo een systeem te ontwikkelen voor de classificatie van de verschillende storingstypes. Dit kan dan gebruikt worden voor het bepalen van een diagnose.

Het ontwikkelde systeem wordt gevalideerd op data van drie echte operationale windturbineparken. Dit heeft als voordeel dat de performatie van het systeem getest kan worden alsof het effectief uitgerold is. De storingen zijn echt. Dit betekent dat er geen assumpties moeten gemaakt worden over hoe ze zich tonen in de data. Werken met echte data heeft echter ook uitdagingen omwille van imperfecties. De windturbineparken waarvan de data gebruikt wordt voor de validatie ervaarden meerdere verschillende types storingen tijdens de observatieperiode.

In dit onderzoek worden meerdere verschillende technieken voor elk sub-doel getest. De validatie toont aan dat de bestpresterende storingpredictietechniek de storingen accuraat en vroegtijdig kan ontdekken. Deze techniek wordt momenteel, op vraag van de windturbineoperator, periodiek gebruikt voor het monitoren van een operationeel windturbinepark. De bestpresterende storingsdiagnosemethodologie is in staat om patronen te detecteren die gerelateerd zijn aan bepaalde storingstypes. Deze patronen kunnen gebruikt worden om een accurate storingsdiagnose uit te voeren.

De thesis is als volgt gestructureerd. Ze begint met een introductie die de probleemstelling en de onderzoeksvragen formuleert. Het eerste hoofdstuk geeft een gedetailleerd overzicht van de kenmerken van de data en de storingseigenschappen. Het tweede hoofdstuk behandelt de storingpredictiemethodologie. De achtergrond wordt beschreven. Dit wordt gevolgd door een beschrijving van de methodologie, een discussie van de resultaten en een conclusie. Het derde hoofdstuk focust op de storingsdiagnose. Ook nu wordt eerst de achtergrond beschreven. Dit wordt gevolgd door een beschrijving van de methodologie, de resultaten en een conclusie. De thesis eindigt met een algemene conclusie.

Acknowledgments

First and foremost, I would like to thank my supervisors, Prof. Dr. Ann Nowé and Prof. Dr. Jan Helsen, for their extensive guidance and support during the 4 years of my PhD. I would also like to thank Dr. Timothy Verstraete for technical advice and support. Furthermore, this research would not have been possible without adequate financing. For this reason, I would like to thank the Flemish Government for supporting this research through the Flanders AI Plan. I would also like to thank the European Union for financing the research through H2020 PLATOON (Pr. No: 872592). Special thanks also go to FWO and VLAIO for their financial support through respectively the SBO Robustify project (S006119N), and the VLAIO ICON project Supersized 4.0. And finally, I would like to thank my parents for their support during the past 4 years.

Contents

List of Jury Members **3**

Summary **5**

Samenvatting **7**

Acknowledgments **9**

Contents **11**

List of Figures **15**

List of Tables **21**

List of Abbreviations **22**

1 Introduction **29**

1 General problem description 29

2 Overview of the wind turbine 30

2.1 Main structural parts 30

2.2 The wind turbine power curve and operating conditions 38

2.3 Pitch system 40

2.4 Yaw system 41

2.5 Generator 42

2.6 Rotor shaft, bearings and mountings . . . 45
2.7 Gearbox . . . 47
3 Failure modes . . . 49
4 Data sources and signals . . . 51
5 Estimating the performance of the methodologies . . . 53
6 High-level overview state of the art and research gaps . . . 55
7 Research objectives . . . 57
8 Scope definition . . . 58
9 Research strategy . . . 59
10 Main contributions . . . 60
11 Structure of the thesis . . . 61

2 Description of the data and failure cases 63
1 Overview of the characteristics of the data . . . 63
1.1 Description of the wind farms . . . 63
1.2 The structure of the data . . . 64
1.3 Overview of the selected signals and status codes . . . 65
1.4 The data quality . . . 67
1.5 Autocorrelation in the signals . . . 72
1.6 Correlations between the different signals . . . 75
1.7 Analysis of the frequency spectrum of the generator and gearbox signal . . . 78
2 Overview of the characteristics of the failure cases . . . 82
2.1 The failure modes of WF1 . . . 82
2.2 The failure modes of WF2 . . . 84
2.3 The failure modes of WF3 . . . 85
2.4 Forced outages . . . 86
3 Wind turbine ontologies as a way to standardize the wind turbine data . . . 86
3.1 Defining ontology . . . 87
3.2 Applications of ontologies . . . 88
3.3 Ontologies in wind energy research . . . 90
3.4 Making sense of temperature signals, status codes, and failure comments using ontologies . . . 91
4 Conclusion . . . 94

3 Failure prediction **95**
1 Background . . . 97
1.1 The current state of the art . . . 99
2 Data preprocessing steps used in this research . . . 104
2.1 Wind farm data normalization . . . 104
2.2 Transforming the data into a wind turbine-stacked format . . . 108
2.3 Identifying healthy and unhealthy data . . . 110
2.4 Splitting data in training, validation, and testing datasets . . . 112
2.5 Aggregating the data . . . 114
2.6 Handling measurement errors . . . 115
2.7 Handling missing values . . . 117
2.8 Scaling of the data . . . 118
3 Methodologies for modeling the normal behavior . . . 119
3.1 The wind farm median as benchmark . . . 119
3.2 Shallow machine learning pipeline . . . 120
3.3 Autoencoder pipeline . . . 126
4 Validation of the NBM models . . . 129
5 Comparison of the performance . . . 130
5.1 WF1 . . . 131
5.2 WF2 . . . 133
5.3 WF3 . . . 136
5.4 Conclusion based on comparison of results . . . 137
6 Concluding remarks concerning the normal behavior modeling (NBM) modeling . . . 137
6.1 Limited evidence for added value of using highly non-linear models as NBM when using data from which the wind farm median has been subtracted . . . 137
6.2 Limited added value of using signal lags or models that can model the time-dependencies . . . 139
6.3 Some evidence for inter-turbine differences in the prediction errors 139
6.4 Little evidence for drift over time . . . 140
6.5 Limited evidence that after a component replacement retraining is required . . . 141
6.6 No clear evidence that the autoencoder pipeline models transient behavior worse than steady-state behavior . . . 142
7 Identifying anomalies in the reconstruction error . . . 145

7.1 Detecting mean-shifts with cumulative sum (CUSUM) 146
7.2 Detecting point anomalies with MAD-IOD 152
7.3 Concluding remarks on the identification of anomalies in the reconstruction error 156

4 Automated failure diagnosis for wind turbines 159
1 Background 160
1.1 Frequent itemset mining 161
1.2 Rare itemset mining 165
1.3 High-utility itemset mining 166
1.4 Contrasting itemset mining 167
1.5 Rule-based classification 170
1.6 Conclusion 171
2 Identifying failure characterizing fingerprints 172
2.1 The methodology and results 173
2.2 Some final thoughts on using treatments or contrast-sets as basis for rule-based classifiers 186
3 Failure identification for wind turbines 188
3.1 Transforming the data into a format suitable for pattern mining . 189
3.2 Wind turbine ontologies: connecting the signals and the status codes to the appropriate system 190
3.3 Transforming the data into a transaction database 190
3.4 Association rule mining as a way to extract rules that can be associated with forced shutdowns 192
3.5 Going from frequent itemsets to failure diagnosis: making sense of the frequent pattern growth algorithm (FP-growth) generated patterns. 195
3.6 Some final thoughts on using frequent association rule mining, ontologies, and clustering for failure diagnosis. 201

5 Conclusion and future work 203

A Appendices 207

Curriculum Vitae 211

Bibliography 215

List of Figures

1.1 An example of a typical horizontal axis wind turbines (HAWT). Source: AVRG-VUB. 31
1.2 An example of a large vertical axis wind turbines (VAWT) (Darrieus wind turbine). Source: Guillom [2005]. 31
1.3 Schematic representation of the wind turbine. The representation is not accurately scaled. Some components are shown larger relative to others than what is the case in reality. 32
1.4 Schematic depiction of ring flanges. Source: Hashemi [2024]. 33
1.5 The nacelle of a Haliade X wind turbine. Source: Froese [2019] 35
1.6 Schematic representation of gravity foundation with piles. Source: CTE Wind France [2024] 36
1.7 Wind turbine on jacket foundation. Source: AVRG-VUB. 37
1.8 Gravity foundations under construction in France. Source: Lewis [2022] . 37
1.9 Wind turbine on a floating foundation. Source: Untrakdrover [2012]. . . . 38
1.10 Schematic overview of the power curve and operational zones. Source: Bilendo et al. [2023] 39
1.11 Schematic overview of pitch system. Source: Nielsen et al. [2014] 41
1.12 Schematic overview of the yaw system. Source: Chen et al. [2020] 42
1.13 Schematic overview of the different generator types for wind turbines and their associated converters. Partially based on Spiteri Staines et al. [2015] 43
1.14 An example of a squirrel-cage induction generator. Source: Heier [1996] . 45

1.15 An example of a doubly-fed induction generator. Source: Ruviaro et al. [2012] . . . 45
1.16 Several main/rotor shafts for wind turbines. Source: Fernandes [2023] . . 46
1.17 Rendering of a double tapered roller bearing for main/rotor shafts of wind turbines. Source: SKF [2024] . . . 47
1.18 A spur gear. Source: Industrial Quick Search [2024c] . . . 48
1.19 An example of helical gears. Source: Industrial Quick Search [2024a] . . . 48
1.20 A planetary gear. Source: Industrial Quick Search [2024b] . . . 48
1.21 A multistage gearbox. Source: Bajric et al. [2015] . . . 48
1.23 Schematic overview validation techniques. . . . 54
1.24 Overview of the different blocks or steps in the methodology. . . . 62

2.1 Schematic overview of the impact of removing missing values from the data given the nature of the missingness. MCAR stands for Missing Completely At Random, MAR stands for Missing At Random and MNAR stands for Missing Not At Random. . . . 68
2.2 Examples of measurement errors. . . . 70
2.3 Average autocorrelation (up to 1000 lags) for the different signals of WF1. The black vertical line indicates lag 144, which corresponds with 1 day (144 times 10 minutes = 1 day). . . . 73
2.4 Average autocorrelation (up to 1000 lags) for the different signals of WF2. The black vertical line indicates lag 144, which corresponds with 1 day (144 times 10 minutes = 1 day). . . . 74
2.5 Average autocorrelation (up to 1000 lags) for the different signals of WF3. The black vertical line indicates lag 144, which corresponds with 1 day (144 times 10 minutes = 1 day). . . . 74
2.6 Correlation heatmap showing the correlations between the base signals of WF1. . . . 76
2.7 Correlation heatmap showing the correlations between the base signals of WF2. . . . 77
2.8 Correlation heatmap showing the correlations between the base signals of WF3. . . . 78

2.9 Average frequency spectrum for WF1. The top plot shows the amplitudes for all calculated frequencies. The black vertical line indicates the amplitudes for the 1 / day frequency. The bottom plot shows a detailed view of the amplitudes for the frequencies lower than 1 / day. The black vertical line indicates an amplitude peak that corresponds more or less to the 1 / year frequency. 79
2.10 Average frequency spectrum for WF2. The top plot shows the amplitudes for all calculated frequencies. The black vertical line indicates the amplitudes for the 1 / day frequency. The bottom plot shows a detailed view of the amplitudes for the frequencies lower than 1 / day. The black vertical line indicates an amplitude peak that corresponds more or less to the 1 / year frequency. 80
2.11 Average frequency spectrum for WF3. The top plot shows the amplitudes for all calculated frequencies. The black vertical line indicates the amplitudes for the 1 / day frequency. The bottom plot shows a detailed view of the amplitudes for the frequencies lower than 1 / day. The black vertical line indicates an amplitude peak that corresponds more or less to the 1 / year frequency. 81
2.12 Schematic representation of replacement cleaning. Red bars are replacements that are not classified as failures. Green bars are replacements that are classified as failures. 85
2.13 Schematic representation of an ontology-based schema-based data integration scenario. The implementation layer shows several different implementations, i.e. two relational databases and one object-oriented software implementation. The conceptual model layer shows three different conceptual models, i.e. an enhanced entity-relationship (EER) (left), an object-relation mapping (ORM) (center), and a unified modeling language (UML) Class Diagram (right) [Keet, 2020]. 89
2.14 Schematic representation of an ontology-based data-level integration [Keet, 2020]. 90

3.1 Schematic overview of NBM framework. 98
3.2 Example of a decomposition of a generator rear bearing temperature signal of a wind turbine of WF1 in a common and idiosyncratic component . . . 106
3.3 Example of a decomposition of a generator phase temperature signal of a wind turbine of WF1 in a common and idiosyncratic component 107
3.4 Example of a decomposition of a nacelle temperature signal of a wind turbine of WF1 in a common and idiosyncratic component 108

3.5 Schematic overview of the shallow ML-pipeline. 121
3.6 Schematic overview of a perceptron. 124
3.7 Schematic overview of how the data is used for training and testing of the shallow ML methodology. t_0 is the first healthy observation for a wind turbine. 125
3.8 Schematic representation of undercomplete autoencoder 127
3.9 Schematic representation of the selection of healthy and unhealthy data . 129
3.10 mean absolute error (MAE) on healthy WF1 data for wind farm median (benchmark), shallow machine learning (ML), and autoencoder pipeline. . 132
3.11 MAE ratios on WF1 data for wind farm median (benchmark), shallow ML, and autoencoder pipeline. The blue horizontal line indicates a UHH-ratio of 1. 133
3.12 MAE on healthy WF2 data for wind farm median (benchmark), shallow ML and autoencoder pipeline. 134
3.13 unhealthy-healthy mean absolute error ratio (UHH-ratio)s on WF2 data for wind farm median (benchmark), shallow ML and autoencoder pipeline. The blue horizontal line indicates a UHH-ratio of 1. 135
3.14 MAE on healthy WF3 data for wind farm median (benchmark), shallow ML, and autoencoder pipeline. 136
3.15 Average reconstruction error on healthy data for different signals over different wind turbines. Each blue dot represents a wind turbine. 140
3.16 The figure shows for several days the active power of the wind turbine (blue curve), the steady-state zones (green), and the transient zones (red) for a wind turbine in WF1. 143
3.17 Autocorrelation for 700 lags for different signals of a wind turbine of WF1 before and after first-differencing the signals. Each line corresponds to a signal. The figure does not contain a legend since it would clutter the figure, and the exact signal is also not that important. 148
3.18 CUSUM values for two generator signals of 2 wind turbines that experienced a generator failure. 150
3.19 CUSUM values for 2 gearbox signals of 2 wind turbines that experienced a gearbox failure. 151

3.20 median absolute deviation - iterative outlier detection (MAD-IOD) scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for generator phase 2M1 for a wind turbine in WF1. The failure happened just after month 9. Month 0 is just a reference. The exact date to which it corresponds cannot be disclosed. 154
3.21 MAD-IOD scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for generator phase 3M1 for a wind turbine in WF1. The generator failure happened between months 4 and 5. Month 0 is just a reference. The exact date to which it corresponds cannot be disclosed. 155
3.22 MAD-IOD scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for a wind turbine in WF2. The failure happened just before the start of year 4. Year 0 is just a reference. The exact date to which it corresponds cannot be disclosed. 155
3.23 MAD-IOD scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for a wind turbine in WF2. The failure happened just before the start of year 7. Year 0 is just a reference. The exact date to which it corresponds cannot be disclosed. 156

4.1 Schematic overview of the pipeline of the first methodology consisting of Symbolic Aggregation Approximation (SAX), treatment learning/contrast-set mining, and rule-based classification. 173
4.2 Schematic representation of low and high failure risk zones. 175
4.3 Schematic representation of splitting of data in training and testing. . . . 176
4.4 Detailed view of several days of data from the front generator bearing temperature sensor of a wind turbine. The temperature signal (blue line) is normalized. The result after application of the piecewisse aggregate approximation (PAA) step is also shown (orange line). The PAA result is converted into a symbolic representation represented by a color scheme at the bottom of the plot. Green corresponds to "c", yellow to "d" and red to "e". 177
4.5 WCSM: best rule summary for NDE failures. 178
4.6 WCSM: best rule summary for DE failures. 179
4.7 NWCSM: top-20 best rule summary for NDE failures. 180
4.8 NWCSM: top-20 best rule summary for DE failures. 181

4.9 WCSM: DE failure prediction. 184
4.10 WCSM: DE failure prediction. 184
4.11 NWCSM: DE failure prediction. 185
4.12 NWCSM: DE failure prediction. 185
4.13 Schematic overview of the failure identification methodology using association rule mining and forced shutdowns. 189
4.14 Histograms showing the distribution of the absolute support, confidence, and lift of the rules identified by the FP-Growth association rule miner. . 194
4.15 Schematic overview of the methodology that transforms the FP-growth association rules into failure diagnosis 196
4.16 Example correct identification of converter failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time. 199
4.17 Example correct identification of a line-side inverter failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time. 200
4.18 Example correct identification of a generator failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time. 200
4.19 Example correct identification of a generator high shaft current failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time. 201

List of Tables

2.1 Example of the structure of the supervisory control and data acquisition (SCADA) data. The example is completely artificial. 64
2.2 Example of the structure of the status logs. The example is completely artificial. 65
2.3 Example of the structure of a typical failure event file. The example is completely artificial. 65
2.4 Temperature signals from the SCADA data from WF1. 66
2.5 Temperature signals from the SCADA data from WF2. 66
2.6 Temperature signals from the SCADA data from WF3. 67
2.7 Table with an overview of the failure modes/types for each wind farm. . . 83
2.8 Table containing the temperature signal ontology. The temperature signals for WF1, WF2, and WF3 are represented by their signal IDs, which can be found in Tables 2.4, 2.5, and 2.6. 92
2.9 Status code ontology for WF1. 93

3.1 Example of (fictive) data in a turbine-stacked format. The wind farm contains M wind turbines (T1, T2, ..., TM). Each wind turbine has a certain number of observations (one per 10 minutes). Each wind turbine has N signals. 109
3.2 Amount of training-validation and test data for WF1-3 114
3.3 Table with an overview of how the wind farm median and the wind farm prediction error is calculated for a fictive wind farm with three wind turbines. 120

3.4 Mean absolute reconstruction error for transient and steady-state behavior for WF1 144
3.5 Mean absolute reconstruction error for transient and steady-state behavior for WF2 144
3.6 Mean absolute reconstruction error for transient and steady-state behavior for WF3 145

4.1 Wind turbine state definitions generated by the wind turbine state annotator. 174
4.2 Confidence and support thresholds used for postprocessing of the patterns. 180
4.3 Accuracy, true positive rate (TPR) and true negative rate (TNR) results for rule-based classifiers based on weighted or non-weighted contrast set mining algorithms for drive-end (DE) and non-drive-end (NDE) replacements. 183
4.4 Table shows for each cluster the component score. The higher the score the more linked the cluster is to that specific component issue. The highest score for each cluster is put in bold. 197
4.5 Table shows 10 transactions randomly selected from the transaction database combined with the issue that was assigned to it by the methodology. 199

A.1 Upper and lower measurement error thresholds for WF1. The * indicates that the value has been hidden for confidentiality reasons. 208
A.2 Upper and lower measurement error thresholds for WF2. The * indicates that the value has been hidden for confidentiality reasons. 209
A.3 Upper and lower measurement error thresholds for WF3. The * indicates that the value has been hidden for confidentiality reasons. 210

List of Abbreviations

AC alternating current. 41, 42, 44

ACF autocorrelation function. 72

AI artificial intelligence. 55

ARIMA autoregressive integrated moving average. 147

ARL average run length. 147

CARs class association rules. 170

CART Controls Advanced Research Turbine. 103

CBA classification based on associations. 171

CMAR classification based on multiple association rules. 170, 171

CUSUM cumulative sum. 14, 97, 145–147, 149, 152, 154–156

CV cross-validation. 112

DC direct current. 42, 43

DE drive-end. 22, 85, 135, 176, 178, 179, 183, 185–187

DFIG doubly-fed induction generator. 44

DFT discrete fourier transform. 79

DUOS discovery of utility-aware outlier sequential rules. 167

ECLAT equivalence class clustering and bottom-up lattice traversal. 162, 163

EER enhanced entity-relationship. 17, 89

EESG electrically excited synchronous generator. 42–44

EFB exclusive feature bundling. 123

ELU exponential linear unit. 127

FFT fast fourier transform. 79

FN false negative. 183, 184

FNN feedforward neural network. 99, 124

FOIL first order inductive learner. 171

FP false positive. 183, 184

FP-growth frequent pattern growth algorithm. 14, 20, 163–165, 170, 192, 193, 195, 196

FPI frequent pattern isolation. 164

FPOF frequent pattern outlier factor. 165

FRC fully rated converter. 44, 45

FSIG fixed speed induction generator. 44

GBDT gradient boosting decision tree. 123

GBM gradient boosting machine. 122, 123, 130, 138

GOSS gradient-based one-side sampling. 123

GSC grid-side converter. 44

GW gigawatt. 29

HAWT horizontal axis wind turbines. 15, 30, 31

HSS high-speed stage. 49

Hz hertz. 51, 103

IEC International Electrotechnical Commission. 39, 86

IGBT insulated-gate bipolar transistor. 50

IOD Iterative Outlier Detection. 152, 153

IOR Iterative Outlier Removal. 152, 153

ISS intermediate-speed stage. 49

kW kilowatt. 47

LASSO least absolute shrinkage and selector operator. 121

LSS low-speed stage. 49

LSTM long short-term memory. 99, 139

m meter. 33–35, 47

MAD median absolute deviation. 125, 152, 153

MAD-IOD median absolute deviation - iterative outlier detection. 19, 97, 145, 152, 154–156

MAE mean absolute error. 18, 53, 129, 131–137, 142

MAR missing at random. 69, 117, 118

MCAR missing completely at random. 68, 100, 117, 118

MFPOF maximal frequent pattern outlier factor. 164

MICE multivariate imputation by chained equations. 101

ML machine learning. 18, 55, 60, 96, 104, 110, 111, 114, 115, 117, 119, 120, 129, 130, 132–138, 141

MLP multilayer perceptron. 130, 138

MNAR missing not at random. 69

mRI minimal rare itemsets. 165

MSE mean squared error. 126

MW megawatt. 47, 49, 63–65, 73, 82, 85, 86

m² square meter. 34

NBM normal behavior modeling. 13, 17, 55, 56, 86, 96–102, 110, 112–114, 119, 120, 125, 129–131, 136, 137, 139, 141, 143, 145, 186

NDA non-disclosure agreement. 51, 56

NDE non-drive-end. 22, 85, 135, 176, 178, 179, 183–187

NREL National Renewable Energy Laboratory. 103

NWCSM non-weighted contrast set mining. 168, 178, 179, 183, 187, 202

OLS ordinary least squares. 99, 147

OO object-oriented. 88, 89

ORM object-relation mapping. 17, 89

PAA piecewisse aggregate approximation. 19, 177

PCA principal component analysis. 102, 126

PMSG permanent magnet synchronous generator. 42, 44

PRC partially rated converter. 44, 45

PRM predictive rule miner. 171

PWM pulse width modulation. 44

QTXDB quantitative transaction database. 166

RAM reliability, availability and maintainability. 50

RDS-PP Reference Designation System for Power Plants. 91

ReLU rectified linear unit. 124, 127

RF random forest. 99, 138

RIPPER repeated incremental pruning to produce error reduction. 171

RSC rotor-side converter. 44

RSPM rare sequential pattern mining. 165

RUL remaining useful life. 99

SAX symbolic aggregation approximation. 164, 172, 175–177

SCADA supervisory control and data acquisition. 21, 51, 53, 58, 59, 64–67, 72, 95, 98, 99, 102–104, 161, 166, 173, 175

SCIG squirrel cage induction generator. 44

SD standard deviation. 153

SMOTE synthetic minority oversampling technique. 102

SPC statistical process control. 146, 147

STUCCO search and testing for understandable consistent contrasts. 168, 169, 178

SVM support vector machine. 99, 145

TN true negative. 183, 184

TNR true negative rate. 22, 183

TP true positive. 183, 184

TPR true positive rate. 22, 183

TSS-MAE transient vs. steady-state mean absolute error. 142, 143

TW terawatt. 29

TXDB transaction database. 162, 166, 167, 197

UHH-ratio unhealthy-healthy mean absolute error ratio. 18, 53, 129, 131, 132, 134–137

UML unified modeling language. 17, 89

VAWT vertical axis wind turbines. 15, 30, 31

WCSM weighted contrast set mining. 170, 178, 179, 183, 187, 202

°C degrees celcius. 69, 71, 116, 117

1 Introduction

1 General problem description

In recent years large investments have been made in the wind energy industry, as part of the push to transition the economy from fossil-based energy sources to renewables. The number of installed wind turbines has increased fast. 77.6 gigawatt (GW) of new wind energy production capacity was installed in 2022, which brings the total installed capacity to 906 GW. It is expected that by the end of 2024, this number will have grown to over 1 terawatt (TW) [Hutchinson and Zhao, 2023]. This evolution was and is driven to an important extent by ambitious climate goals and a changing geopolitical situation. So far important steps have been taken, however, to complete the transition many more investments are required.

To guarantee further growth of the investments and the industry as a whole, the profitability of the former needs to be guaranteed. Several factors influence this, but an important role is played by the operational and maintenance costs of the wind turbines. Studies estimate that they account for 25-40% of the levelized cost of energy [Pfaffel et al., 2017]. An important part of these costs is caused by wind turbine failures.

Unexpected failures are the most problematic. They result often, depending on the component that has failed, in long downtimes during which the wind turbine does not generate an income for the owner. For example, unexpected generator failures (e.g. short-circuits or bearing failures), can result in months of downtime for offshore wind turbines. This is in part the result of the difficult accessibility of the offshore wind turbines [Carroll et al., 2016]. The size of the wind turbines is also a factor. Large components (e.g.

gearboxes, generators, ...) require specialized expensive equipment (i.e. jack-up type vessels). These cost hundreds of thousands of euros per day, which is much more than the thousands of euros a day that ordinary access vessels, which are used for standard maintenance, cost [Dinwoodie et al., 2013].

Even though failures of large wind turbine components are rare, their high cost makes them highly influential on the overall costs. A possible solution to this problem would be to repair these components during regular maintenance before they actually fail. This is only possible if the failing components can be identified sufficiently in advance. Sufficiently in the case of major failures is at least one month and preferably several months, because large equipment, e.g. a ship with a crane, ..., must be found and brought to the location, and the broken component needs to be ordered and brought to the site. If the wind turbine operator is notified in time, the maintenance can be added to the normal maintenance schedule, or can be organized more efficiently.

2 Overview of the wind turbine

There are several types of wind turbines. The main difference is how they rotate, i.e. horizontally for HAWT or vertically for VAWT. Figure 1.1 shows a typical HAWT. Figure 1.2 shows a Darrieus wind turbine which is a type of VAWT. The most common wind turbine type today is the HAWT. The research presented in this thesis focuses on HAWT wind turbines like the one in Figure 1.1.

2.1 Main structural parts

Figure 1.3 shows a schematic representation of a typical wind turbine. As can be seen in the figure, the wind turbine consists of many different components. Not all components, except for the most important ones, e.g. gearbox, generator, converter, blades, transformer, ..., are shown to avoid cluttering the figure. In what follows, several of the main structural parts, e.g. the tower, nacelle, and the foundation are discussed. Components like for example the pitch and yaw systems, the gearbox, the generator, and the converter are discussed later.

Figure 1.1: An example of a typical HAWT. Source: AVRG-VUB.

Figure 1.2: An example of a large VAWT (Darrieus wind turbine). Source: Guillom [2005].

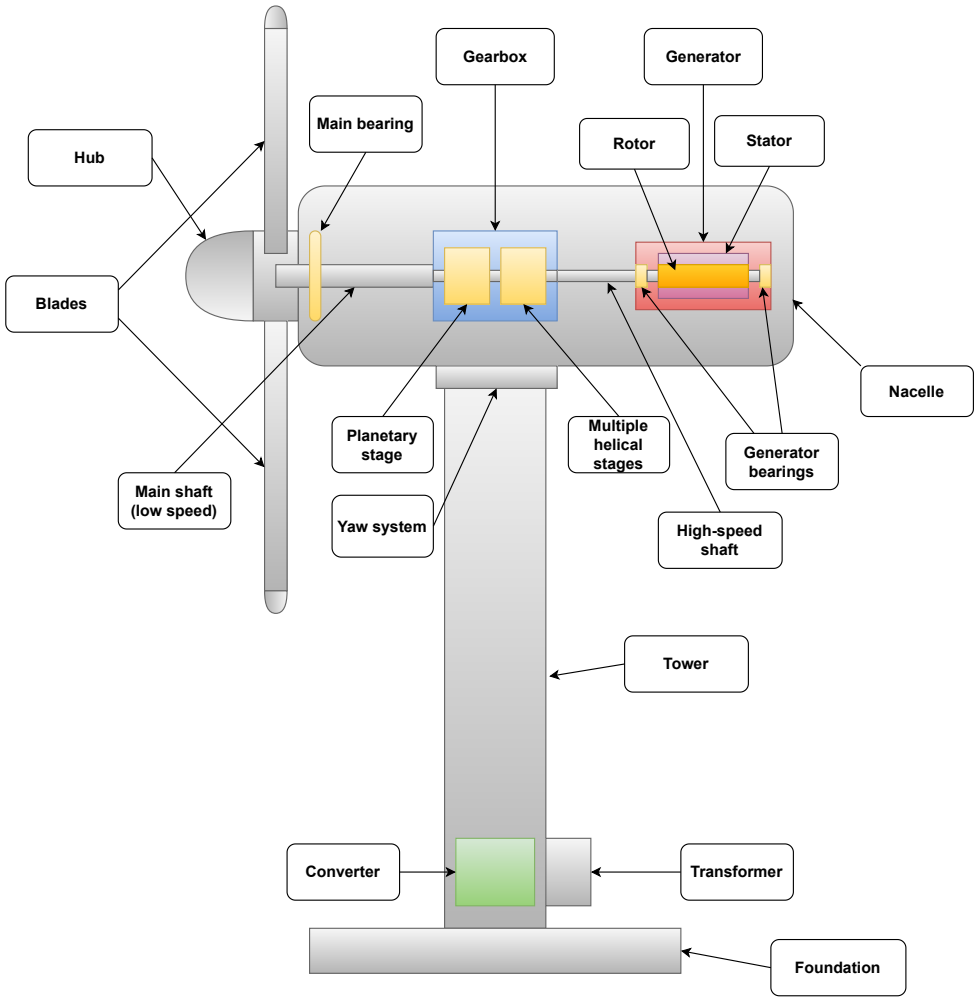


Figure 1.3: Schematic representation of the wind turbine. The representation is not accurately scaled. Some components are shown larger relative to others than what is the case in reality.

Tower The tower supports the nacelle. Over the years the size of the wind turbines has increased. Currently, the largest ones that are put in production have a tower with a height (also called hub height) well over 100 meter (m). For example, the Vestas V236-15MW will have a hub height of around 145 m (although the actual height will be site-specific), and the Vestas V172-7.2 MW wind turbine will have a hub height of up to 199 m according to the Vestas website. There are two reasons for using such a large tower. Firstly, it is necessary to accommodate the ever-increasing size of the wind turbine blades (see further). Secondly, in general, at higher altitudes, the average wind speed is higher. This is also the case for the capacity factor of the wind turbines [Lantz et al., 2019].

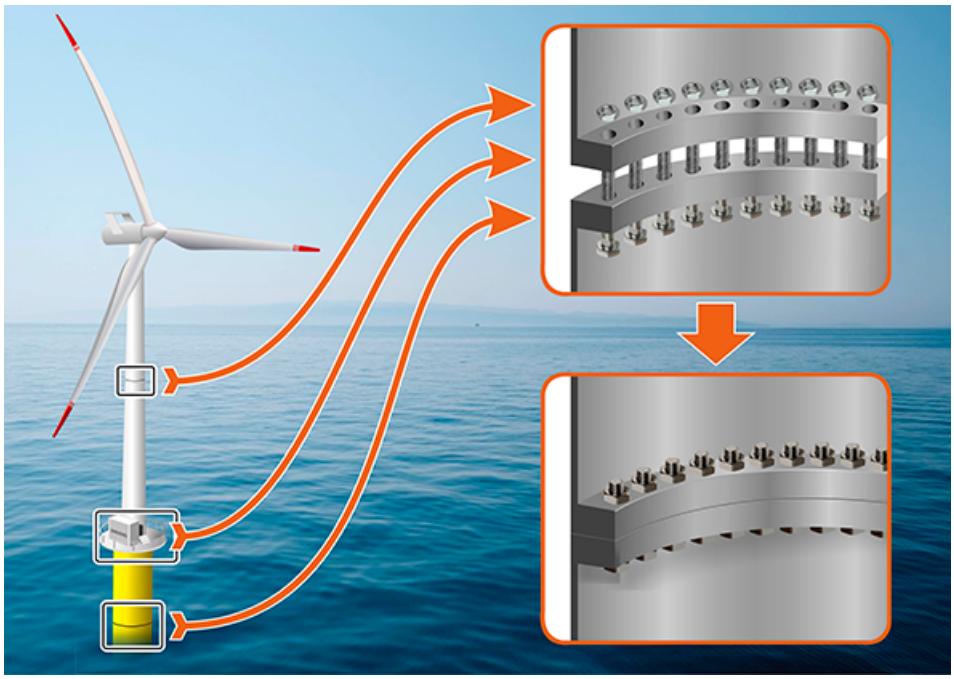


Figure 1.4: Schematic depiction of ring flanges. Source: Hashemi [2024].

The tower generally is a steel tube that is built up from several individual sections. These are connected using ring flanges with prestressed bolts (see Figure 1.4). The tower is connected to the foundation of the wind turbine. It is expected to last at least 20 years. During that time it is exposed to large loads. During the design phase of the wind turbine, fatigue and extreme loads are taken into account to make sure that the wind turbine will

survive its planned usage time. The tower makes up around one-third of the overall cost of the wind turbine [Faber, 2014].

Blades The wind turbine blades are the most prominent part of the wind turbine rotor. Just like the tower, they have increased in size over time. For example, according to the Vestas website, the rotor diameter of the Vestas V236-15MW wind turbine is 236 m, with blades of 115 m in length. This results in a swept area (i.e., the area of the virtual circle with a radius equal to the length of a blade) of 43,742 square meter (m^2). Most modern wind turbines have three blades. However, wind turbines have been constructed with only two or even one blade.

The design of wind turbine blades is important since it determines the capacity yield. For small rotors (short blades) the wind loads dominate the strength design. For large rotors, the bending moments and the weight of the blades, which increase exponentially with the blade length, play a more important role. The blades for modern wind turbines are constructed from fiber-reinforced plastics because they are strong and the material is relatively cheap [Dannenberg, 2014b].

The wind turbine blades are exposed to large loads during their lifetime. These can be the result of many factors, e.g. wind pressure at constant wind velocity, extreme wind due to extreme gusts that appear once every 50 or 100 years, changes in the direct flow velocity and direction due to the distribution of the wind velocity of the height, short-term changes in wind velocity, turbulence of the wind, passing of the rotor blade in front of the tower, ... [Dannenberg, 2014b]. These need to be taken into account during the design phase.

Nacelle The nacelle contains the components that are necessary to create electrical energy. The rotor converts the kinetic energy of the wind into a rotational movement [Siegfriedsen, 2014]. This movement is converted in the nacelle into electrical energy. The nacelle sits on top of the tower. For the large wind turbines, this nacelle is large and heavy (see Figure 1.5). It contains important components (e.g. the gearbox, generator, ...). The mass of the nacelles of the current generation of wind farms is several hundred tons, which makes it challenging to place it on top of the tower.

The nacelle is subjected to strong forces. The rotor transfers torque, pitch, and yaw moments to it. These loads must be taken into account during the modeling of the nacelle. This is a particularly important step since damage to it has a large cost [Siegfriedsen, 2014].

Figure 1.5: The nacelle of a Haliade X wind turbine. Source: Froese [2019]

Foundation To make sure that the wind turbine does not topple it is built on top of a foundation. A distinction needs to be made between onshore and offshore wind turbines. For onshore wind turbines, several different types of foundations are used.

The first type is the gravity foundation. The moment of the wind turbine is absorbed by a heavy, large-area foundation. Different shapes of foundations are possible, e.g. hexagonal, octagonal, ... The gravity foundation can only be used if the ground conditions allow it. If the soil is too soft, the foundation will start to sink. A second type adds piles to the gravity foundation (for an example see Figure 1.6). This prevents the sinking of the foundation. The piles are typically organized in a star shape [Faber, 2014].

A third type of foundation is a guyed system. Cables that are anchored in the ground are attached to the tower. This relieves the tower which makes it possible to use a tower with a smaller diameter and thinner walls. The cables are anchored in the ground using piles. This foundation type requires a thorough analysis of the soil conditions between the foundation and the piles [Faber, 2014].

Also for offshore wind turbines, different types of foundations are used. Depending on the size of the wind turbine, the depth of the water, the properties of the sea bed, and environmental conditions (e.g. currents, tides, waves, ice, and loads, ...) some types are more suitable than others [Dannenberg, 2014a].

For water with a depth of at most 60 m, the following types can be used: monopiles (max. depth 30-35 m) (see Figure 1.3), tripods (max. depth 40 m), jackets/framed (max. depth 60 m) (see Figure 1.7), suction buckets (max. depth 25 m), weighted or gravity foundations (max. depth 20-30 m) (see Figure 1.8), and variations/combinations of the previous types.

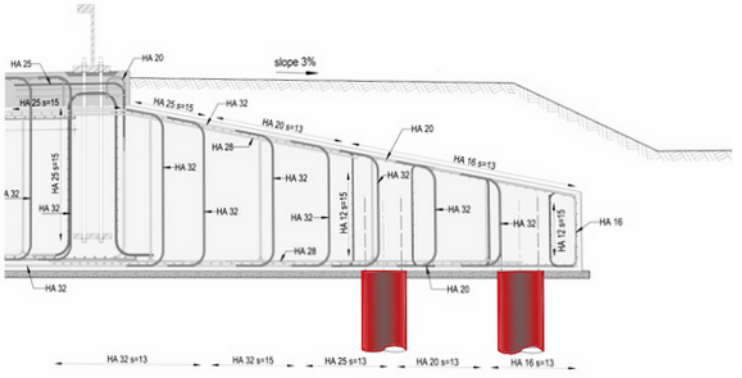


Figure 1.6: Schematic representation of gravity foundation with piles. Source: CTE Wind France [2024]

For water with a greater depth floating foundations are used. Several floating types are possible. Firstly, *tension legs*, make use of large floating bodies that are kept underwater by cables that are anchored in the seabed. Secondly, *spar buoys*, make use of vertical floating towers that are anchored in the seabed using cables. Figure 1.9 shows a wind turbine on a floating foundation. Thirdly, variations of the above [Dannenberg, 2014a].

Due to the large cost of offshore wind turbine foundations, they are only used in combination with large wind turbines. Floating foundations are currently the topic of much research.

Figure 1.7: Wind turbine on jacket foundation. Source: AVRG-VUB.

Figure 1.8: Gravity foundations under construction in France. Source: Lewis [2022]

Figure 1.9: Wind turbine on a floating foundation. Source: Untrakdrover [2012].

2.2 The wind turbine power curve and operating conditions

Power curve An important metric that indicates the performance of the wind turbine is the *power curve*. This curve shows the non-linear relation between the wind speed at hub height and the active power produced by the wind turbine. This relation is unique for each wind turbine. Figure 1.10 gives a schematic overview of a power curve. Eq. 1.1 expresses the relation between the two mathematically. The equation shows that the active power production is not just influenced by the wind speed at hub height, but also by the air density, the swept area, and the power coefficient (which denotes the percentage of power in the wind captured by the wind turbine) [Bilendo et al., 2023], [Sohoni et al., 2016].

$$P = \frac{1}{2}\rho A C_p(\lambda, \beta) v^3 \tag{1.1}$$

where:

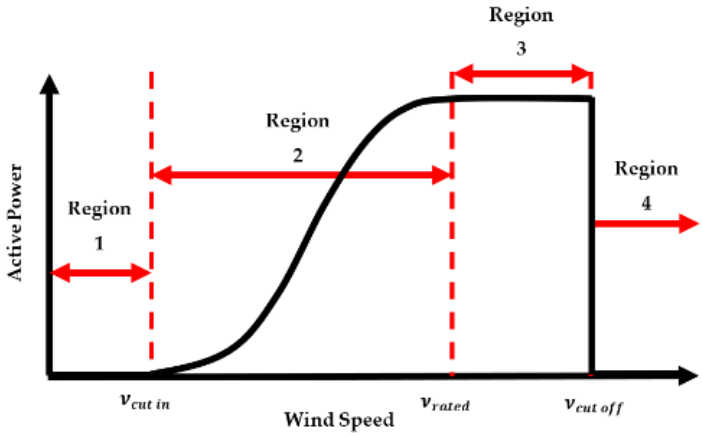


Figure 1.10: Schematic overview of the power curve and operational zones. Source: Bilendo et al. [2023]

ρ = Air density in $\frac{kg}{m^3}$.
A = Swept area or cross-sectional area of the wind in m^2.
$C_p(\lambda, \beta)$ = Power coefficient (i.e. measure of the wind turbine efficiency).
λ = Tip-speed ratio (i.e. the ratio of the tip speed and the wind speed).
β = Blade pitch angle (i.e. the angle of the blade rotation plane and the tip line).
v = Hub-height wind speed in $\frac{m}{s}$.

The power coefficient depends on the tip-speed ratio and the blade-pitch angle. The tip-speed ratio is the ratio between the tangential speed of the tip of the blade and the wind speed. Eq. 1.2 expresses this ratio mathematically [Bilendo et al., 2023].

$$\lambda = \frac{\omega R}{v_1} \tag{1.2}$$

where:

ω = Rotational speed of the rotor in $\frac{rad}{s}$.
v_1 = Wind velocity in $\frac{m}{s}$.

The power curve of a wind turbine is certified by the builder. This is done according to the procedure developed by the International Electrotechnical Commission (IEC). At the test site, measurements of the wind speed and the power output are done during a

long observation period. The *method of bins* is used to determine the power curve. This entails that the wind speeds are normalized and binned. This is also done for the power [Lydia et al., 2014].

Operating conditions A wind turbine has several operational modes (also called *operating conditions*). These correspond to certain regions on the power curve that are determined by the wind speed and wind turbine control. The different regions are defined as follows [Bilendo et al., 2023]:

- Region 1 = No power: wind speed lower than cut-in wind speed (i.e. the wind speed at which the blades start rotating). The wind speed is too low to let the wind turbine generate power.
- Region 2 = Torque control: wind speed between cut-in and rated wind speed. The power production evolves from 0 to its rated power.
- Region 3 = Pitch control/torque control: wind speed between rated and cut-off wind speed (i.e. the wind speed above which the wind turbine is stopped). The wind turbine produces at its rated power.
- Region 4 = Extended mode: wind speed higher than cut-off wind speed. The wind turbine is turned off to avoid damage.

2.3 Pitch system

The wind turbine output can be regulated in two ways: pitch regulation/control and stall regulation. The latter was used mainly on old wind turbines. These systems did not allow for changing the pitch of the blades because they were fixed to the rotor. Control of the wind turbine was then done using the stall effect. With the increasing size of the wind turbines, it became economically viable to install a more complex pitch system. This system is used to change the angle of the blades. By combining the pitch system with control algorithms, the loads (and the power output) of the wind turbine can be kept under control [Siegfriedsen, 2014].

During normal grid operation, the pitch angle is varied between 0°and 30°. To let the rotor turn slowly, which might be useful e.g. for lubrication, a pitch of 70°can be used. To stop the rotor rapidly the blades can be set at an angle of 90°(aerodynamic braking). This is also called the feathering setting [Siegfriedsen, 2014].

Several types of systems have been developed. In the past, a system was used that worked with a single actuator that put all blades at the same pitch angle. A disadvantage of this is that if the pitch system fails, the rotor can only be stopped using a mechanical brake. If the pitch angle at the time of failure is not set at the feathering position, it

causes strong forces in the drivetrain [Siegfriedsen, 2014]. To solve this, a system was designed in which each blade has its own pitch setting and actuator. This is commonly used in modern wind turbines. Research is ongoing to use the pitch setting of each blade to remove load unbalances within a single rotation of the rotor [Siegfriedsen, 2014].

The pitch system requires a rotatable supported connection to the rotor hub. The blade support uses a dual-row four-point bearing [Siegfriedsen, 2014]. Figure 1.11 gives a schematic overview of the pitch system seen from the side. On the left side of the figure, the electric motor is shown, and on the right side the rotor blade.

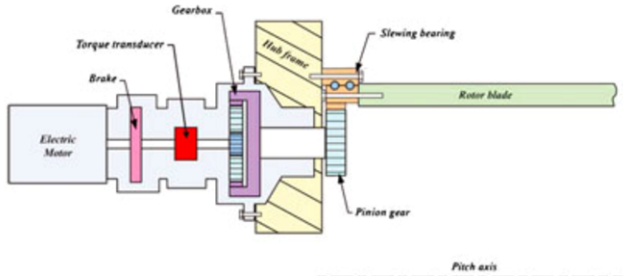


Figure 1.11: Schematic overview of pitch system. Source: Nielsen et al. [2014]

2.4 Yaw system

To track the wind direction, wind turbines have a yaw system. The purpose of this system is to keep the nacelle directed into the wind. The control system does not react to every small change in wind direction. This would be too demanding for the system and the wind turbine as a whole. Only if after a certain amount of time an average fixed-angled direct flow has been established the controls activate [Siegfriedsen, 2014].

The yaw system consists of different components. Figure 1.12 gives a schematic overview. Firstly, there is the tower head bearing. This is a large ball bearing with external or internal gears. It is designed as a single or double-row four-point bearing (see slewing or turntable bearing in Figure 1.12). Secondly, there are one or more electrical or hydraulic drives. These consist of a drive motor with a drive-side pinion, several coaxial epicyclic gear steps, and a flanged alternating current (AC) motor. The AC motor has usually an integrated stop brake [Siegfriedsen, 2014].

Thirdly, the brakes. In general, these consist of a brake disc and several hydraulic caliper disc brakes. Two pressure levels are used: a lower pressure for damping and a higher one for

braking. Some systems work with an electrical brake system. Other types do not use disc brakes but a glide bearing with the brakes integrated into the drive [Siegfriedsen, 2014].

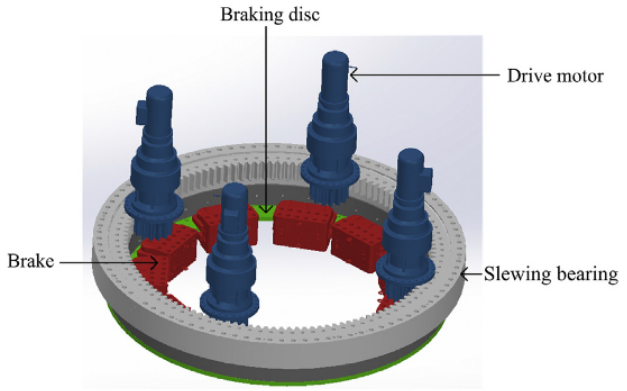


Figure 1.12: Schematic overview of the yaw system. Source: Chen et al. [2020]

2.5 Generator

The generator plays an important role in the generation of electric power from the rotational movement of the rotor. Multiple generator types are used in wind turbines, each with advantages and disadvantages. Figure 1.13 gives a schematic overview of the different generator types and the converters they are associated with.

There are three main types of wind turbine generators: direct current (DC), AC synchronous, and AC asynchronous generators. Each type can be run in principle at fixed or variable speed. Variable speed is most useful for wind turbines due to the variable nature of the wind power. It reduces the physical stress on the wind turbine blades and the drivetrain, and it improves the aerodynamic efficiency of the system and torque transient behaviors [Cao et al., 2012].

DC generators These generators are not often used for wind turbines except in low-power demand situations [Cao et al., 2012]. Because of this, they will not be discussed in more detail.

AC synchronous generators AC synchronous wind turbines can be divided into two sub-types, i.e. the permanent magnet synchronous generator (PMSG) and the electrically

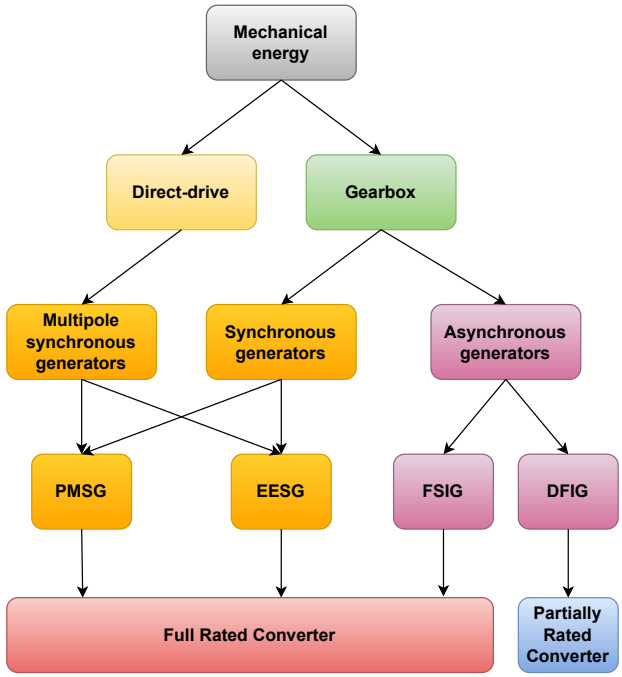


Figure 1.13: Schematic overview of the different generator types for wind turbines and their associated converters. Partially based on Spiteri Staines et al. [2015]

excited synchronous generator (EESG). PMSG takes constant or DC excitations from permanent magnets. For the EESG this is done through electrical excitation. Fixed-speed synchronous generators require that the rotor speed is kept at the synchronous speed [Cao et al., 2012].

The advantage of synchronous generators is that they are proven technology. Disadvantages are that when fixed-speed synchronous generators are used, random wind speed fluctuations and periodic disturbances caused by tower-shading effects and natural resonances of components, are passed to the power grid. Synchronous generators also tend to have a low damping effect. This means that drivetrain transients can not be absorbed electrically. Because of this an additional damping element or a gearbox assembly mounted on springs and dampers is required. Furthermore, synchronizing the frequency

of the generator to that of the grid is a delicate procedure. The generators are in general also more complex, costly, and prone to failure [Cao et al., 2012].

PMSG generators have several advantages, i.e. elimination of the commutator, slip rings, and brushes make the machines rugged, reliable, and simple. The usage of permanent magnets eliminates the field winding and the power losses that are associated with it. The disadvantages of this generator type are that field control is impossible, and the permanent magnets can be very expensive for large machines. PMSG generators can not generate electrical power with a fixed frequency. This is caused by the variability of the wind speeds. To solve this issue AC-DC-AC conversion by power converters is used [Cao et al., 2012].

Both EESG and PMSG generators require a Fully fully rated converter (FRC). A FRC implies that there is a complete decoupling between the generator and the grid, which allows for variable-speed operation and full control of active and reactive power [Chen et al., 2009]. This type of converter is more expensive than a partially rated converter (PRC).

AC asynchronous generators In modern wind turbines, induction generators are extensively used. Two types can be distinguished. The first type is the fixed speed induction generator (FSIG) with squirrel cage rotors (sometimes called squirrel cage induction generator (SCIG) (see Figure 1.14)). The second type is the doubly-fed induction generator (DFIG) with wound rotors (see Figure 1.15) [Cao et al., 2012].

In FSIG generators the stator is connected to the grid via a transformer and the rotor is connected to the wind turbine by a gearbox. The advantages of this type are that they are simple, reliable, inexpensive, and well-developed. Furthermore, they have a high degree of damping and can absorb rotor speed fluctuations and drivetrain transients (i.e. fault tolerant) [Cao et al., 2012].

They have however also several disadvantages. Firstly, they draw reactive power from the grid, which means that some form of reactive power compensation is needed (i.e. capacitors or power converters). Secondly, fixed speed induction generators are limited to operate only within a very narrow range of discrete speeds (SCIG generators can be used for variable speed wind turbines, but the output voltage can not be controlled and reactive power needs to be supplied externally), and there are issues with the generator size, noise, low efficiency, and reliability [Cao et al., 2012].

FSIG generators require an FRC. DFIG generators can work with a FRC or a PRC. With PRCs, there is not a complete disconnect between the generator and the grid. The stator is directly connected to the grid through transformers and the rotor is connected to the grid through pulse width modulation (PWM) power converters. The converters control the rotor circuit current, frequency, and phase angle shifts. The PRC of DFIG generators consists of two converters: a rotor-side converter (RSC) and a grid-side converter (GSC). This generator type has several advantages. Firstly, it can operate at a wide slip range. This

results in a higher energy yield, a reduction of mechanical stresses and power fluctuations, and reactive power is controllable. Secondly, the PRC is cheaper than a FRC [Cao et al., 2012].

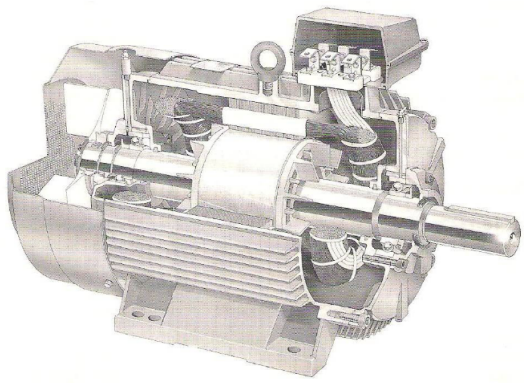


Figure 1.14: An example of a squirrel-cage induction generator. Source: Heier [1996]

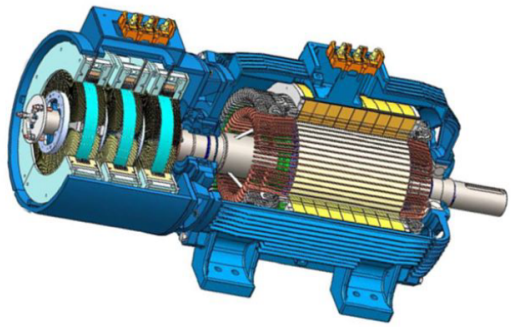


Figure 1.15: An example of a doubly-fed induction generator. Source: Ruviaro et al. [2012]

2.6 Rotor shaft, bearings and mountings

The rotor shaft (see Figure 1.16) and bearings connect the rotor to the rest of the drivetrain and the structure of the wind turbine. The bearings, also called main bearings, guide the

Figure 1.16: Several main/rotor shafts for wind turbines. Source: Fernandes [2023]

rotor loads into the structure of the nacelle. These bearings are subjected to high loads. To prolong their life and ensure a more or less friction-free operation a central lubrication system is used. This system is not only used for the rotor bearings but also for the blade bearings, the yaw or azimuth bearing, and the generator bearings [Siegfriedsen, 2014].

The rotor shaft is held in place by one or more bearings. There are different mounting systems for the rotor shaft, i.e. double mounting, three-point support, and torque support. The double mounting system can be subdivided into double bearings with separate housings and double bearings with common housing. The latter is more compact than the former. The three-point support system has a rear rotor shaft bearing that is integrated into the gearbox. The front rotor shaft bearing is directly connected to the machine carrier. The advantage of this system is that it is compact but still accessible. The disadvantages are that it puts extra loads on the gears of the gearbox, and if some bearings are damaged they can not be easily replaced [Siegfriedsen, 2014].

The torque support is the most compact system. A double-row tapered roller (see Figure 1.17) bearing is used as the support of the rotor shaft. This system is demanding on the machine carrier since the loads it has to transfer to the tower are high [Siegfriedsen, 2014].

Figure 1.17: Rendering of a double tapered roller bearing for main/rotor shafts of wind turbines. Source: SKF [2024]

2.7 Gearbox

The gearbox is another important part of the wind turbine drivetrain. Just like several other components it has increased in size over time. Currently, gearboxes with a diameter of up to 3 m and power of up to 15 megawatt (MW) are built [Nejad et al., 2022]. The gearbox is an expensive component, and the failure may result in high downtimes and is expensive to repair. For this reason, a lot of research has been done on improving their reliability. The gearboxes in wind turbines are usually multistage or multi-stepped (except for the smallest wind turbines). It converts the large torque that is created by the rotor into a high rotation speed that can be used to drive the generator.

Small wind turbines (around 100 kilowatt (kW)) use only spur-geared drives. A spur gear or straight-cut gear is a simple gear with straight tooth faces parallel to the axis of rotation. They have only one point where the torque is transferred [Siegfriedsen, 2014]. This is however insufficient for larger wind turbines. Figure 1.18 shows a typical spur gear.

Helical gears are gears for which the teeth lie at an angle to the axis of the shaft. Due to this at no time are the teeth of a gear fully used, and before the contact seazes between a pair of teeth the contact between the next pair of teeth is commenced. This allows for a smoother transition under heavy load. Furthermore, due to the angle of the teeth, they are longer, which makes them stronger. This allows them to transfer larger torque loads and makes the operation more smooth [Simmons et al., 2012]. Figure 1.19 shows a typical helical gear.

Figure 1.18: A spur gear. Source: Industrial Quick Search [2024c]

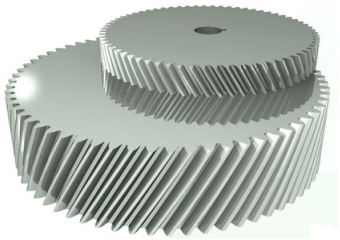


Figure 1.19: An example of helical gears. Source: Industrial Quick Search [2024a]

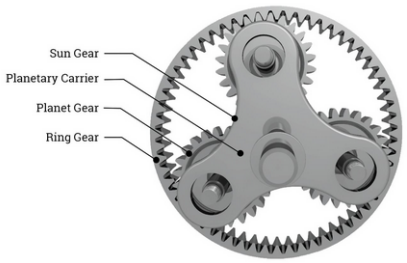


Figure 1.20: A planetary gear. Source: Industrial Quick Search [2024b]

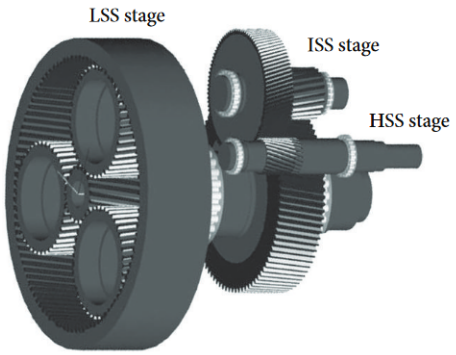


Figure 1.21: A multistage gearbox. Source: Bajric et al. [2015]

For larger wind turbines multistage gearboxes are used. Multistage indicates that they consist of several gear stages. These are often planetary gears. Figure 1.20 shows a planetary gear. Their advantage is that the torque is divided between several points. A geared drive has normally between three and five epicyclic gears, combined with a sun gear and a hollow wheel. Wind turbines with a rated power of more than 2.5 MW have a second step that also consists of epicyclic gears [Siegfriedsen, 2014].

To further spread the torque, manufacturers use the capacity branching concept. This distributes the torque evenly over several steps. The advantage of this is that the individual steps are subjected to a smaller and more even loading. It also allows for a more compact construction [Siegfriedsen, 2014].

The gearboxes of the larger wind turbines also contain a spur gear. This gear is used to achieve an axial displacement between the ingoing and outgoing shafts. This is necessary because through the gearbox runs a hollow shaft that is used for passing the cable through to the rotor [Siegfriedsen, 2014].

Multistage gearboxes have made torque densities of $200Nmkg^{-1}$ and speed-increasing ratios of up to 200 (this is the ratio of the rotation speed of the input and the output) possible [Nejad et al., 2022]. Figure 1.21 shows a multi-stage gearbox with a single planetary stage low-speed stage (LSS) and several helical stages, called intermediate-speed stage (ISS) and high-speed stage (HSS).

3 Failure modes

A wind turbine is a complex piece of machinery containing many different components. Since components tend to fail eventually, a wind turbine has many different failure types or modes (*failure mode* and *failure type* will be used interchangeably). Failures can have an important impact on the profitability of a wind turbine. For this reason, the industry looks with great interest to the failure rates and costs of the different components. Several overview papers have been written on the topic (e.g. Zappalá and Tavner [2022], Li et al. [2022], Dao et al. [2019], and Carroll et al. [2016]).

Unfortunately, there is not one definition of *failure* that is used in all the literature. There are several different definitions in use. In general, a failure is defined as an event requiring a repair action. This means that a person needs to be sent to the wind turbine for maintenance. However, others have defined it as downtimes with a length of more than a certain minimum duration [Zappalá and Tavner, 2022].

In the last couple of years, several studies on the topic of the reliability of wind turbine components have been performed. Comparing the different results of these studies is however not trivial due to several issues. Firstly, there is a lack of standardization in

the collection, processing, and publication of reliability, availability and maintainability (RAM) data. Secondly, technologies of different maturity, in different operating years, are expected to fail differently [Cevasco et al., 2021]. The goal of this section is however not to compare the results in different studies, but to present an overview of some of the most important reviews on the topic.

Carroll et al. [2016] presents an extensive survey on the failure rate of wind turbines. It found 8.3 failures per wind turbine per year, i.e. 6.2 minor repairs, 1.1 major repairs, and 0.3 major replacements. The researchers also found that the failure rate for offshore wind turbines is substantially higher (about 8 times) than for onshore ones. The failure rate is for both correlated with the wind speed. The higher failure rate of offshore wind turbines is explained by the authors by higher average offshore wind speeds, differences in maintenance, on average higher active power of offshore wind turbines, and harsher environment.

The pitch and hydraulics systems, which cause most failures according to Carroll et al. [2016], make up 13% of the failures. A group called "Other components" which consists of auxiliary components (e.g. lifts, ladders, hatches, ...) causes 12.2% of the failures. The generator, gearbox, and blades are the third, fourth, and fifth most important categories. They cause respectively 12.1%, 7.6%, and 6.2% of the failures [Carroll et al., 2016].

Gearbox and generator failures make up 95% of the major replacement category, with the gearbox experiencing more failures than the generator. The reason for this is that failures of those two components generally result in expensive replacements (more than 10,000 euros). The power supply/converter issues are the third lowest category with relatively speaking many major repairs (repairs that cost between 1,000 and 10,000 euros). This is mainly due to insulated-gate bipolar transistor (IGBT) pack replacements [Carroll et al., 2016].

Another extensive overview of failure rates is done in Dao et al. [2019]. In this research, several wind turbine databases (in total 18,000 wind turbines) are analyzed. They find the highest median failure rates for the electrical system, blades and hub, generator, gearbox, hydraulics, and control system, and the longest downtimes for the gearbox, generator, shafts, bearings, and structure. The analysis indicates also that there are large differences between the datasets. Another relevant overview is given in Li et al. [2022]. It is based on 76 wind turbines located in China. These wind turbines experienced 2.57 failures per wind turbine per year. The generator, the cooling, and the hydraulics experienced the most failures.

In the research presented in this thesis, data is used from three large offshore wind farms. Two of the wind farms did experience major failures for which information from the wind turbine operator has been made available. For Wind Farm 1 (WF1) there is information on mechanical failures, i.e. gearbox bearing failures, and electrical failures, i.e. generator short-circuits. For Wind Farm 2 (WF2) there is information on mechanical failures, i.e.

generator bearing failures. Wind Farm 3 (WF3) did not experience any large failures during the observation period. However, it is assumed that, since the wind farm consists of the same wind turbine types as WF1, it might suffer from the same issues as WF1. A more detailed analysis of these failure modes is given in the next chapter.

4 Data sources and signals

The availability of data for wind turbine research has always been complicated. This is mainly because most data is in the hands of private companies, who are not always willing to share the data. This is considered one of the great challenges for the digitalization of the wind industry [Clifton et al., 2023].

In the past, this resulted frequently in the usage of simulated failures. The disadvantage of this is, that the validity of the analysis depends on how close the simulated failures approach reality. In the case of failure detection for wind turbines, this is not easy to assess, since this is an area of research that is developing. How specific failures show themselves in the data is still not very well understood, and is the subject of research. This means that simulating failures realistically is difficult.

In recent years, an increasing number of collaborations were set up between academia and the industry, which resulted in the sharing of data. This has made real data available for academic research, be it although most of it is covered by non-disclosure agreement (NDA). This has some implications for the reproducibility of research results. The data cannot be shared, which means that research cannot be reproduced with exactly the same data. Nevertheless, condition monitoring and fault detection research is a heavily researched domain. There are in general three ways for doing this: SCADA-based, vibration-based, and acoustics-based [Nejad et al., 2022].

SCADA-based condition monitoring uses SCADA data as input. The SCADA data contains in general hundreds of signals, a.o. temperature signals collected by temperature sensors (see Figure 1.22a). Most often the resolution of the data is 1 observation for each 10-minute window (the data is sampled at 1hertz (Hz) and subsequently aggregate values, i.e. mean, min, and max, are calculated for each 10-minute window). Other resolutions do occur but are less frequent, e.g. 1-minute and 1-second.

Vibration-based condition monitoring is the most widely used. This has to do with the ease of instrumentation and the reliable response to the formation of damage [Randall, 2011]. Vibration data can be collected by placing accelerometers (see Figure 1.22b) on the components of interest. These observe the vibrations of the component in operation. Changes in these vibrations, e.g. changes in frequency and/or amplitude, can be indicative of damage that is forming. The analysis of the spectrum of the vibrations can also be used for failure diagnosis [Nejad et al., 2022].

(a) An example of a typical PT100 temperature sensor that is used in wind turbines. Source: Lund and Sorensen A/S [2024]

(b) An example of an accelerometer that typically is used for vibration analysis on wind turbines. Source: Wilcoxon [2024]

(c) An example of an acoustic sensor that can be used for capturing acoustic emissions in wind turbines. Source: Cornel et al. [2021]

Acoustics-based condition monitoring is relatively new. This method uses acoustic emissions to detect failures, such as for example sudden cracks. To this end, acoustic sensors are used (see Figure 1.22c). The technique has been used for detecting cracks in pressure tanks, and research has been done on applying it to rotating components (e.g. bearings). Challenges are distinguishing the relevant acoustic emissions from the background [Nejad et al., 2022]. Research is also ongoing on combining several of these techniques. For example, monitoring acoustics emissions and electrical effects, or combining it with SCADA data [Nejad et al., 2022].

The research presented in this thesis combines 10-minute SCADA and status log data. More specifically, SCADA data temperature signals are used together with wind turbine context information, e.g. active power, rotor speed, wind speed, ... A more detailed overview is given in the next chapter. The status log data consists of events. These can be warnings or errors, which can be valuable information because the description of the events contains expert knowledge in the form of alarm thresholds on certain signals. They also give extra context on the condition of the wind turbine.

The data used in this thesis is real data coming from several operational offshore wind farms. The advantage is that this makes the validation of the methodologies possible in realistic scenarios. The disadvantage is that it makes the analysis more complex. An in-depth discussion of this is given in the next chapter.

5 Estimating the performance of the methodologies

An important part of the validation is estimating the performance of the developed framework. To this end, a performance estimation procedure, consisting of several different techniques, is set up. Figure 1.23 gives a schematic overview.

The failure prediction and diagnosis methodology or framework consists of different methods, i.e. a failure prediction and a failure diagnosis step. The failure prediction step can be further subdivided into modeling the normal behavior and identifying anomalies in the prediction error. For each step, the performance is estimated. Since each step has a different goal, the performance estimation is also different.

For the normal behavior modeling steps, two performance metrics are used. Firstly, the MAE on healthy test data. This indicates how well the different models model normal behavior. A smaller MAE is better. Secondly, the ratio of the MAE on unhealthy and healthy data is calculated. This metric is called *UHH-ratio*. This indicates how well the model can distinguish between unhealthy and healthy data. A larger UHH-ratio is better. How healthy and unhealthy data are selected is explained in Chapter 3.

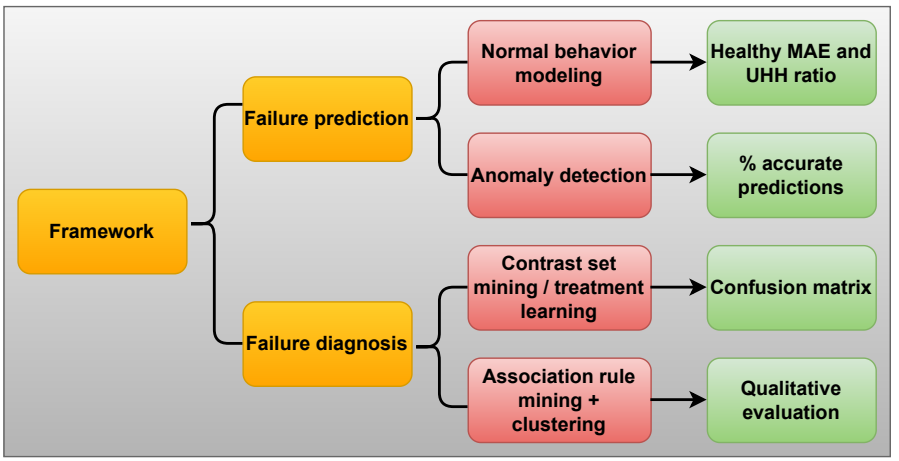


Figure 1.23: Schematic overview validation techniques.

For the anomaly detection algorithms, an accuracy metric is designed. If the anomaly scores exceed a certain threshold before a failure and decrease significantly after the repairs, then this is considered to be a correct detection. The accuracy of the detector is the percentage of correct detected failures. A higher accuracy is better.

The failure diagnosis section tests two different algorithms, i.e. an algorithm that is based on contrast-set mining and treatment learning, and one based on association rule mining and clustering. Each of the algorithms requires a different validation procedure. The performance metric for the first algorithm is the accuracy calculated on a confusion matrix. A higher accuracy is better. How the confusion matrix is determined is explained in Chapter 4. For the second algorithm, the performance is estimated by calculating how often the methodology predicts the correct failure mode. A higher percentage of correct detections is better.

Calculating a confusion matrix on real data is not trivial. The temperature signals that are used for identifying specific failure types are not only influenced by the failures of interest but also by other failures or problems. These failures are often unknown to us, because the information has not been shared, or they are also unknown to the wind turbine operators. The problem is that these unknown failures can also cause anomalies, which can not be linked to a failure. The anomalies would be wrongly considered false positives, which would result in an unjustified poor performance score for the algorithm.

One solution to this problem is only taking a time interval around the failures of interest into account and ignoring all the rest. This is what is done to validate the first model of Chapter 4 (for more details see Chapter 4). Another solution is calculating an accuracy metric that does not require the calculation of a confusion matrix (see validation anomaly detection techniques in Chapter 3). Neither technique is perfect, but they do give a good indication of how well the models perform.

6 High-level overview state of the art and research gaps

A lot of research currently focuses on the prediction and identification of failures. The scientific literature contains many examples in which artificial intelligence (AI), e.g. ML, is used to identify anomalies in certain wind turbine signals. Currently, several methodologies compete in the domain. Helbing and Ritter [2018] distinguishes model-based signal processing and data-driven methodologies.

A more detailed subdivision can for example be found in Black et al. [2021]. A distinction is made between trending, clustering, NBM, damage modeling, alarm assessment, and performance monitoring. Most current state-of-the-art research uses the NBM approach, which has already shown its value for the detection of anomalies. This is also the methodology that will be used here.

Most current research focuses mostly on the detection of anomalies in wind turbine time series (signals). The investigation ends with a discussion of the generated anomalies, the number of failures that have been detected, and how early they have been found. However, designing an automated failure identification algorithm that interprets the identified anomalies and links them to specific failure types is only rarely done. Notable exceptions are for example Du et al. [2023], Roelofs et al. [2021], Dienst and Beseler [2016], Schlechtingen et al. [2013] and Schlechtingen and Santos [2014], which do take steps towards automatic root-cause analysis.

Even though it is often ignored, automated failure diagnosis and root-cause analysis have become increasingly important for the industry. The abundance of data that is currently available makes it for humans more challenging to interpret the anomalies correctly. However, automating the analysis of the data is not trivial and has several major challenges.

1. **Limited number of failure examples**: The number of failures that can be used to learn from is generally small. This is because most wind turbines (and industrial machines in general) fail rarely. This means that during an observation period, there are often only a limited number of examples of a specific failure type. The failure list that is made available by the wind turbine operator is in general also not exhaustive. This is because it is often quite labor-intensive to extract the list from the data, or

the fact that this information is considered highly confidential, or because certain failures are unknown.

2. **Data errors**: Working with real data (meaning not simulated) has several practical challenges. It contains often measurement errors produced by sensors or information that is incorrectly registered (for example errors in the date of a failure, ...).

3. **Unknown timestamp of first damage**: An important unknown factor when using real data is the exact time the first damage appeared on a component. The moment that a component fails is often not the moment of first damage. The fact that this is unknown makes it more difficult to validate a model. It can not be verified that the first appearance of anomalies coincides with the first damage. Furthermore, there is no information on whether a component is damaged, or how severe the damage is, at a certain point in time. This means that during the validation there is uncertainty on whether the generated anomalies are truly indicative of damage, or whether they are simply artifacts or imperfections of the model. The best that can be done is to check whether the anomaly concentrations can be related to known failures and whether the concentrations are significantly higher before a failure than afterward.

4. **Identifying healthy data**: An important element of the NBM methodology is the creation of a healthy training dataset. When working with real data this is not a trivial task. It is difficult to detect when a wind turbine is truly healthy. The data is not labeled. The best that can be done is selecting data that is most likely healthy. This can for example be done by excluding data close to known failures. To be extra certain also the data that is close to forced shutdowns can be excluded. The size of the exclusion interval around the failures or forced shutdowns is up to the expert to determine and depends on the failure type. This method will only result in approximately healthy data because not all failures are known, and it is unknown whether a component is damaged or not.

5. **Explainability**: Failure prediction and diagnosis models are in general used in engineering environments. A black-box model might generate accurate predictions, however, it is not interpretable. This makes the interaction with engineers more difficult. It assumes that the engineers take a leap of faith when using the models. Understanding why the model makes a certain prediction becomes difficult.

6. **Comparative evaluation of models**: Most data from wind turbines is proprietary, which means that in general, it is covert by an NDA. Reproducing the results of papers is therefore often difficult. There is no straightforward and fast solution for this problem, since after all, most data is considered sensitive by most companies, and can not be shared.

7 Research objectives

The main goal of the research presented in this thesis is the development of a failure prediction and diagnostics framework that can be used in a realistic industrial context. This means that the constraints that apply in this context are taken into account. These were identified through a close collaboration with several industrial partners (mainly wind turbine operators). The constraints are the following:

1. The data quantity requirements should be such that rapid deployment of the framework on new wind farms is possible. This means that the minimum required training data should be as low as possible. Requiring several years of training data is not acceptable since this would mean that during the first years of operation, no failures can be detected.

2. The computational complexity of the framework should be as low as possible. High computational complexity has an economic cost. It requires larger and more expensive servers which consume more electricity. The training of the models also takes longer, which makes it more difficult to quickly iterate over different configurations, and react to new requests.

3. The maintenance complexity of the framework should be limited as much as possible. This means that highly complex pipelines with many intermediate steps should be avoided as much as possible. Maintenance of software is labor-intensive and expensive. This means that there is an economic incentive to reduce the complexity of the pipelines.

4. The results and the way the framework generates them should be as transparent as possible since the framework will be often used in engineering environments. Engineers want to understand the results and how they are calculated. This means that the reliance on *Black Box*-algorithms should be minimized.

Taking these constraints into account, several different configurations of failure prediction and diagnostics frameworks are developed and tested. This means that this thesis is to an important extent a comparative analysis of different techniques. This validation is based on data from three operational offshore wind farms.

The main goal of this thesis can be divided into two sub-goals.

1. The development of a failure prediction methodology that reliably detects abnormal values in component temperature signals. Such abnormal behavior is considered to be a proxy for an imminent component failure.

2. The development of a failure diagnostics framework that can automatically interpret and classify the patterns in the component temperature signals or the anomalies generated by the temperature signals.

The development of the failure prediction methodology can be further subdivided into several smaller sub-goals:

1. Develop several methodologies that can model the normal behavior of wind turbine component temperatures.
2. Validate the different normal behavior modeling methodologies on data from three operational offshore wind farms.
3. Compare these methodologies, discuss their pros and cons, and performance.
4. Develop several anomaly detection techniques that can analyze the difference between the observed component temperatures and the predicted normal component temperatures.
5. Validate the anomaly detection techniques on data from three operational offshore wind farms.
6. Compare these techniques, discuss their pros and cons, and performance.

The development of the failure diagnostics methodology can also be subdivided into several smaller sub-goals:

1. Develop several methodologies that can suggest likely failure modes given the component temperature signals/anomaly scores generated by the failure prediction methodology, and status codes.
2. Validate the methodologies in data from operational wind farms.
3. Compare the methodologies and discuss their pros and cons, and performance.

8 Scope definition

The framework that is developed in this thesis can be applied to all (industrial) machines for which component temperatures (SCADA data) and status logs are available. These data sources are often available for larger machines. There is no data format-related reason why this would not be possible. The scope of this thesis will however be limited to wind

turbines. This decision was taken for practical reasons. Data from offshore wind turbines was available and there were collaborations with several wind turbine operators, which facilitated the interpretation of the data and the results. Other types of machinery have not been used as a validation case. For this reason, it is for the moment unclear how well the framework would perform on those machines. However, there is no particular reason why it should not work. The validation on data of different types of machinery is a topic for future research. The data contains information from two types of wind turbines. However, there is no reason to assume that the results would be fundamentally different for other wind turbine types. The SCADA signals used are component temperature signals. A property of these signals is that they are time series. The framework is however not limited to temperature signals.

9 Research strategy

The research presented in this thesis has as its goal the development of an automated failure detection and diagnosis framework for wind turbines. The methodology does this by using data that is readily available from most wind farms, e.g. SCADA and status log data. The training of the methodology requires however also a list of failures (not necessarily exhaustive). This can normally be given by the wind turbine operator or extracted from the status logs.

As indicated in section 6, this research has several challenges that need to be handled. This is done as follows:

- Industrial machines and wind turbines do not fail often. This challenge is handled by using unsupervised techniques for anomaly detection and pattern mining techniques for rule extraction.
- The data quality of real machines is not perfect. Data errors are solved through thorough data analysis of which the results are used to set measurement error thresholds and cleaning of the failure information.
- Data from real machines does not contain a healthy-unhealthy label which indicates whether one or more components are damaged or not. Healthy data is selected by excluding data that can be associated with a failure, an error, or a warning. This methodology is not perfect. The imperfections that remain are neutralized by careful selection of the appropriate model.
- The methodology should be usable in an engineering context. This means that explainability is important. The problem of explainability is tackled by using techniques that are as transparent as possible, e.g. pattern mining and rule-based classification for the failure diagnosis.

- It is unknown when a component is damaged. This problem is solved by making some expert knowledge-based assumptions. Bearing failures in general form slowly over time. By excluding large windows of data around bearing failures, it is relatively safe to assume that the data associated with the damaged component is identified. It is however more difficult to identify damage to components that are associated with less known failures or failures that do not concern the wind turbine operator as much, since it is likely in that case that there is less information on the failures available, which makes it more difficult to identify them reliably). To exclude this data, all data that surrounds forced outages of the wind turbine are also removed.

The framework consists of two parts. The first part is the failure prediction methodology that searches for abnormal behavior in the component temperature signals of the wind turbines. To this end, a pipeline is designed that uses several machine learning and statistics-based techniques. Statistical, shallow ML and deep learning techniques are tested for the modeling of normal temperature behavior. Several statistical techniques are tested to analyze the difference between the observed and predicted temperatures.

The second part of the framework is a methodology that interprets the patterns in the temperature signals or the identified anomalies. For this data mining techniques are extensively used to extract patterns from the data. These patterns are in the next step post-processed. The result is used to determine the failure mode.

10 Main contributions

The contributions of this thesis are the following.

1. Industrial usability: The methodologies developed in this thesis have been constructed in accordance with industry requirements. These requirements were formulated by the industrial partners. These are high accuracy, ease of use, easily maintainable, low computational cost, interpretability, and short start-up time.

2. Validation on real failures: The methodologies are validated and compared using data from real operational wind farms. These wind farms experienced several types of real failures. This means that no assumptions need to be made on how a failure looks in the temperature data.

3. Prediction of real failures: Several pipelines are developed that can be used for failure prediction. An extensive analysis is done of the advantages and disadvantages of statistical, shallow machine learning, and deep learning methodologies. The end result is an autoencoder-based pipeline that can detect several different failure types reliably well in advance. This has economic value for the industrial partners.

4. Identification of real failures with data mining: Several pipelines are developed, validated, and compared that can be used for automated failure diagnosis. Automated failure diagnosis for wind turbines is still in its infancy. Only a limited amount of research has been published on the topic. A contribution of the research in this thesis concerns exploring the usefulness of data mining algorithms. It develops a methodology that performs well and is interpretable by domain experts.

11 Structure of the thesis

The methodology presented in this thesis consists of several blocks. Figure 1.24 gives an overview. Each block is treated in a separate chapter. The structure of the thesis is as follows:

- The second chapter discusses the structure of a wind turbine, and gives a detailed overview of the properties of the data, e.g. data quality, signal correlations, autocorrelation, ..., and failures. The analysis should give an idea of potential pitfalls and issues with the data that might complicate the analysis.
- The third chapter develops a failure prediction methodology (block 1). Several techniques for modeling the normal behavior of the wind turbine temperature signals are discussed. Furthermore, several anomaly detection techniques that are used to analyze the difference between the observed and predicted temperatures are also discussed. The advantages and disadvantages of the different methodologies are discussed.
- The fourth chapter focuses on automatic failure diagnosis (block 2). Several methodologies are presented. Their results are discussed, together with their advantages and disadvantages.
- The fifth chapter formulates a general conclusion and discusses potential future work.

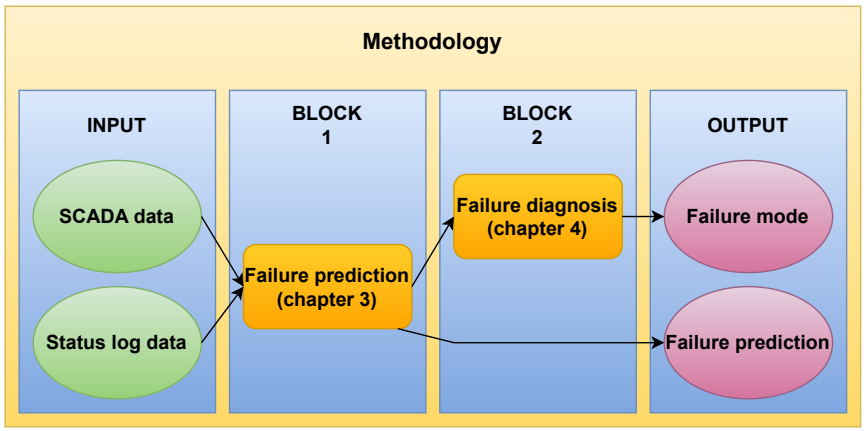


Figure 1.24: Overview of the different blocks or steps in the methodology.

2 | Description of the data and failure cases

1 Overview of the characteristics of the data

In this section, several properties of the data are discussed. Before any modeling is started, it is important to know the data. This can be done through a first inspection or some preliminary data analysis. This step is important and can not be ignored because it can point to certain problems with the data that might result, if left untreated, in unwanted results. This section is built up as follows. The first part describes several characteristics or properties of the data. The second part gives an overview of the different failure modes that will be used for the validation. The third part discusses temperature signals and status code ontologies. At the end of this chapter, a conclusion is formulated.

1.1 Description of the wind farms

The research presented in this thesis is based on data from three operational offshore wind farms, called WF1, WF2, and WF3. The size, rated power production, and observation period of the wind farms vary quite a lot between the different wind farms.

WF1 consists of over 20 MW wind turbines with a rated active power of over 9 MW. The observation period is around 3 years in length. WF2 consists of over 20 wind turbines with a rated active power of over 3 MW. The observation period for this wind farm is

Windturbine name	**Datetime**	**Signal 1**	**Signal 2**	...	**Signal k**
WT1	1/1/2012 00:00:00	9.0	11.5	...	10.0
WT1	1/1/2012 00:10:00	9.5	11.0	...	10.0
...	...	...	...	...	...
WTn	1/1/2012 00:00:00	8.0	12.0	...	9.5
WTn	1/1/2012 00:10:00	7.5	12.5	...	9.5

Table 2.1: Example of the structure of the SCADA data. The example is completely artificial.

more than 10 years in length. WF3 consists just as WF1 of over 20 wind turbines with a rated active power of over 9 MW. This wind farm came into operation more recently than WF1. Because of this, the observation period is only around 1.5 years in length.

1.2 The structure of the data

The data that is used in this research comes from several different sources, i.e. SCADA data, status log data, and failure logs. The SCADA data is typically in tabular format when it is used as input. This means that it consists of rows and columns. Each column corresponds to a signal or wind turbine information, and each row is an entry or observation that coincides with a certain moment in time. For this research, SCADA data with a resolution of 10 minutes is used, which means that there is an entry every 10 minutes.

The data that is made available by the wind turbine operators consists of SCADA and status log data. The SCADA data consists of data from more than 100 signals, e.g. temperatures of certain components, the active power, nacelle temperature, ..., of the wind turbine. Not all of those signals are useful for the cases at hand. Table 2.1 shows an artificial example of SCADA data. The status log data consists of status code events. During the operation of a wind turbine, the machine occasionally throws warnings or errors. These are registered in the status log. This log contains information on the wind turbine that threw the warning or error, the status code, a description of the status code, a start time, and an end time. The event-based format of the status logs is not ideal to combine it with the SCADA data. For this reason, it will be transformed (see next chapter). Table 2.2 gives an artificial example of status log data.

For some wind farms, the wind turbine operator also made available a file containing the occurrences of specific failure events. This is not the case for all wind turbines, and in general, the list is limited to only a couple of failure types. Such failure event files contain often the following information: the wind turbine name, the timestamp of the

ID	Code	Name	Timestamp start	Timestamp end	Comment
1	120	Description.	1/1/2012 00:11:56	1/1/2012 00:28:01	
12	458	Description	1/1/2012 01:45:32	12/03/2012 18:45:35	Failure

Table 2.2: Example of the structure of the status logs. The example is completely artificial.

Windturbine name	Timestamp	Failure type
WT1	1/1/2012	Generator failure
WT3	8/5/2013	Gearbox failure

Table 2.3: Example of the structure of a typical failure event file. The example is completely artificial.

failure (sometimes the start and end of the event), and the failure type. Table 2.3 gives an artificial example of a failure event file.

1.3 Overview of the selected signals and status codes

The SCADA data contains information from hundreds of sensors. For the research case at hand, only a subset is relevant. Which signals these are depends on the sensors that are installed on the wind turbines. Modern wind turbines (e.g. the 9.5 MW wind turbines) contain more sensors than older ones (e.g. the 3 MW wind turbines). Some signals are available for all wind turbines and are always selected because they give information on the context of the wind turbine, i.e. wind speed, outside temperature, rotor speed, nacelle temperature, and torque.

For each wind farm, several signals are selected that are related to the generator and gearbox (if required). The availability of these signals depends on the wind turbine type and the wind farm (even wind turbines of the same type but installed at different wind farms can sometimes have different signals in the SCADA data). The component temperature signals that are selected from the SCADA data are shown in Tables 2.4, 2.5, and 2.6. Each signal is given an ID that will be used where necessary in this thesis. The environmental, e.g. wind speed, outside temperature, ... and operational signals, e.g. rotor speed, generator speed, ..., that are used to model these temperature signals are not included in the tables, to keep them more compact. The component temperature signals will be used to find anomalous behavior. They are valuable sources of information for mechanical failures. Furthermore, the SCADA data contains by default a substantial amount of them, which makes them interesting for analysis.

Signal name	Signal ID
Generator cooling water temp.	1
Main bearing temp.	2
Generator rear bearing temp.	3
Generator front bearing temp.	4
Generator phase 1 temp.	5
Generator phase 2 temp.	6
Generator phase 3 temp.	7
Generator phase 1M1 temp.	8
Generator phase 1M2 temp.	9
Generator phase 2M1 temp.	10
Generator phase 2M2 temp.	11
Generator phase 3M1 temp.	12
Generator phase 3M2 temp.	13
Gearbox BPNREC bearing temp.	14
Gearbox A2BPNREC bearing temp.	15
Gearbox A2BPREC bearing temp.	16
Gearbox A1PREC bearing temp.	17

Table 2.4: Temperature signals from the SCADA data from WF1.

Signal name	Signal ID
Generator front bearing temp.	18
Generator rear bearing temp.	19
Generator cooling water temp.	20
Generator phase 1 temp.	21
Generator phase 2 temp.	22
Generator phase 3 temp.	23
Generator slipring temp.	24
Gearbox hollow shaft bearing (gen. end) temp.	25
Gearbox hollow shaft bearing (gen.) temp.	26
Gearbox hollow shaft bearing (middle) temp.	27
Gearbox hollow shaft bearing (rotor end) temp.	28
Gearbox oil L1 temp.	29
Gearbox oil temp. basis	30

Table 2.5: Temperature signals from the SCADA data from WF2.

Signal name	Signal ID
Generator bearing temp.	31
Generator phase 1 temp.	32
Generator phase 2 temp.	33
Generator phase 3 temp.	34
Generator bearing 2 temp.	35
Generator air outlet 1 temp.	36
Generator air outlet 2 temp.	37
Generator lubrication inlet temp.	38
Generator phase 1B temp.	39
Generator phase 2B temp.	40
Generator phase 3B temp.	41
Converter generator water cooling temp.	42

Table 2.6: Temperature signals from the SCADA data from WF3.

A second source of information are the status logs. These contain status codes that are triggered when certain events occur. Wind turbines can trigger hundreds of unique status codes. However, just like it is the case with the SCADA data, only a subset is relevant. For the problem at hand, only those that can be associated with errors (which might result in a forced shutdown) and warnings are selected. Many status codes are purely informational or are related to certain communication events. These contain no useful information and for this reason, they are not used.

1.4 The data quality

Getting an idea of the data quality is important before an analysis is started. Poor quality can result in problems during the analysis. For example, long stretches of missing data might make some analyses invalid, or generate unrealistic results. To assess the quality of the data, the following properties are analyzed: the number of missing values, and the number of measurement errors.

Missing values

The SCADA data for the three wind farms contains missing values but their number is limited. A suitable metric to estimate the significance of the missing value issue is the missingness ratio, which is the ratio of the number of missing values to the total number of values.

The missingness ratio is low for all three wind farm datasets. For WF1 the average over all relevant signals is 0.037, the minimum is 0.03 and the maximum is 0.05. For WF2 the average, minimum, and maximum over all relevant signals are around 0.01. For WF3 the average is around 0.01, with a minimum smaller than 0.005 and a maximum of around 0.01.

The missingness ratio does not take into account the missing observations during the offline time of the wind turbine. Furthermore, these numbers apply to the raw data, not the preprocessed data. During preprocessing new missing values are created (due to the removal of measurement errors, see further) which will increase the missingness ratio.

An important question that needs to be answered is how the missing values are generated since this will indicate how the missing values are best treated. Certain types of missingness are more problematic than others. In general, missingness is divided into three categories [van Buren, 2021]. Figure 2.1 gives an overview of the impact of naively removing missing values from the data.

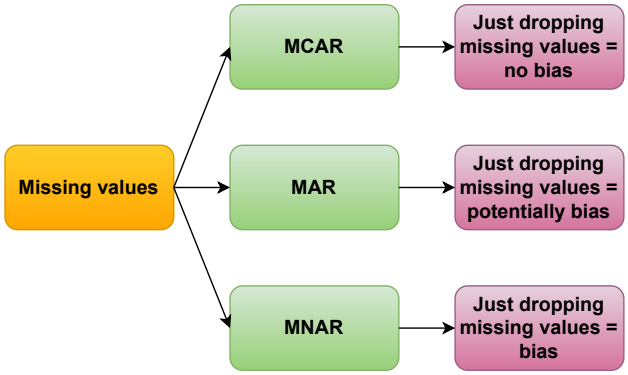


Figure 2.1: Schematic overview of the impact of removing missing values from the data given the nature of the missingness. MCAR stands for Missing Completely At Random, MAR stands for Missing At Random and MNAR stands for Missing Not At Random.

Firstly, missing completely at random (MCAR), which is the situation where the probability of being missing is for all data (observed/non-missing and unobserved/missing) the same. This means that the missing data mechanism is unrelated to the signal that contains the missing values. It also means that there is no structural difference between the observations that are missing and non-missing. This type of missingness does not introduce bias, since the observed data is simply a random sample of all data. However, the MCAR conditions are strong and are not often realistic in applications [van Buren, 2021].

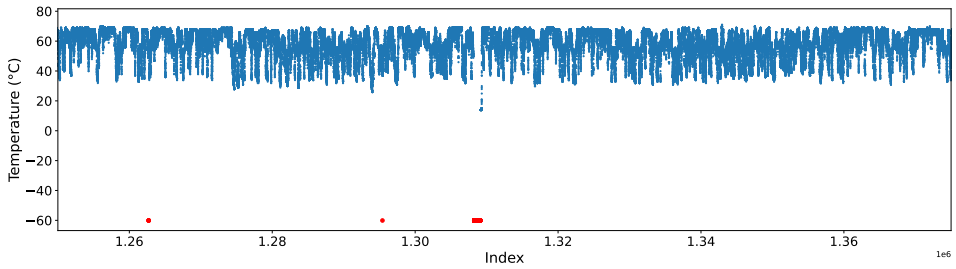
Secondly, missing at random (MAR) is a situation where the missing data mechanism depends on another observed signal. This means that the probability of being missing depends on another signal than the signal that contains the missing values. A hypothetical example is when the missingness in a temperature signal of interest is unrelated to the actual temperature value of the signal but depends on the operational state of the turbine. The state can be determined, which means the missingness is MAR. Handling the missing values by just using the non-missing observations can result in bias if the signal to which the missingness is related is not taken into account. Taking it into account can remove the bias [van Buren, 2021].

Thirdly, missing not at random (MNAR) is a situation where the missing data mechanism depends on an unknown or unmeasured signal. In other words, it means that the probability of being missing depends on one or more unmeasured signals. A hypothetical example would be if the missingness in a temperature signal depends on the (unknown) wake effects of other wind turbines in the wind farm due to certain stresses that they create on components in the drivetrain. This type of missingness is difficult to handle properly. One option is to collect more data on the causes of the missingness, another is to do what-if analysis to see what the impact is under different conditions [van Buren, 2021].

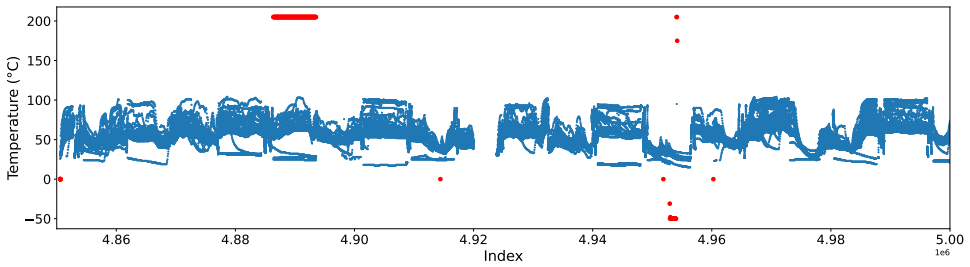
An analysis of the missing values in the data from WF1, WF2, and WF3 showed the following. If values are missing then this is often because the wind turbine is powered down or offline. However, this state is not interesting for our research because it is impossible to use temperature signals for the identification of anomalous behavior if the wind turbine is powered down. For this reason, the power-down or offline state is excluded from the missingness ratio calculations. In some cases, data is missing due to communication errors between the wind turbine and the database. These communication errors are most likely unrelated to the values in the signals because they are caused by external effects [van Buren, 2021].

Measurement errors

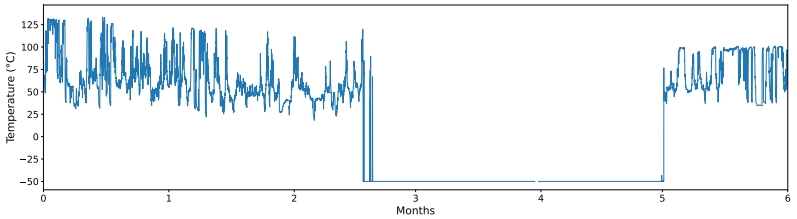
Another problem that needs to be handled is measurement errors. When using real data it can be expected that occasionally a sensor will generate values that are simply wrong. An analysis of the data shows however that the number of measurement errors in the data of all three wind farms is fairly limited. Occasionally measurement errors can be found in the data. These are in general quite easily recognisable because they generate unrealistic values (both on the low and the high side). Figures 2.2a and 2.2b give some examples for WF1 and WF2. The measurement errors on the lower end are clearly errors. Temperatures of -50degrees celcius (°C) or -60°C for certain components can not be explained. Given the environmental temperatures, any values below 0°C can not be explained physically and must be attributed to errors made by the sensors.



(a) Several examples of measurement errors for a component temperature signal for a wind turbine in WF1. The measurement errors are indicated in red.



(b) Several examples of measurement errors for a component temperature signal for a wind turbine in WF2. The measurement errors are indicated in red.



(c) An example of a measurement error that persists for a long time in a signal of WF3.

Figure 2.2: Examples of measurement errors.

The examples in Figures 2.2a and 2.2b show measurement errors that are sudden but shortlived. However, in certain situations, the measurement errors persist for a long time until the sensor is replaced. Figure 2.2c gives an example of a sensor error that returned for more than a month unrealistic values.

The long persistence of certain measurement errors has implications for how they can be handled. If the problem persists for a long time, imputation techniques such as linear interpolation are not suitable because they can result in the generation of unrealistic observations. For this reason, we opted to replace the measurement errors with missing values which will be removed later on. This will be explained in more detail in the next chapter. The consequence of this method is that during the period the measurement error persists, no anomaly detections can be made.

All three wind farm datasets contain measurement errors that show themselves as abnormally low temperatures. WF2 is the only wind farm for which there are clear measurement errors on the higher end. These errors are harder to identify because they obtain values that are less clearly erroneous from a physical or engineering standpoint. Nevertheless, they can be identified by the fact that many of them have the same value of 205.0°C. These are probably default values or maximum values for the sensor and are not related to real temperatures.

Abnormally high values that are isolated from the rest can be measurement errors. In those cases, it is however even less clear which measurement errors are caused by sensor failures and which are caused by damage to a component. This means that removing those values is not without risk. A possible guideline is using the fact that damage to components does not show itself in general as a single abnormally high value but as a cluster of values. In the anomaly detection chapter, several techniques will be discussed on how these measurement errors are handled.

To get an idea of how big the measurement error issue is, the measurement error ratio for each temperature signal for each wind farm is calculated. Only the negative values are taken into account because they are more likely to be measurement errors.

WF1 contains a small amount of measurement errors. The number of measurement errors is for all signals lower than 0.000005. For WF2 the measurement error ratios of the signals are slightly higher. The average for all signals is around 0.000062. For the signal with the highest measurement error ratio, the value is 0.0006. The low values mean that the measurement error problem is not severe. For WF3 the average measurement error ratio over all relevant signals is around 0.0043. The maximum value is around 0.05 or 5%. An analysis of the data showed that some sensors gave unrealistic values during a prolonged period. This will have some impact on the analysis (see Chapter 3).

1.5 Autocorrelation in the signals

In general, SCADA data consists of time series with a sampling rate of one sample per 10 minutes. Other sampling rates are in use, e.g. 1 minute, 1 second, ... but 10 minutes is for the moment the most widespread. Due to this, and the fact that temperatures tend to change slowly, but also the fact that the temperatures are driven to a certain extent by environmental factors that also change relatively slowly, the data contains a lot of long-lasting autocorrelation. Autocorrelation is a phenomenon where a sample is correlated with one or more of the previous samples and occurs typically in time series data.

To get an idea of how strong and long-lasting the autocorrelation is in time series, the autocorrelation function (ACF) can be calculated. The ACF is defined as follows (Brockwell and Davis [2002]):

Definition 1.1 (ACF)**.** Let $x_1, ..., x_n$ be observations of a time series. The sample mean of $x_1, ..., x_n$ is

$$\bar{x} = \frac{1}{n} \sum_{t=1}^{n} x_t \tag{2.1}$$

The sample autocovariance function is:

$$\hat{\gamma}(h) := \frac{1}{n} \sum_{t=1}^{n-|h|} (x_{t+|h|} - \bar{x})(x_t - \bar{x}), -n < h < n \tag{2.2}$$

with h the number of lags or the number of steps the observation is removed from x_t.

The sample autocorrelation function is:

$$\hat{\rho}(h) = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)}, -n < h < n \tag{2.3}$$

It is possible to determine confidence bands for the ACF using Bartlett's approximation. If the autocorrelation falls outside the confidence bands then this can be considered as statistically significant evidence. This approximation is used in Statsmodels plot_acf method, which is used here to create the ACF plots.

Figures 2.3, 2.4 and 2.5 shows the average autocorrelation for the different signals of respectively WF1, WF2 and WF3. The average autocorrelation for each signal is determined by calculating the average for each lag (lags 0 to 1000) over all the wind turbines in the wind farm.

The figures demonstrate that the correlation between the signals and their lags is long-lasting (meaning that the correlation between an observation and its lagged values remains

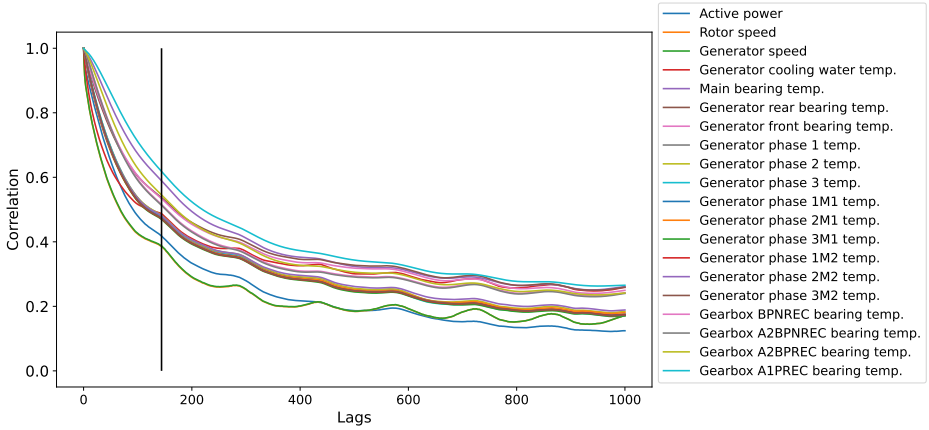


Figure 2.3: Average autocorrelation (up to 1000 lags) for the different signals of WF1. The black vertical line indicates lag 144, which corresponds with 1 day (144 times 10 minutes = 1 day).

relatively strong even for high lag numbers). This phenomenon is more present for WF1 and WF3 than for WF2. For WF2 the correlation descends to 0 around lag 144, which coincides with a time delta of 1 day (1 sample every 10 minutes during a 24-hour window). This can be explained by the influence of intra-day fluctuations in environmental conditions (e.g. ambient temperature, ...).

This is much less the case for WF1 and WF3, where the correlation after 144 lags is still around 0.4-0.6 for WF1 and 0.3-0.6 for WF3. It is unknown at this time why exactly this is, as the methodology for calculating the autocorrelation is the same for all three wind farms. The correlations for the 9.5 MW wind turbines appear to contain also a recurring pattern that has a length of 144 10-minute timestamps (see cyclic patterns in Figures 2.3 and 2.5). This can be explained by the intra-day fluctuations, which create not just a positive correlation between the current day and the previous day, but also between the current day and the day before and the one before, ...

A possible explanation for this phenomenon is that the cooling system of the 9.5 MW wind turbines is less able to neutralize the intra-day fluctuations than one of the 3 MW wind turbines. It seems also that for the 9.5 MW wind turbines, the signals are influenced by other more long-lasting factors (perhaps seasonal fluctuations). This can potentially also be explained by a less-performing cooling system.

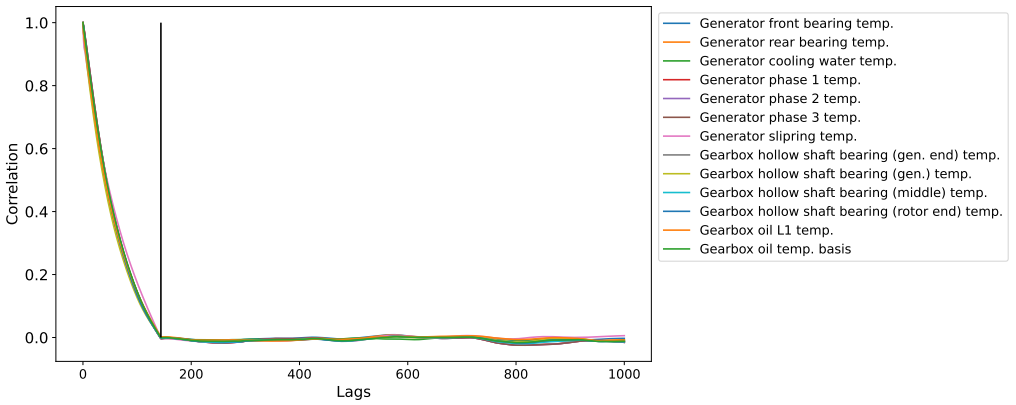


Figure 2.4: Average autocorrelation (up to 1000 lags) for the different signals of WF2. The black vertical line indicates lag 144, which corresponds with 1 day (144 times 10 minutes = 1 day).

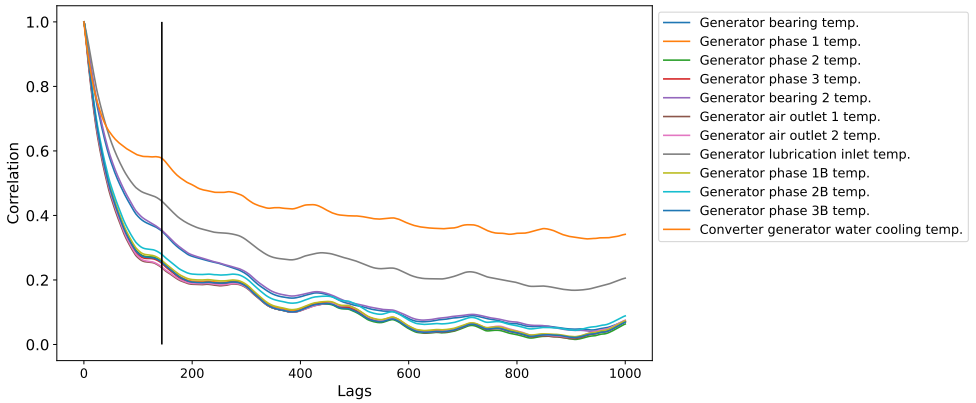


Figure 2.5: Average autocorrelation (up to 1000 lags) for the different signals of WF3. The black vertical line indicates lag 144, which corresponds with 1 day (144 times 10 minutes = 1 day).

1.6 Correlations between the different signals

Before any modeling is attempted, it is always useful to know how the different signals are related to each other. This can be done by calculating the strength and direction of the correlation between the different signals. There are several ways to calculate correlations. Here, the Pearson correlation coefficient (see Eq. 2.4) is used. This method measures the linear correlation between two signals. This means that certain non-linear correlations might be missed.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{2.4}$$

where:

X	= Is a random variable.
Y	= Is a random variable.
σ_X	= Is the standard deviation of X.
σ_Y	= Is the standard deviation of Y.
cov(X,Y)	= Is the covariance of X and Y.

Figures 2.6, 2.7 and 2.8 show correlation heatmaps for respectively WF1, WF2 and WF3. The heatmaps indicate that for all three wind farms, there are strong correlations between the different generator phase temperatures (for WF1 and WF2 ρ is close to 1, for WF3 $\rho > 0.8$). This is as expected since they are located closely together in the generator. The gearbox signals are for WF1 and WF2 (there are no gearbox signals for WF3) also strongly correlated (for WF1 ρ is close to 1, for WF2 $\rho > 0.7$).

The component temperatures in all three wind farms show a relatively strong correlation with the nacelle temperature (for WF1 $\rho \in [0.3 - 0.7]$, for WF2 $\rho \in [0.3 - 0.8]$, and for WF3 $\rho \in [0.4 - 0.8]$). The latter is expected since the components of interest are all located inside the nacelle. An interesting finding is that the component temperatures are not correlated with the ambient temperature. This finding might seem strange when comparing it to the results in the autocorrelation section. There it was found that there were clear intra-day fluctuations in the temperatures of the components. It seems here however that the correlation with the ambient or outside temperature is close to 0.

This result becomes more surprising when considering that the nacelle temperature is strongly correlated with the ambient temperature (for WF1 ρ around 0.7, for WF2 ρ around 0.5, and for WF3 ρ around 0.8). The latter is expected. During warm days the nacelle will heat up more than during cool days. However, since the component temperatures are reasonably correlated with the nacelle temperature, it is surprising that the component temperatures are uncorrelated with the outside temperature. After all the

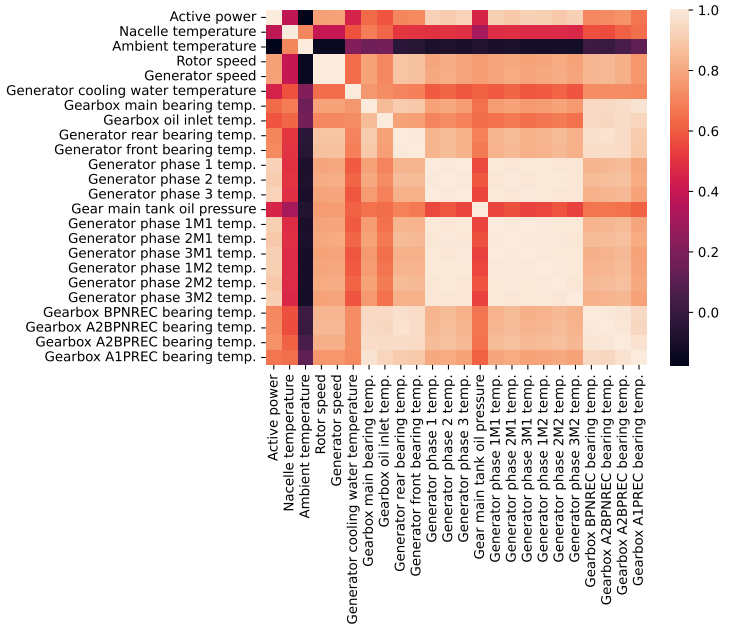


Figure 2.6: Correlation heatmap showing the correlations between the base signals of WF1.

component temperatures are relatively strongly correlated with the nacelle temperature, and the nacelle temperature is strongly correlated with the ambient or outside temperature. It is thus natural to expect that the component temperatures are also strongly related to the ambient temperature. But this appears not to be the case.

Possible explanations for the lack of correlation with the ambient temperature are the insensitivity of the Pearson correlation coefficient for the relation between the two caused by the non-linearity of the relation, a delay between the ambient temperature and the temperature of the components (it takes time for the ambient temperature to have an impact on the component temperatures), and a potential impact of measurement errors. Analysis of the results for WF1 showed that removing all measurement errors results in a mildly positive correlation between the outside temperature and the generator cooling water temperature, the gearbox main bearing temperature, and the gearbox oil inlet temperature. However, the correlation of the ambient temperature with the other temperature signals remains around 0.

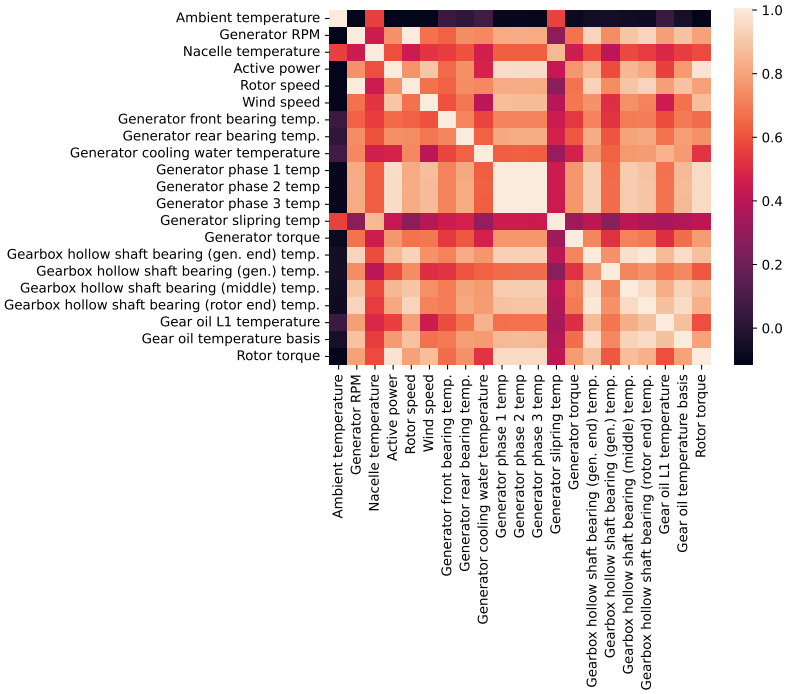


Figure 2.7: Correlation heatmap showing the correlations between the base signals of WF2.

Furthermore, the figures show also that there is often a substantial amount of correlation between the different signals. This is a good thing since this shows that the signals are potentially valuable for modeling the behavior of each other (see Chapter 3).

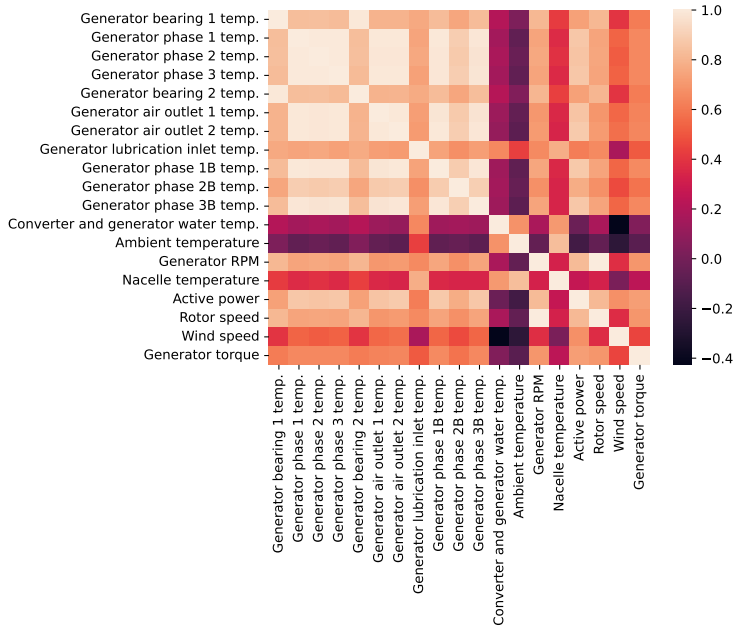


Figure 2.8: Correlation heatmap showing the correlations between the base signals of WF3.

1.7 Analysis of the frequency spectrum of the generator and gearbox signal

The frequency spectrum of a signal, determined through the Fourier transform, can be used to identify recurring patterns. According to Brigham [1988], the Fourier transform identifies the different frequency sinusoids and their amplitudes that together form a certain function. Mathematically this relationship is expressed by Eq. 2.5.

$$S(f) = \int_{-\infty}^{\infty} s(t) e^{-i2\pi f t} dt \tag{2.5}$$

where:

s(t) = The function that must be decomposed into a sum of sinusoids.
S(f) = The Fourier transform of $s(t)$.
i = The imaginary unit.

The analysis can be done efficiently using the fast fourier transform (FFT). The ordinary discrete fourier transform (DFT) takes $O(n^2)$ operations for a dataset of size n, and the FFT only $O(nlog(n))$. This is achieved by a factorization of the Fourier matrix [Strang, 1994], which results in a massive reduction of the computational time when n is large.

Figures 2.9, 2.10, and 2.11 shows the frequency spectra for respectively WF1, WF2 and WF3. For WF1 and WF2 there are clear amplitude peaks for the 1 / day and 1 / year frequencies. These correspond with daily (i.e. day/night) and yearly (i.e. winter/summer) cycles. For these wind farms also other peaks with a higher frequency can be found. For example, there are peaks at 2 / day or 1 / 12 hours frequency. For WF1 even more peaks can be found. It is unclear at this time what causes these.

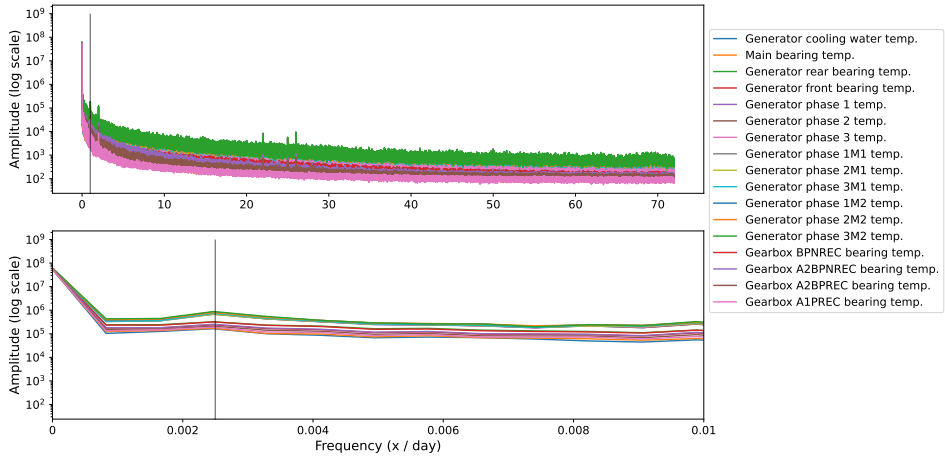


Figure 2.9: Average frequency spectrum for WF1. The top plot shows the amplitudes for all calculated frequencies. The black vertical line indicates the amplitudes for the 1 / day frequency. The bottom plot shows a detailed view of the amplitudes for the frequencies lower than 1 / day. The black vertical line indicates an amplitude peak that corresponds more or less to the 1 / year frequency.

For WF3 no clear patterns can be found in the results. Only for some signals, there is an amplitude peak at the 1 / day frequency. There are no amplitude peaks around the 1 / year frequency. This can be explained by the short observation period. Less than 1 year of data was available for this analysis.

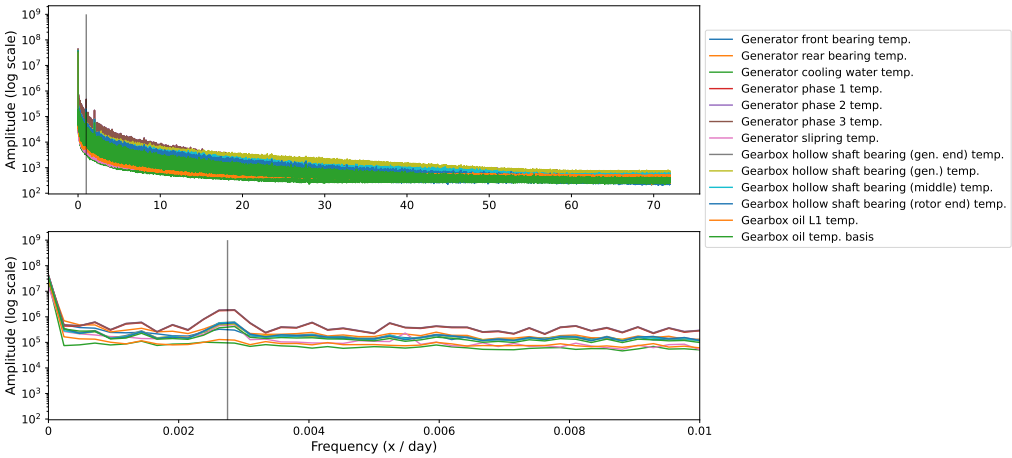


Figure 2.10: Average frequency spectrum for WF2. The top plot shows the amplitudes for all calculated frequencies. The black vertical line indicates the amplitudes for the 1 / day frequency. The bottom plot shows a detailed view of the amplitudes for the frequencies lower than 1 / day. The black vertical line indicates an amplitude peak that corresponds more or less to the 1 / year frequency.

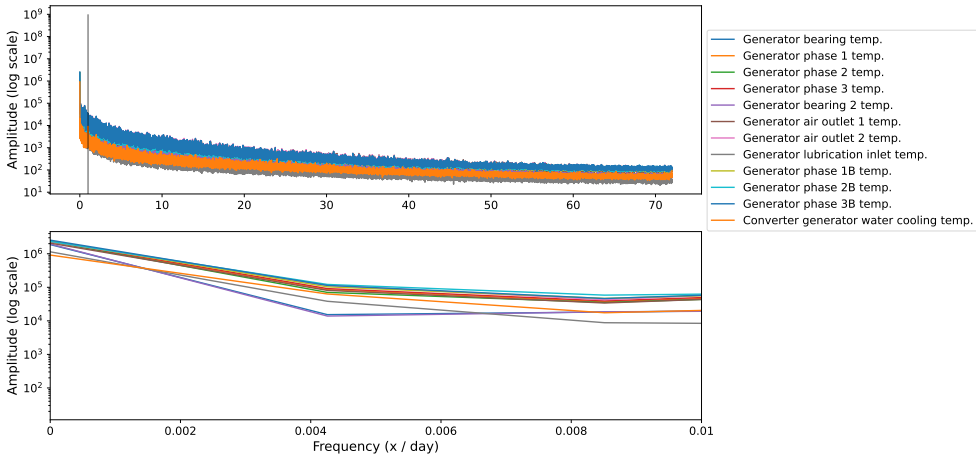


Figure 2.11: Average frequency spectrum for WF3. The top plot shows the amplitudes for all calculated frequencies. The black vertical line indicates the amplitudes for the 1 / day frequency. The bottom plot shows a detailed view of the amplitudes for the frequencies lower than 1 / day. The black vertical line indicates an amplitude peak that corresponds more or less to the 1 / year frequency.

2 Overview of the characteristics of the failure cases

In this section, a detailed analysis is given of the different failure types that will be used for the validation of the methodology. This should clarify the nature of the failures and why it is not always straightforward to detect them. However, this can not be done without first introducing the structure of the wind turbine and more specifically the drivetrain. Failures are caused by specific components. However, the way the failure or the damage that will result in the failure is visible in the data depends on several physical factors, e.g. the proximity of the damaged component to other components, or the fact that the component is in the same cooling circuit as other components.

A wind turbine is a complex machine composed of many different components. This means that it typically also has many different failure modes. Some failures are a minor nuisance resulting in only a short downtime of the wind turbine, while others are much more problematic. Some failure modes can result in downtimes of more than a month and require the replacement of expensive components (e.g. generator, gearbox, ...). Chapter 1 (Introduction) contains an overview of the prevalence of different failure modes.

In the research presented in this thesis, the focus will be on a subset of failure modes. These have been selected by the wind turbine operators because they are major failures that require expensive replacements. The frequent occurrence of these failures was the main motivation of the wind turbine operators to collaborate on this research. Several of these failure modes are related to mechanical issues for which temperature analysis is well suited. One failure mode is related to an electrical failure which is normally impossible to detect using temperature analysis. However, this failure is caused by a mechanical problem which we hypothesize to be visible in the temperature signals. For this reason, it has been included. Table 2.7 gives an overview of the different failure modes or types for each wind farm, together with the prediction challenges.

2.1 The failure modes of WF1

The wind turbines in WF1 experienced several serious generator failures. More specifically, several generators had to be replaced because they were short-circuited. This was the consequence of a problem with the 9.5 MW wind turbine generator. This problem causes damage to the generator, which eventually results in a short-circuit.

The short-circuit is not easy to detect before the actual event since it is a sudden event. However, the wind turbine operators hypothesized that the damage caused by the problem can under certain conditions result in damage to the generator phase coating. This can cause overheating of the generator phases, which should be visible in the temperature

Farm	Component	Mode/Type	Challenges
WF1	Generator	Short-circuit (electrical)	- Only indirectly detectable. - Damaged coating always related to failure? - Other influences generator phase temp.? - Few examples.
	Gearbox	Bearing failure (mechanical)	- Unclear which bearing failed. - Only temperatures for planetary stages. - Few examples.
WF2	Generator	Bearing failure (mechanical)	- Front vs. rear generator bearing failure? - Bearing slip masking trends. - Unclear which replacements are failures.
WF3	Generator	Short-circuit (electrical)	- Same challenges as WF1. - No clear failure examples.

Table 2.7: Table with an overview of the failure modes/types for each wind farm.

signals. It is also likely that the damage starts forming sometime before the short-circuit. This makes it probably detectable using 10-minute SCADA data.

There are however several challenges in detecting this failure mode. Firstly, it can only be detected indirectly if the generator phase coating gets damaged. It is unknown how often this is the case. Secondly, it is unclear whether damage to the coating is always a precursor to short-circuits. Other events that are not at all related to this failure mode might also damage it.

Thirdly, other factors not related to the failure mode can overheat the generator phases. For example, reactive power can have an impact. This means that it is likely that abnormally high temperatures in the generator phases are found that are not related to the failure mode. This will make it more difficult to determine the false positive rate of the methodology. Fourthly, the failure mode occurred only 6 times during the observation period, which means that there are few examples to work with.

WF1 also experienced several gearbox bearing failures. Although this failure mode is a more standard mechanical problem for which temperature signal analysis might be useful, the detection of it was still hampered by several challenges. Firstly, the number of examples is relatively small, i.e. 6 during the observation period. Secondly, it is unknown which gearbox bearing exactly failed. It is unlikely that all 6 failures were caused by the failure of the same bearing. Thirdly, there is only temperature information available for several of the planetary stage bearings, but not for the bearings of the other stages.

2.2 The failure modes of WF2

WF2 is a large wind farm for which over 10 years of data is available. During the observation period, the wind turbines suffered from many generator-bearing failures. The advantage of this failure mode is that it is a typical example of a type of failure that should be detectable using temperatures from these bearings. An added advantage is the number of times the failure occurred. More than 100 bearing replacements have been reported. However, a complication is that a replacement is not necessarily a failure (see further). A distinction is made between failures of the front and rear generator bearings. This information has been made available by the wind turbine operator.

There are however some challenges. Firstly, distinguishing between a failure of the front and rear generator bearing is non-trivial because they are part of the same cooling circuit. Secondly, under certain conditions, the generators suffered from severe bearing slip. Thorough data analysis indicated that the problem mostly occurred at the front generator bearings. The bearing slip showed itself as sudden large spikes in the bearing temperatures. This phenomenon is widespread and will eventually result in the failure of the bearing.

The widespread appearance of the problem has also some unfortunate consequences. It tends to mask or hide the slowly deviating pattern in the front generator bearing temperature that can normally be found just before a failure. This complicates the prediction of front generator bearing failures. Nevertheless, issues in the front generator bearing can to a certain extent also be seen in the rear generator bearing temperatures. Because of this, it is still possible to predict some of the front generator bearing bearing failures.

Thirdly, the information provided by the wind turbine operator lists all the generator bearing replacements. A replacement is unfortunately not necessarily a failure. Due to the frequent occurrence of the problem, occasionally replacement campaigns were done. During such a campaign bearings that were damaged (but had not failed yet) were replaced. A damaged bearing will likely have a less pronounced temperature deviation than a bearing that is about to fail.

During this research, different methods were tried to clean the replacements. The first method is the result of discussions with the wind turbine operator. Preventive maintenance can be distinguished from failures by looking at the downtime of the wind turbine. In general preventive maintenance can be done in one work day. If the wind turbine is offline for a longer period, then it is likely that it is a real failure. This methodology is unfortunately not 100% accurate but should clean the replacements to some extent. It is used by the first methodology in Chapter 4.

The second method removes all replacements if they occur less than two years after the previous replacement of the same bearing, or if the replacement happens at the same

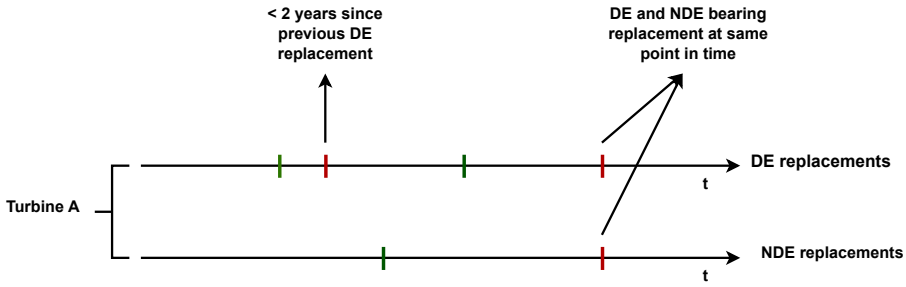


Figure 2.12: Schematic representation of replacement cleaning. Red bars are replacements that are not classified as failures. Green bars are replacements that are classified as failures.

time as the replacement of the other bearing (e.g. NDE bearing is replaced at the same time as the DE bearing, or vice versa) (see Figure 2.12). After cleaning, 55% of the replacements are considered failures, which is sufficient for the analysis. Unfortunately, it likely removes a substantial amount of true failures, and probably misses also some preventive maintenances. This method is used by the second methodology in Chapter 4.

The third method uses a rule-based classifier that categorizes replacements as preventive maintenance, detected failure, or failure. A replacement is classified as preventive maintenance if, during the period preceding the replacement, the wind turbine was not stopped or derated. It is classified as a detected failure if during the period preceding the replacement the wind turbine was derated. It is classified as a failure if the wind turbine was stopped (not due to weather) during a prolonged period of time before the failure. This results in 37 failures that can be used for the validation. This is less than a quarter of the original number. This method is quite strict. It likely removes many true failures. However, it probably results in the cleanest dataset. This method is used in Chapter 3 for the validation of the anomaly detection algorithms.

None of the methods are perfect, however, they do succeed in removing a large part of the preventive maintenances. Some of the methods are more strict than others, which results in the removal of more replacements. There is a trade-off between finding the cleanest dataset, and not removing too many true failures. Understanding which technique performs the best will be the topic of further research.

2.3 The failure modes of WF3

WF3 consists of wind turbines of the same type as those of WF1, i.e. the 9.5 MW wind turbine. The wind turbine operator assumes, although this is not completely certain,

that they suffer from the same generator short-circuit problem as the wind turbines of WF1. During the observation period, the wind farm did not experience any major failures. Information obtained from the wind turbine operator indicated that recently the wind farm experienced two generator failures. It is expected that a thorough inspection of the generators will show that the failures were caused by the same problem as in WF1. However, at the time of writing this thesis, this was not completely certain yet. The short observation period for this wind farm makes it difficult to use it for actual failure predictions. WF3 will mostly be used to see how well the NBM model models the normal behavior.

2.4 Forced outages

Forced outages are defined in IEC 61400-26-1 as a state where the wind turbine experienced damage, a failure, or an alarm that disables the wind turbine. This problem can be detected manually or automatically. In other words, a forced outage or shutdown is triggered when something serious has gone wrong that requires attention. The failures discussed above, e.g. generator and gearbox failures, all result in forced outages.

Next to the above-mentioned failures, also other failures can result in a forced shutdown. Examples that can be found in the WF1 status logs are problems with the converter, peak loads on the blades, problems with the yaw system, problems with generator fans, problems with PT100 sensors, ... The forced outage types for WF3 are similar to those of WF1 since they are wind turbines of the same type (i.e. 9.5 MW wind turbine). In the status logs of WF2, forced outages can also be found. The causes of the forced outages are similar to those of the other wind farms.

Compared to the generator and gearbox failures listed by the wind turbine operator, the forced outages appear much more frequently. This has advantages when training models. Forced outages will be used by the second methodology in Chapter 4.

3 Wind turbine ontologies as a way to standardize the wind turbine data

In this section, several wind turbine ontologies are presented that are used to standardize the wind turbine data. The input data, e.g. SCADA data and status logs, is only to a limited extent standardized when it is delivered by the wind turbine operators. The SCADA data is formatted as a dataframe with rows (observations) and columns (signals) (see Table 2.1). The status logs are formatted as event logs (see Table 2.2).

A major challenge is however that the signals in the SCADA data are often different depending on the wind turbine type. This has several reasons. Firstly, the components of

the wind turbines can be different. For example, there are wind turbines with and without a gearbox (the latter is also called direct-drive). Secondly, some wind turbines have more sensors installed than others. The number of installed sensors has increased over the years. This means that newer wind turbines have more sensors (and signals) than older versions. Thirdly, there is no convention on how the signals should be named in the data that comes from the wind turbine. This means that different wind turbine constructors and operators can name the same or similar signals differently.

A consequence of this lack of standardization is that the data becomes more difficult to interpret. It is also more difficult to train AI models on data from multiple wind farms and wind turbine types. It also complicates failure diagnosis, since this requires understanding the relation between the signals, status codes, and the wind turbine components. To solve these problems, standardization of the data is applied in this thesis. This is done through the development of several ontologies, which will be used by the second methodology in Chapter 4.

This section is built up as follows. Firstly, it is defined what an ontology is. Secondly, several use cases in wind energy research are discussed. Thirdly, it is discussed why ontologies are useful for our research case. Fourthly, the different wind turbine ontologies are presented.

3.1 Defining ontology

Defining what an ontology is, is not a trivial task. There is currently no single definition that is unanimously accepted. Definition 3.1 which was first presented in Gruber [1993] is often used as a basis in the *applied ontology* literature. This definition might need some explanation. The term *conceptualization* refers to an abstract view or model of the world, i.e. the types of objects, concepts, and other entities that are assumed to exist in a domain of interest and their associated properties and relationships. An ontology is then a specific and concrete representation of that conceptualization. This makes it possible that the underlying assumptions about the domain are made explicit. This facilitates their communication and processing [Marykovskiy et al., 2024].

Definition 3.1 (Gruber [1993])**.** An ontology is a specification of a conceptualization.

The problem is that this definition is not deemed sufficient by everyone. Keet [2020] points out that this definition has been criticized from a philosophical standpoint because it introduces new vague concepts, i.e. specification and conceptualization. This has led to the formulation of new definitions. Definition 3.2 was first formulated in Studer et al. [1998]. This definition has also been criticized for its introduction of vague or poorly defined concepts, i.e. *conceptualization*, *formal, explicit specification*, and *shared*. A third definition was presented in Guarino [1998]. Also, this definition is not fully accepted.

Definition 3.2 (Studer et al. [1998])**.** An ontology is a formal, explicit specification of a shared conceptualization.

Definition 3.3 (Guarino [1998] and parafrazed by Keet [2020])**.** An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, i.e. its ontological commitment to a particular conceptualization of the world. The intended models of a logical language using such a vocabulary are constrained by its ontological commitment. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating these intended models.

The definitions used in the computer and information science context are typically more pragmatic and practical. This means that even though they are in some cases inspired by the philosophical definitions, they are more focused on the application. For this reason, they are often categorized as *applied ontologies* or *computational ontologies*. Definition 3.4 formulated in Gruber [2009] is relevant in this context. Other definitions have been presented in papers. However, the goal of this part of the thesis is not to give an exhaustive discussion of all the different definitions, their weaknesses and strengths, and philosophical problems. It suffices to say that defining the concept *ontology* is difficult and that in computer and information science applications Definition 3.1 is used often.

Definition 3.4 (Computational ontology according to Gruber [2009])**.** In the context of computer and information sciences, an ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application. ... Due to their independence from lower-level data models, ontologies are used for integrating heterogeneous databases, enabling interoperability among disparate systems, and specifying interfaces to independent, knowledge-based services.

3.2 Applications of ontologies

There are several situations in which ontologies have an added value. They were first proposed to solve problems with data integration. In Keet [2020] ontology-driven schema-based data integration and data-based data integration are discussed. The former is used in situations where different legacy systems need to be integrated. Figure 2.13 gives a schematic representation. The figure shows two (relational) databases and an object-oriented (OO) C++ application (Implementation layer). These different systems need to be integrated.

The conceptual model layer shows the content and structure of the data. The two databases and the OO-application contain the same information, it is however represented differently in the conceptual model layer (the data from the first database is represented in an EER diagram, the data from the second database in an ORM and the data from the C++ application as a UML Class Diagram). The ontology layer is used to combine or integrate the different conceptual models. It gives a shared common vocabulary, which matches the similar concepts of the different databases and the application [Keet, 2020].

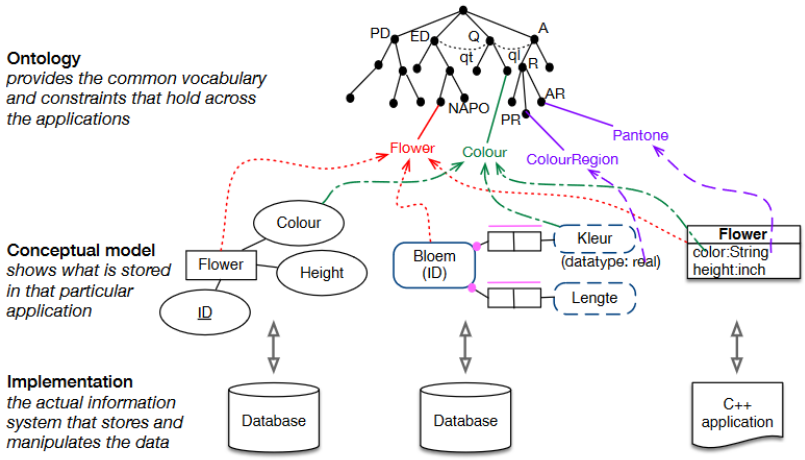


Figure 2.13: Schematic representation of an ontology-based schema-based data integration scenario. The implementation layer shows several different implementations, i.e. two relational databases and one object-oriented software implementation. The conceptual model layer shows three different conceptual models, i.e. an EER (left), an ORM (center), and a UML Class Diagram (right) [Keet, 2020].

The ontology-based data-level integration works differently. It was developed by researchers from the molecular biology field. They searched for interoperability at the instance level, in the situation where there are multiple databases spread over the internet. This was achieved using lightweight ontologies or structured controlled vocabularies (a controlled vocabulary is a set of terms or phrases that have been pre-selected for use in a particular domain or context [Marykovskiy et al., 2024]) [Keet, 2020].

Figure 2.14 gives an example with two databases, i.e. the KEGG and the Interpro databases. From each database, one entry is shown. The entries have an attribute or field that refers to the Gene ontology. This makes it possible to match these entries,

even though they are stored in different databases, have identifiers from different identifier schemes, and have different attributes [Keet, 2020].

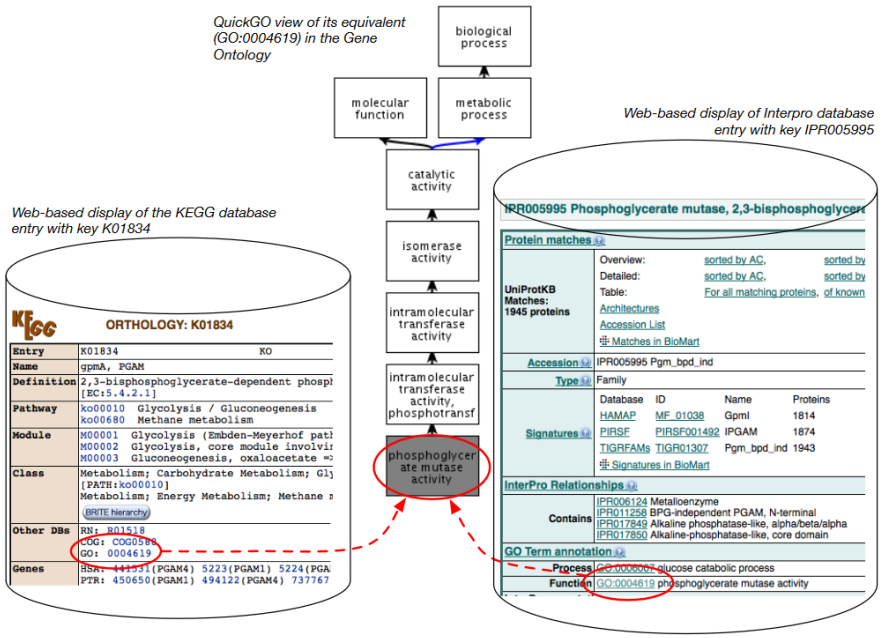


Figure 2.14: Schematic representation of an ontology-based data-level integration [Keet, 2020].

According to Keet [2020] ontologies have also been useful for other application scenarios, e.g. e-learning, information retrieval, management of digital libraries, improving the accuracy of question-answering systems, and many more. The biggest success was achieved however with the Gene Ontology. Much more can be said about the applications and the success stories. It would however lead us too far.

3.3 Ontologies in wind energy research

Ontologies have been developed for applications and research related to wind turbines. An example is OntoWind presented in Küçük and Küçük [2018], which is a domain ontology specifically for wind energy. In Papadopoulos and Cipcigan [2009a] a simple ontology

model is presented for condition monitoring of wind turbines. The ontology is used there to link failure symptoms to sensor readings.

Ontologies also play an important role in making the data from wind turbines more accessible (digital transformation of the wind energy sector). Knowledge engineering and representation are important elements in this. Ontologies are used to formalize these representations. This will be the basis for the development of Knowledge-Based Systems and new types of Digital Twins [Marykovskiy et al., 2024].

3.4 Making sense of temperature signals, status codes, and failure comments using ontologies

Two ontologies are developed in this thesis, i.e. a temperature signal ontology, and a status code ontology. The temperature signal ontology shown in Table 2.8 links temperature signals to several main components, i.e. generator, gearbox, main bearing, and sub-components, i.e. slipring, cooling, phases, bearings, lubrication, for the generator, and bearings, hollow shaft, oil for the gearbox. The main bearing has no sub-components in this ontology. This ontology is constructed for all three wind farms, i.e. WF1, WF2, and WF3.

The status code ontology links the status codes to the main components. Due to the large number of unique status codes the linking is done in a semi-automated fashion. A status code is connected to a system if it is associated with a stop or warning and it contains a certain substring in its name. Conflicts where a status code is assigned to two systems are resolved manually using expert knowledge. Table 2.9 gives an overview of how the status codes are connected to the different systems.

The status codes are only linked to components, and not to sub-components. This is because there is too much uncertainty on how the status codes are related to sub-components. To avoid the data being polluted with wrongly connected status codes, only connections with components are made. The number of components in the status code ontology is however larger than in the temperature signal ontology. This is because most temperature signals are related to the generator, gearbox, and main bearing. There are only a few temperature signals linked to the other components. The status code ontology has so far only been built for WF1. Future work will expand it to other wind farms.

The ontologies presented here are limited in complexity and scale. Future research will integrate these ontologies with other ontologies, e.g. OntoWind, ..., and/or international wind turbine standards, e.g. Reference Designation System for Power Plants (RDS-PP),... The addition of a failure ontology that maps patterns of events (e.g. temperature anomalies, status codes, ...) to failure types is also a possibility.

Component	**Sub-component**	**Wind farm**		
		WF1	**WF2**	**WF3**
Generator	Slipring		24	
	Cooling	1	20	36, 37, 42
	Phases	5, 6, 7, 8, 9, 10, 11, 12, 13	21, 22, 23	32, 33, 34, 39, 40, 41
	Bearings	3, 4	18, 19	31, 35
	Lubrication			38
Gearbox	Bearings	14, 15, 16, 17		
	Hollow shaft		25, 26, 27, 28	
	Oil		29, 30	
Main bearing		2		

Table 2.8: Table containing the temperature signal ontology. The temperature signals for WF1, WF2, and WF3 are represented by their signal IDs, which can be found in Tables 2.4, 2.5, and 2.6.

Component	**Status codes**
Gearbox	All status codes associated with stops/warnings/forced shutdowns with substring "Gear" in name.
Generator	All status codes associated with stops/warnings/forced shutdowns with substring "Gen" in name.
Converter	All status codes associated with stops/warnings/forced shutdowns with substrings "MSI" or "LSI" or "Chopper" in name.
Pitch	All status codes associated with stops/warnings/forced shutdowns with substring "Pitch" in name.
Blade	All status codes associated with stops/warnings/forced shutdowns with substring "Blade" in name.
Yaw	All status codes associated with stops/warnings/forced shutdowns with substring "Yaw" in name.
Brake	All status codes associated with stops/warnings/forced shutdowns with substring "Brake" in name.
Rotor	All status codes associated with stops/warnings/forced shutdowns with substring "Rotor" in name.
Water cooling	All status codes associated with stops/warnings/forced shutdiwns with substring "WaterCool" in name.

Table 2.9: Status code ontology for WF1.

4 Conclusion

Overall it can be stated that the data is of good quality, and sufficient data is available. This should facilitate the validation of the methodologies. However, it can not be ignored that there are several important challenges.

Firstly, there is strong evidence for seasonal (see frequency spectrum plots) and daily fluctuations (see frequency spectrum and autocorrelation plots). This must be taken into account during the modeling or it might result in unreliable anomaly detections. Secondly, the failure modes that are available for validation contain several complications. For the known failure modes, there are only a few examples available.

Furthermore, there are also failure modes that remain unknown, which might influence the temperatures of the components. These will then be identified as anomalous by the anomaly detector (see Chapter 3). This means that an extra step will be required to distinguish the different failure modes. This can only be done by taking into account the anomaly patterns in different signals and extra information (e.g. status codes, ...) (see Chapter 4).

The problem with certain unknown failure modes is that they might influence the temperatures in such a way that they mask the anomaly pattern associated with the failure mode of interest. This will result in a lower detection rate by the model. We hypothesize that it might still be possible to detect the failure mode indirectly in a different way.

The fact that unknown failure modes can generate anomalies in the same signals that are used for the identification of the failure modes of interest, means that a naive calculation of the accuracy based on confusion matrices is not ideal. New strategies will need to be developed to give a more correct estimation of the accuracy. It is unavoidable that to a certain extent visual inspection of the results and subjective judgment will be required.

3 | Failure prediction for wind turbines using 10-minute SCADA data

The first block of the pipeline (for the complete pipeline see Figure 1.24) focuses on failure prediction using temperature signals (e.g. gearbox and generator bearing temperatures, generator phases, ...), and wind turbine operation signals (e.g. active power, nacelle temperature, rotor speed, ...), as input. Failure prediction for wind turbines is not a trivial task. It is currently a hot research topic that has not been solved so far. Many papers have been and are written that test different anomaly detection techniques on SCADA and status log data. Nevertheless, a final verdict on the best way to identify anomalies and predict failures has not been reached.

During the research which is presented in this thesis, several methodologies have been tested, e.g. purely statistical techniques, shallow machine learning, and deep learning. One technique has been selected as the best one to generate the output that is used as input by the second block of the pipeline (see Figure 1.24).

The goal was to find the least complex methodology that can accomplish the requested task, namely predicting the failures of different wind turbine components using 10-minute SCADA data. The emphasis on the method complexity comes from the fact that this research works towards a solution that can eventually be deployed in the field. This means that it needs to comply with restrictions defined by the operators. These restrictions

are ease of use and interpretation, low computational cost, low complexity, and low data restrictions (more detailed information on these restrictions can be found under the Research objectives in Chapter 1).

The investigation started with the least complex model that might be used to predict failures. Gradually, the complexity of the models was increased, until models were found that achieved sufficiently high performance. By using this model selection scheme, the hope was that the least complex methodology would be found to solve the problem.

The least complex method that was tested is the wind farm median for a specific signal. The median is used as an approximation of the normal behavior of the wind turbines given certain environmental conditions. This technique has several advantages. Firstly, it does not require an explicit model, meaning it does not need predictor selection or training, ... Furthermore, the model does not depend on assumptions by the researcher that may be based on imperfect knowledge. Secondly, it has a low computational demand. Thirdly, it has low data requirements because it does not need to be trained. The disadvantage is however that it is not able to model behavior that is specific to a wind turbine, i.e. a cool-down due to a shutdown of the wind turbine that does not coincide with a general shutdown of the wind farm. This model is used as a benchmark for the other methodologies.

A second, more complex methodology is based on shallow machine learning techniques, e.g. elastic net, random forest, gradient boosting, ... for the modeling of the normal behavior of the temperatures. Shallow machine learning techniques require in general substantially more data and are also more computationally demanding, but they can model non-linear relations better. Several configurations of these models were tested. One configuration resulted in an ensemble of models in which the results of several machine learning algorithms are combined in an attempt to get better predictions.

A third methodology is based on deep learning models, i.e. autoencoders. These models are even more capable of modeling non-linear relations. In principle, they require more data than the shallow machine learning models, and they are more computationally demanding.

This chapter is built up as follows. The first section explains the background of this research, e.g. explaining the different parts of the NBM framework and situating it in the current state-of-the-art research. The focus will mainly be on research that is based on the NBM principle. This keeps the background discussion as relevant as possible. The second section discusses the preprocessing of the data. The input data has often several deficiencies that need to be corrected before any modeling can be done. Several of these deficiencies and their solutions are discussed.

The third section discusses several methodologies that have been tested in the course of the research, e.g. a wind farm median NBM, a shallow ML pipeline, and an autoencoder pipeline. The results of these methodologies are discussed and compared in the fourth section. The fifth section formulates several concluding remarks concerning the NBM

modeling process and several observations that were made in the course of the investigation of the wind turbine data. The sixth section discusses two statistics-based anomaly detection algorithms, i.e. CUSUM and MAD-IOD, that are used to analyze the prediction error of the autoencoder NBM. The results are discussed and compared.

1 Background

Anomaly detection and failure prediction on industrial machine signals and more specifically wind turbines is a much-researched topic. This is due to its high economic relevance and the fact that over time more sensor data has become available [Helsen, 2021]. Unexpected component failures increase the operational and maintenance costs of the machines. This is especially true for wind turbines where large maintenance works are often slowed down by the remoteness of the location (e.g. offshore wind turbines, mountainous terrain, ...), and the infrastructure required for handling the large components (e.g. generator, gearbox, blades, ...).

Deducing component state or health from temperature or vibration signals is unfortunately not a trivial task. In the case of wind turbines, the analysis is complicated by the often changing operational states, sensor imprecisions, sensor measurement errors and failures, large variability in wind turbine types, ... The unsolved and complicated nature of this problem means that currently a large amount of research focuses on this topic. Experiments are done with many different algorithms. Their ability to identify anomalies is tested.

Different methodological families compete in this domain. According to Helbing and Ritter [2018], they can be divided into model-based signal processing and data-driven methods. An alternative classification can be found in Black et al. [2021], where a distinction is made between 1) trending, 2) clustering, 3) NBM, 4) damage modeling, 5) alarm assessment, and 6) performance monitoring. In Tautz-Weinert and Watson [2017] five categories are identified: 1) trending, 2) clustering, 3) NBM, 4) damage modeling, and 5) assessment of alarms and expert systems.

An extensive overview of all these different techniques would go too far. Instead, the main focus will be on the NBM methodology, which is the basis of the research in this thesis. NBM models have shown their merits in the past. Furthermore, the advantage of having a method that can model the normal expected behavior of a wind turbine, is that it can be used for generating new normal data, which is useful in certain contexts where artificial data is required. Also, if easily interpretable models are used, the predictions can be interpreted and a better understanding can be achieved of the relation between the different signals.

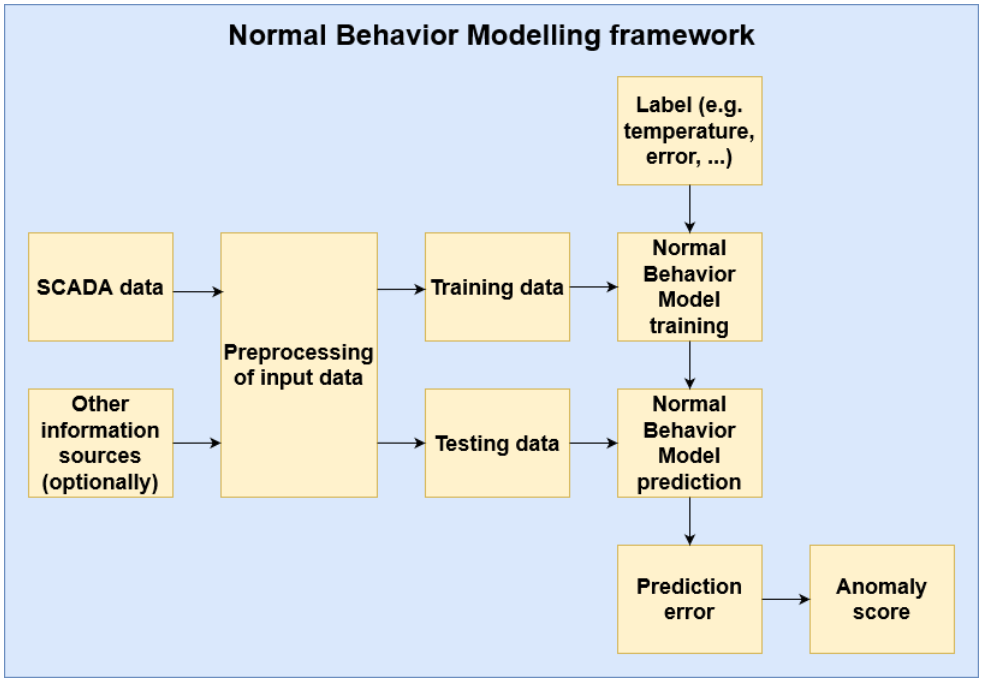


Figure 3.1: Schematic overview of NBM framework.

The NBM family is very diverse. Many different algorithms can be used to model the normal or healthy behavior of a wind turbine signal. However, in all this diversity, there are several commonalities. Figure 3.1 gives an overview of a standard NBM flow. In what follows the different steps are discussed in more detail.

Splitting the data in training and testing datasets Splitting the data is done before the modeling step to avoid information leaks. The test dataset is a random subsample that is set aside for the final validation of the methodology and should not be used during training. How this split is made depends on the type and the amount of data. Often 80-20% or 70-30% random splits are made. However, other options are possible.

If the data are time series, which is the case when using SCADA data, and the NBM models use lagged predictors, or the use case requires that the data is kept in chronological order, then the train-test split should be done more carefully. A possible way is assigning the first x% of the observations (based on their timestamp) to the training dataset, and the rest to the testing dataset. This method is used in this thesis.

Preprocessing of the data Preprocessing is done to clean the signals (e.g. removing measurement errors, filling in missing values, ...) and in some cases to reduce the noisiness of the signals (e.g. binning, ...). Filtering is sometimes used if it is expected that the relation between the signals is influenced by certain other factors (e.g. wind turbine states, ...). The preprocessing is done on the training and testing dataset separately. However, the same techniques are used on both datasets. If thresholds are required, they are determined or tuned only on the training dataset. The optimal values are then used on the test dataset.

Training of the NBM model Training the NBM model is done only on the training dataset. In this research, anomalies in temperature signals are used as indicators for the health of the wind turbine components. The normal or healthy behavior of the component temperatures is modeled by an NBM. The supervised machine learning algorithms are suitable for this task. Examples are ordinary least squares (OLS), random forest (RF), support vector machine (SVM), feedforward neural network (FNN), long short-term memory (LSTM), ...

The NBM model is trained on healthy data (meaning not polluted with anomalies that can be associated with a failure). Once the NBM model has been trained it can be used for predicting the expected normal behavior on the test dataset. The target that is used during the modeling process is the temperature signal of which the normal behavior needs to be modeled. The predictors can be other component temperature signals, wind turbine operational signals, and environmental signals.

Analysis of the prediction error In this step the difference between the predicted and the observed behavior (prediction error) is analyzed. Unusual trends in this difference can be considered indicators of a problem. The deviation is then transformed into an anomaly score that says something meaningful about for example the probability of failure or the remaining useful life (RUL).

1.1 The current state of the art

In what follows an overview is given of different techniques that are used in the state-of-the-art literature for each step of the NBM pipeline (see Figure 3.1). The papers that will be discussed in this section have the following properties: firstly, they are based on wind turbine SCADA data, secondly, they perform condition monitoring and anomaly detection on temperature signals of the wind turbine (this excludes, for example, research that focuses on the power curve), and thirdly, they follow the NBM methodology. By limiting the scope of the overview, it can be more exhaustive, and give the reader a better insight into what has been tried in the literature.

Overview of the preprocessing techniques found in the literature

Preprocessing is an important, although often somewhat underexposed, part of the NBM pipeline. Decisions taken during this step can influence the training and performance of the NBM models later on. Different preprocessing techniques exist and have been used in recent research. The choice of a technique is to a certain extent guided by the properties of the input data. E.g. for time series data the order of the data points, and the relation between them, is relevant. This means that only preprocessing techniques that retain this property of the data should be used. But even then multiple preprocessing techniques are usable. Why a certain technique is chosen over a different one is often not thoroughly explained in papers.

This section attempts to give an overview of which techniques are used in current state-of-the-art research. An analysis of the literature shows that the preprocessing of the data is used for among other things the handling of missing values, outliers, noise reduction, filtering, and transforming the data.

Missing values can be problematic for certain statistical and machine learning models. For this reason, they need to be treated/filled in properly. Several techniques are used in the literature. The first technique is removing the observations with missing data (see Maron et al. [2022], Miele et al. [2022], Cui et al. [2018] and Bangalore et al. [2017]). This can be difficult when time series modeling is used. Furthermore, the question needs to be asked why the data is missing. If it is not MCAR this can result in bias [Emmanuel et al., 2021].

A different solution is single imputation. A first example of this is carry forward and/or backward. In this technique, the missing value is replaced by the last known value preceding the missing value (carry forward) or the first known value following on the missing value (carry backward). This is used in Bermúdez et al. [2022], Campoverde et al. [2022], Chesterman et al. [2022], Chesterman et al. [2021] and Mazidi et al. [2017]. An alternative is interpolation. This can be done in several ways, e.g. Hermite interpolation (see Bermúdez et al. [2022] and Campoverde et al. [2022]), linear interpolation (see Chesterman et al. [2022], Chesterman et al. [2021] and Miele et al. [2022]),...

Several elements need to be taken into account when using imputation. First of all, large gaps in time series are a problem since the imputation can become meaningless. This can result in pollution of the relation between multiple signals. High dimensionality of the data can also be a problem [Emmanuel et al., 2021].

If the amount of missing data is fairly limited (and it does not contain long stretches of missing data), an aggregate like for example the mean or median can be calculated and used as a proxy. The resulting time series has of course a lower resolution, but the missing values are gone. This method is used in Verma et al. [2022]. The success of this

methodology of course depends on how much data is missing, and on how much data each aggregated value is based on.

Another solution is what is called machine learning-based imputation. This technique uses certain machine learning models to impute the missing values [Emmanuel et al., 2021]. There are several ways to do this. For example, clustering algorithms (e.g. k-nearest neighbors, ...) can be used to find similar complete observations. This is used in Black et al. [2022]. The success of this technique of course depends on the quality of the imputation model. If the model is poor, this results in poor imputations, which will degrade the performance of the anomaly detection model, no matter which modeling technique is chosen.

The above overview gives several techniques that have been used in research that focus on anomaly detection in temperature signals from wind turbines. However, there exist several other techniques that can be used for imputing missing values, e.g. hot-deck imputation, regression imputation, expectation-maximization, and multiple imputation [Emmanuel et al., 2021]. To the best of the authors' knowledge, no (or a very limited number of) papers in the wind turbine condition monitoring research domain have been written that use these techniques. The lack of research that uses, for example, multiple imputation (e.g. multivariate imputation by chained equations (MICE) [van Buren and Groothuis-Oudshoorn, 2011]) can be considered a blind spot. Multiple imputation techniques have several advantages (a.o. less bias) since they combine multiple estimations of the missing values. A drawback is however that they are more complex and require more processing power.

A second problem that might influence the NBM training, is outliers. If these occur in sufficient quantity they can impact the modeling severely. Oftentimes the decision is made in the literature to simply remove them. This of course implies that the outliers can be detected in the first place. The simplest solution is using, if these are available, expert-knowledge-based thresholds. They require no statistical model, and if they are tied to real physical knowledge or knowledge of the inner workings of the machine, then they can be very useful. However, this information is not always available or is in some cases difficult to collect.

Possible alternatives are data-driven methods. This can be done for example by using the interquartile range (see Campoverde et al. [2022]), or a custom dynamic or user-defined threshold (see Chesterman et al. [2022], Chesterman et al. [2021] and Castellani et al. [2021]), the 5-sigma rule (see Miele et al. [2022]) or clustering (see Cui et al. [2018] and Bangalore et al. [2017]). Removing outliers needs to be done carefully to avoid abnormal values associated with the failure of interest being removed. This requires a good understanding of the data. In some cases, the outliers will be clearly visible. This can for example be the case when measurement errors result in values that are multiple

times larger or smaller than what is physically possible. In these cases removing the outliers is straightforward. However, in other cases, the difference between outliers caused by the failing component and outliers caused by another reason is much less clear. Removing these types of outliers should only be done after careful consideration.

Some papers also use noise reduction techniques. Noise on signals can make it more difficult for the NBM algorithm to model the relation between them. If it is possible to clean the signal this should be considered, since it will improve the performance of the NBM model. This can be done for example by aggregating the data to a lower resolution (see Chesterman et al. [2022], Chesterman et al. [2021] and Turnbull et al. [2021]), or cleaning or filtering the data using expert knowledge (see Peter et al. [2022], Takanashi et al. [2022], Verma et al. [2022], Turnbull et al. [2021], Udo and Yar [2021], Beretta et al. [2020] and Kusiak and Li [2011]).

In some cases, it might be useful to transform the data. This can result in features with more favorable properties. For example, the principal component analysis (PCA) transformation (see Campoverde et al. [2022] and Castellani et al. [2021]) and Zero-Phase Component transformation (see Renström et al. [2020]) result in uncorrelated features which are linear combinations of the original signals. This can be beneficial for the training of the NBM.

Another transformation that might be done is rebalancing the dataset. This can be necessary when certain operational states of the wind turbine are underrepresented in the training data. Oversampling of the minority class or undersampling of the majority class is an option. A more sophisticated technique is the synthetic minority oversampling technique (SMOTE) used in for example Verma et al. [2022] and Liu et al. [2023]. Lastly, in some papers, new features are created by clustering the original signals of the SCADA data in several groups using clustering algorithms. The new features are in the next step used as input to the NBM model (see Liu et al. [2020]).

This overview shows that many different preprocessing techniques are available and have been tried. However, the technique choice is often not well motivated in the papers. Also, the impact of a certain technique on the results is in general not extensively discussed, even though it is known from statistical research that this can be significant.

Overview of the data and wind turbine signals found in the literature

SCADA data can come in different resolutions. The most available resolution is 10 minutes since it reduces the amount of data that needs to be transmitted [Yang et al., 2014]. This means that the dataset contains for each 10-minute window the average signal value. In general, the SCADA data also contains information on the minimum, maximum, and standard deviation of the signal during the 10-minute window. Less common resolutions

are for example 1 sample per minute or second. In the state-of-the-art literature, the following resolutions can be found:

- 10-minutes: Bermúdez et al. [2022], Black et al. [2022], Campoverde et al. [2022], Chesterman et al. [2022], Maron et al. [2022], Mazidi et al. [2017], Miele et al. [2022], Peter et al. [2022], Takanashi et al. [2022], Beretta et al. [2021], Beretta et al. [2020], Castellani et al. [2021], Chen et al. [2021], Chesterman et al. [2021], Meyer [2021], Turnbull et al. [2021], Udo and Yar [2021], Beretta et al. [2020], Liu et al. [2020], McKinnon et al. [2020], Renström et al. [2020], Zhao et al. [2018], Bangalore et al. [2017], Dienst and Beseler [2016], Bangalore and Tjernberg [2015], Bangalore and Tjernberg [2014], Schlechtingen and Santos [2014], Schlechtingen and Santos [2012], Zaher et al. [2009], Garlick et al. [2009].
- 5-minutes: Kusiak and Li [2011].
- 10-seconds: Kusiak and Verma [2012].
- 7-seconds: Liu et al. [2023]
- 1-second: Sun et al. [2016], Li et al. [2014].
- 100 Hz: Verma et al. [2022], Kim et al. [2011].

The 100 Hz data used in for example Verma et al. [2022] comes from the Controls Advanced Research Turbine (CART) of the National Renewable Energy Laboratory (NREL). The fact that it is a research wind turbine makes it possible to sample at much higher rates than what is normally possible.

The SCADA data contains information on many different parts of the wind turbines. This implies that the datasets consist in general of dozens or even hundreds of signals (depending on the wind turbine type). However, not all of them are relevant to the case that is being solved. Some papers focus on a small subset of expert-selected signals (see for example Peter et al. [2022], Bermúdez et al. [2022] and Chesterman et al. [2022], ...). Other papers use a large subset of signals and reduce the dimensionality of the problem during preprocessing or during a model-based automatic feature selection step that extracts the relevant information (see for example Lima et al. [2020], Renström et al. [2020], Dienst and Beseler [2016], ...). Some papers select signals based on the internal structure of the wind turbine (for example in Marti-Puig et al. [2021] on the subsystem level).

The advantage of the first method is that the number of signals used for training is limited, which reduces the computational burden of the training process. The disadvantage is however that for cases in which the expert knowledge is not complete, important signals might be missed. This is less likely when the second method is used. The disadvantage of this method is however that the computational cost is significantly higher and that the selected subset can change over different runs. The third method uses the wind turbine

ontology or taxonomy as a guideline. Its performance depends of course on the quality of the ontology or taxonomy.

The signals that are often used for condition monitoring of wind turbines can be more or less divided into three groups:

1. Environmental data, e.g. wind speed, outside temperature, ... (used in for example Bermúdez et al. [2022], Bermúdez et al. [2022], Black et al. [2022], Campoverde et al. [2022], Mazidi et al. [2017], Miele et al. [2022], ...).
2. Operational data from the wind turbine, e.g. active power, rotor speed, ... (used in for example Bermúdez et al. [2022], Black et al. [2022], Chesterman et al. [2022], Miele et al. [2022], Peter et al. [2022], ...).
3. Wind turbine temperature signals, e.g. the temperatures of the generator bearings, the temperature of the main shaft bearing, the temperature of the generator stator, ... (used for example Bermúdez et al. [2022],Black et al. [2022], Campoverde et al. [2022], Chesterman et al. [2022], Mazidi et al. [2017], ...).

The first two groups are often used by default, independent of the target signal that needs to be modeled or the failure that needs to be detected. This is because they contain information on the wind turbine context (e.g. it is a stormy day, a very hot day, the wind turbine is derated, ...). The third group of signals is much more tied to the case at hand. For example, generator temperature signals are used if the focus lies on generator failures, and gearbox temperature signals are used if gearbox failures need to be detected.

Some papers combine the SCADA data with other information sources, e.g. event logs that contain wind turbine alarms (see Miele et al. [2022], Beretta et al. [2020], Renström et al. [2020] and Kusiak and Li [2011]) or vibration data (see Turnbull et al. [2021]).

2 Data preprocessing steps used in this research

The raw machine data can not be used straightaway as input to the models. Several steps need to be taken to transform and clean it. Which steps and how many depends on the models and the methodology design that is used in the pipeline. Multiple pipelines are discussed in this thesis. These pipelines preprocess the data differently in some aspects. In what follows an overview is given.

2.1 Wind farm data normalization

This step is only done by the shallow ML pipeline, not by the autoencoder pipeline. The wind turbine signals in the SCADA data are quite complex, meaning there are a lot of

factors that influence them. This complexity makes it more difficult to model the normal behavior. This means that if a part of it could be removed it would reduce the complexity of the problem, which normally should result in an improved modeling performance and an overall more data-efficient model. This can be accomplished by calculating the wind farm median of a signal (e.g. temperature of generator bearing 1 at time t for turbines 1, 2, 3, 4, 5) and subtracting it from the wind turbine signals (e.g. temperature of generator bearing 1 at time t of turbine 1). A more detailed explanation of how the wind farm median is calculated can be found in section 3.1. This technique is also used in Chesterman et al. [2022] and Chesterman et al. [2021].

The wind farm median can be seen as an implicit normal behavior model. It is implicit because it does not require the selection of predictors or the training of a model. It models the normal behavior as long as $50\% + 1$ turbines are acting normally at any given time. By taking the median over a whole wind farm, it captures wind farm-wide effects, which are common to all the turbines, which is why it will be called the *common component*. Elements captured by this component are for example the wind speed, the wind direction, and the outside temperature, ... By subtracting the median from the wind turbine signal, these wind farm-wide effects are removed. What is left are wind turbine-specific effects, which will be called the *idiosyncratic component*. Wind turbine-specific anomalies should only be visible in this component.

In practice, this preprocessing step means that from each input signal the wind farm median is subtracted. Figures 3.2 and 3.3 show the results for two signals that have been decomposed into a common and idiosyncratic component. In both cases, certain patterns are much more clear in the idiosyncratic component, compared to the original signal. A substantial amount of the variance in the original signal has disappeared in the idiosyncratic component.

Figure 3.4 shows that the wind farm median is useful for filtering out macro-level or wind farm-wide effects (e.g. seasonal fluctuations, ...) from the data. The common component captures the seasonal fluctuations in the nacelle temperature, which results in an idiosyncratic component that is free of them. Furthermore, it makes the strange jump in the temperatures during year 1 much more clear.

Using the idiosyncratic component instead of the original signal as input to machine learning models has the advantage that less than one year of training data can be used. Seasonal fluctuations have been removed. This means that they will no longer influence the false positive rate. Furthermore, the common component also captures the transient behavior (e.g. cool-downs, ...) that is common to all wind turbines in the wind farm. This means that the idiosyncratic component is free from most transient behavior. What remains of transient behavior are cool-downs that are unique to the wind turbine. This is for example the case when they are caused by a wind turbine that is turned off for

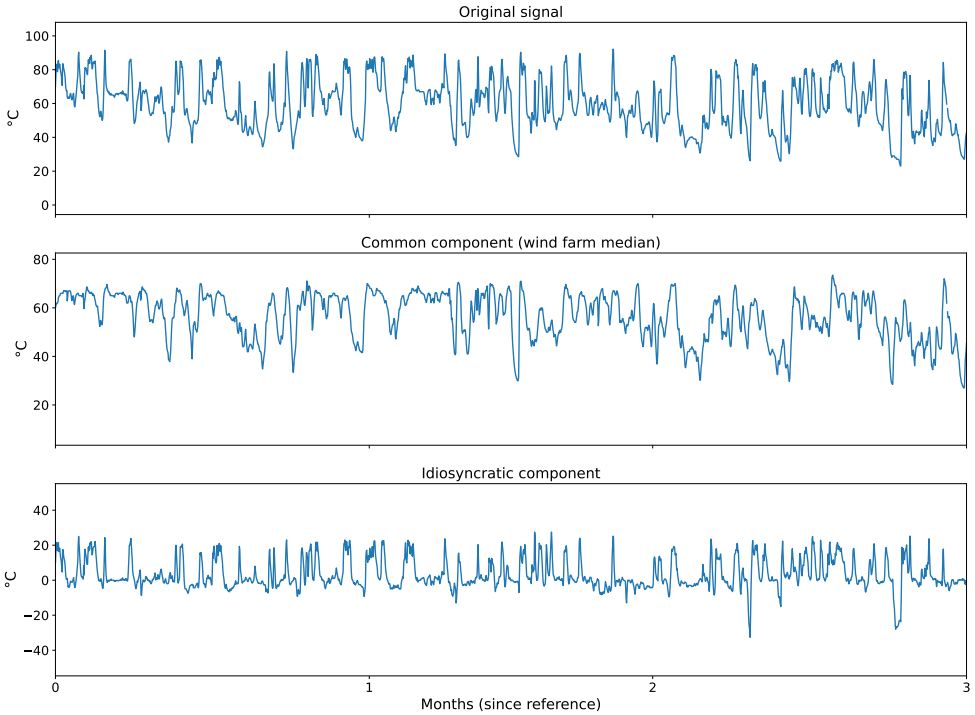


Figure 3.2: Example of a decomposition of a generator rear bearing temperature signal of a wind turbine of WF1 in a common and idiosyncratic component

maintenance. These events are relatively rare, which means that subtracting the wind farm median from the signals has reduced the modeling complexity substantially.

How well the wind farm median removes the macro-level effects depends of course on the quality of the wind farm median. An issue that may arise, especially in small wind farms, is that a substantial amount of wind turbines are offline for maintenance. For example, 2 wind turbines may be offline for maintenance in a wind farm with 4 turbines. This will have an impact on how representative the median is for the normal behavior. For this reason, a rule-based safeguard is added that under certain conditions will convert the wind farm median to NaN. The rules are the following:

- if wind farm size < 5 : No missing values at time t are allowed.
- if $5 \leq$ wind farm size < 10 : At most 20% missing values at time t are allowed.
- if wind farm size ≥ 10 : At most 40% missing values at time t are allowed.

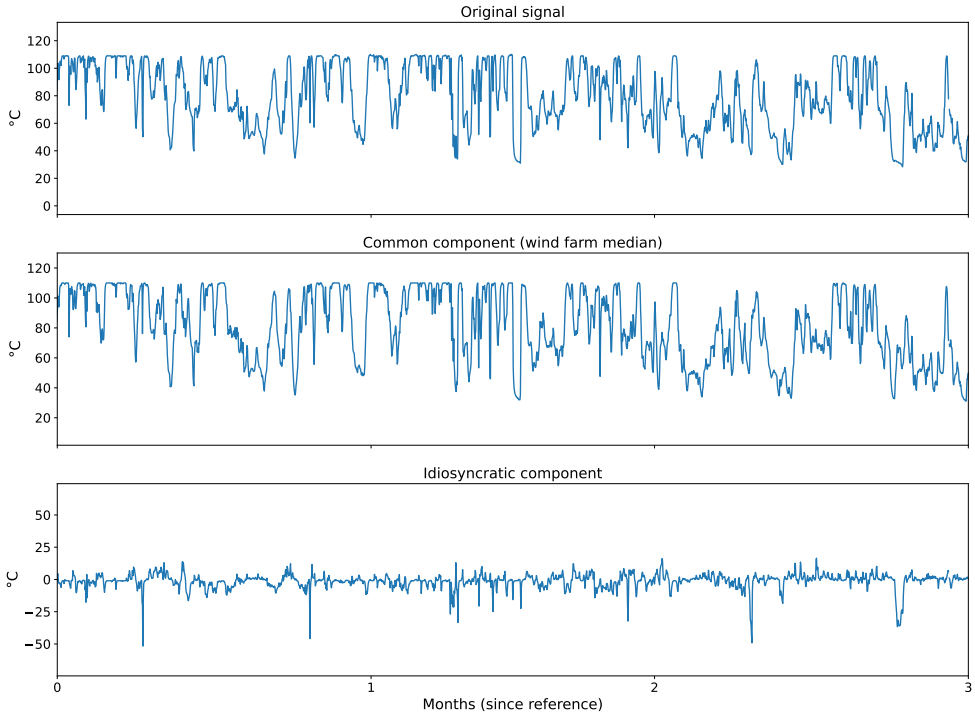


Figure 3.3: Example of a decomposition of a generator phase temperature signal of a wind turbine of WF1 in a common and idiosyncratic component

Subtracting the wind farm signal medians from the signals is not done in the autoencoder pipeline. An analysis of the results indicated that they become less intuitive and more difficult to explain when the wind farm median is subtracted. It makes the pipeline more complicated, while at the same time, it did not appear to improve the results much. However, further research needs to be done on this.

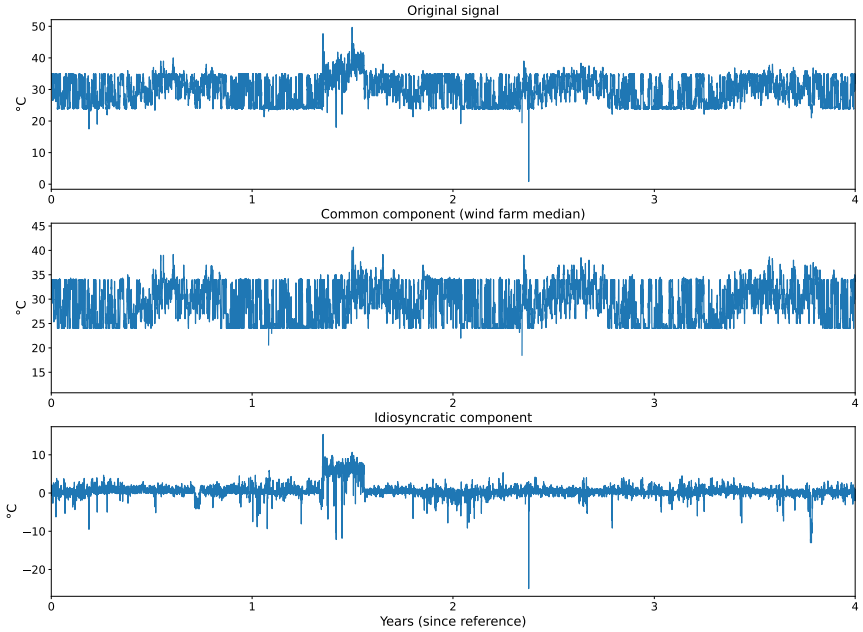


Figure 3.4: Example of a decomposition of a nacelle temperature signal of a wind turbine of WF1 in a common and idiosyncratic component

2.2 Transforming the data into a wind turbine-stacked format

Both the shallow ML and autoencoder pipelines transform the input data into a wind turbine-stacked format (see Table 3.1). This means that the observations of the different wind turbines are stacked as if they would come from a single wind turbine. This transformation is used because the NBM models are trained on data from all wind turbines at once. A side effect is however that potential correlations between different wind turbines are not used during the modeling of the normal behavior. Nevertheless, it was decided that this was the way to go because of several reasons.

Firstly, the correlations between different wind turbines are influenced by external factors, e.g. wind direction, wake effects, and wind farm layout, ... This means that the relation between for example temperature signals of two wind turbines is most likely complex. To model these accurately more complex models will be required. Also, the data demands will increase substantially. This can be an issue when the industry constraints described in the introduction are taken into account. Furthermore, it is unlikely that all the wind

Wind turbine	**Datetime**	**Signal 1**	**Signal 2**	...	**Signal N**
T1	2024-01-01 00:00:00	5.0	6.0	...	8.5
T1	2024-01-01 00:10:00	5.5	5.5	...	9.0
...	...	...	...	...	...
T2	2024-01-01 00:00:00	4.5	5.0	...	7.5
T2	2024-01-01 00:10:00	5.0	4.5	...	6.0
...	...	...	...	...	...
TM	2024-01-01 00:00:00	5.5	4.5	...	8.0
TM	2024-01-01 00:10:00	2.0	8.5	...	7.5

Table 3.1: Example of (fictive) data in a turbine-stacked format. The wind farm contains M wind turbines (T1, T2, ..., TM). Each wind turbine has a certain number of observations (one per 10 minutes). Each wind turbine has N signals.

turbines in the wind farm are relevant. Neighboring wind turbines might have an impact (under certain conditions). It is however unlikely that a wind turbine on the other side of the wind farm will be equally influential. The predictors of that wind turbine will most likely not be very useful for modeling the normal behavior. This means that for each time point, a subset should be selected from the wind farm. Determining this group is not trivial and it is part of ongoing research which is outside the scope of this thesis. It would also further complicate the training of the models.

Secondly, using the correlations between the wind turbines increases the dimensionality of the problem dramatically, because the model does not only need to use signals of the wind turbine itself as predictors but also the predictors of the other wind turbines. This has a substantial computational cost. Experiments showed that using the signals of all the other wind turbines in the wind farm did not result in a clear performance improvement. A substantially higher computation cost without clear performance improvement is not interesting economically. Furthermore, the *curse of dimensionality* can even cause the model performance to degrade.

Thirdly, it can have a serious impact on the number of missing values. Dropping rows with at least one missing value becomes less feasible since it is much more likely to find rows with at least one missing value due to for example one wind turbine being offline or even a sensor error at one of the wind turbines. This means that alternative solutions need to be found that impute the missing values, which can be computationally expensive, and require in some cases a model, which introduces new uncertainties.

2.3 Identifying healthy and unhealthy data

An important step in anomaly detection using an NBM model is the selection of healthy data. Since the data that is used in this thesis comes from a real operational wind farm, its observations are not labeled as healthy or unhealthy. This means that healthy data needs to be selected from the input data. This can be done using a model or by using certain expert-knowledge-based assumptions.

In the research presented here, expert-knowledge-based assumptions are used in the shallow ML and autoencoder pipelines. The wind farm median NBM does not require healthy data. It models all data and the assumption is that the abnormal observations will stand out because they are a minority that is different from the majority. For this reason, the wind farm median NBM can be used as a tool to select healthy data for other algorithms that require healthy data. In principle, the autoencoder pipeline does not require healthy training data either. By using hidden layers that compress the input, it should ignore the anomalies during the reconstruction. The difference between the actual observations and the reconstruction should then reveal the anomalies.

However, new insights in the course of this research indicated that the amount of anomalies due to known (and unknown) failures is quite substantial. This means that there is a risk that the autoencoder is not able to distinguish the healthy data from the unhealthy because the anomalies are to a certain extent masked as normal. For this reason, it was decided to "help" the algorithm by training it only on data that is considered by expert knowledge healthy.

$$D_{unhealthy} = \{X : X \in D \wedge t_X \in [t_{fail} - 4\ months, t_{fail} + 1\ month]\} \tag{3.1}$$

where:

X = Observation/sample
D = Dataset
t_X = Timestamp of observation
t_{fail} = Timestamp of failure or replacement

For the shallow ML pipeline, healthy data is identified as the data that is not considered unhealthy. Unhealthy data is considered to be data that precedes the known relevant failures identified by the wind operator by less than 4 months or follows it by less than 1 month. Eq. 3.1 formally defines the unhealthy data for the shallow ML pipeline.

A preliminary analysis of the data showed that this was sufficient, however, a thorough analysis of the results indicated that wider intervals for unhealthy data would be better. For some failures, excluding only 1 month after the failure is insufficient. In some cases,

the wind turbines are down for several months and after restart, there is some time the wind turbine is tested. This data should not be included in the healthy data.

Furthermore, the lists containing the failures or replacements given by the wind turbine operators are not exhaustive. This means that failures that are not deemed relevant are not included. These failures can nevertheless influence the temperatures significantly. This was also observed when analyzing the results of the shallow ML. It led to the decision to try to include these in the unhealthy data for later research.

The procedure to select healthy data used by the autoencoder pipeline is more strict. The new procedure uses two data sources. Firstly, a failure list created by the wind turbine operator, containing several costly failures. This source is also used by the shallow ML pipeline. Secondly, a list of forced shutdowns was extracted from the status logs of the wind turbine. This is new compared to the procedure for the shallow ML pipeline. Research showed that it is important to take them into account. After all, these forced shutdowns correspond with errors that are sufficiently severe to shut down the wind turbine.

$$\begin{aligned} D_{unhealthy} = \{X : X \in D \\ \wedge\, t_X \in [t_{fail} - 4(6)\ months, t_{fail} + 180\ days] \\ \wedge\, t_X \in [t_{fs} - 6\ hours, t_{fs} + 6\ hours]\} \end{aligned} \tag{3.2}$$

where:

X = Observation/sample
D = Dataset
t_X = Timestamp of observation
t_{fail} = Timestamp of failure or replacement
t_{fs} = Timestamp of forced shutdown

Just like for the shallow ML pipeline, data is considered healthy if it is not unhealthy. t_{fail} is the time of a long-duration failure or the replacement of an important component, i.e. generator or gearbox. The t_{fail} is extracted from the first data source, namely, the list of costly failures. All data that precedes the failure with less than 4 or 6 months, or follows it by less than 180 days is considered to be unhealthy. Eq. 3.2 formally defines the unhealthy data for the autoencoder.

The long period after the failure is used to make sure that shutdown and test data after the replacement are discarded. t_{fs} is the time of a forced shutdown. Only observations that happen less than six hours before or after the forced shutdown are considered unhealthy. This is a much narrower interval, but it can be defended as follows. The forced shutdowns that are not included in the operators' lists are less severe and costly. Their effects are only shortlasting. This justifies the usage of a narrower interval. Furthermore, because

there are many more forced shutdowns than costly failures, it is for practical reasons not possible to use large intervals. There would be no healthy data left.

2.4 Splitting data in training, validation, and testing datasets

One of the industry requirements is that the *waiting time*, which is the time it takes to collect sufficient data for the failure prediction algorithms, is as short as possible. For this reason, the amount of training data given to the pipelines is limited. For example, for WF1 4,380 healthy observations per wind turbine (or roughly 183 days) are used by the shallow ML pipeline for 5-fold CV, while for the same wind farm, the autoencoder pipeline uses 3,000 healthy observations for training and 2,500 healthy observations for validation per wind turbine (in total 5,500 healthy observations per wind turbine or roughly 230 days). More data could have been used for training since more was available, but it was explicitly chosen not to. This improves the economic relevance of the methodologies. A second advantage of using less training data is that the computational demand of training the model is lower. This is also an industry requirement.

There is however a trade-off. Firstly, the training data should not be too short. In general, it is preferable to have at least one year of training data due to seasonal fluctuations. If however the idiosyncratic component is used instead of the original signal, seasonal fluctuations will not be an issue, and less than one year of data can be used. Secondly, a smaller number of observations can result in less well-trained models with a degraded performance. However, since the amount of data is substantial (even if only a couple of months of data is used from each wind turbine and the data is aggregated to the 1-hour level), this only becomes a problem when very short training periods are used (shorter than a couple of months).

An advantage of the wind farm median NBM is that it does not require a traditional split into training, validation, and testing datasets since it has no parameters to tune. The shallow ML pipeline does use training and testing data. The data is split in a time series traditional way, meaning, the data is kept in chronological order. This means that the training data is chronologically situated before the testing data. This allows for the examination of model drift, and it corresponds more to how the model would be used in the field.

4 months of healthy training data are used. For the shallow ML models, 5-fold cross-validation (CV) is used for training. This technique can be used because the shallow ML models do not take into account the time dependencies (an exception is in the experiment where lagged values are used, see further). Eq. 3.3 and Eq. 3.4 show the composition of the training and testing dataset more formally.

$$D_{train} = \{X : X \in D_{healthy} \wedge t_X \geq obs_{start} \wedge t_X \leq obs_{start} + 4\ months\} \quad (3.3)$$

where:

X	= Observation/sample.
$D_{healthy}$	= Healthy dataset.
D_{train}	= Training dataset.
t_X	= Timestamp of observation X.
obs	= Observation period (period for which data is available)
obs_{start}	= Timestamp of start observation period.

$$X_{test} = \{X : X \in D \wedge t_X \geq obs_{start} + 4\ months \wedge t_X \leq obs_{end}\} \quad (3.4)$$

where:

X	= Observation/sample.
$D_{healthy}$	= Healthy dataset.
D_{test}	= Testing dataset.
t_X	= Timestamp of observation X.
obs	= Observation period (period for which data is available)
obs_{start}	= Timestamp of start observation period.
obs_{end}	= Timestamp of end observation period.

The training data spans less than a year. This can potentially result in problems with seasonal fluctuations. This can be avoided by using the idiosyncratic component (see section 2.1).

For the autoencoder pipeline, the system is different. The data is split into several datasets for training and testing. The first x months or years (depending on the length of the observation period) are purely used for training and validating the NBM and the anomaly detection models. This dataset is called the training-validation dataset. The remaining y months or years of data are used as a testing dataset and are purely used to validate the complete pipeline. The amount of training-validation data for each wind farm of course depends on the amount of data that is available and the requirements of the models.

The autoencoder pipeline does not use the idiosyncratic component, but the original data. This means that seasonal fluctuations can be an issue. For this reason, more training data is required. The decision not to use the idiosyncratic component follows from the fact that the results become somewhat less intuitive and more difficult to explain to experts.

Wind farm	Training-validation (months)	Testing (months)	Training (obs.)
WF1	22	24	3000
WF2	24	103	8000
WF3	8	5	2000

Table 3.2: Amount of training-validation and test data for WF1-3

Nevertheless, future research will still look into the usefulness of using it together with the autoencoder pipeline.

The autoencoder pipelines use the following amount of training-validation data: WF1 22 months, WF2 2 years, and WF3 9 months. The training-validation dataset is split into several sub-datasets.

- Training dataset: consists of the first x healthy observations for each wind turbine.
- Validation dataset: consists of the next y healthy observations for each wind turbine.
- Threshold dataset: consists of the next z healthy observations for each wind turbine. This is an optional dataset.
- Unhealthy dataset: consists of the unhealthy observations identified in the training and validation dataset using the above-described method.

The training and validation datasets are used for the hyperparameter tuning of the autoencoders. The threshold dataset, which is an optional dataset, is used for tuning the anomaly detection algorithm if this is necessary. This is best done on a separate dataset to get a realistic representation of the reconstruction error. This is not the case when using the training and validation datasets. If the anomaly detection algorithm does not require the tuning of hyperparameters then the threshold dataset is just added to the testing dataset and the two are sorted chronologically. When calculating validation metrics the threshold dataset is however never taken into account to make sure that the results can be compared over the different pipelines as well as possible. Table 3.2 gives an overview of the amount of training-validation and test data for WF1-3.

2.5 Aggregating the data

The wind farm median NBM models do not use aggregated data. Instead, the original 10-minute resolution is used. The goal is to use these models as a benchmark for the complete shallow ML and autoencoder pipelines. This means that not only the NBM models are compared, but also the preprocessing steps of the pipelines. For this reason, the benchmark is kept as basic as possible, which means that no aggregation of the data is done for the wind farm median NBM models.

Both the shallow ML and autoencoder pipelines use data aggregated to the 1-hour level as input. The decision to aggregate is based on several considerations. Firstly, the failures that are the focus of this research are issues that build relatively slowly over time (several days to several months). This does not require data that has a 10-minute resolution. Secondly, by aggregating the data, part of the noise can be removed from it. Thirdly, the aggregated dataset is smaller in size, and processing it requires less computational power.

2.6 Handling measurement errors

In Chapter 2 it was noted that the data from the three wind farms contained a limited amount of measurement errors. Even though their number is small, it would be unsafe to simply ignore their existence and use them in the training of the models. This means that the measurement error problem needs to be solved. A first step is accurately detecting them. There are several ways to do this. It is possible to set maximum or minimum thresholds using expert knowledge, or it can also be done in a data-driven way.

The advantages of the first method are firstly that it incorporates knowledge of physics and mechanics or experience, which can be useful if the knowledge is accurate, and secondly, it makes the error removal procedure straightforward and clear. Disadvantages are however firstly that it requires thresholds for each signal of interest. These need to be determined by experts, which can if there are many signals, be a labor-intensive job. Secondly, it requires also that each time a new signal is added to the data, new thresholds are defined and set. Thirdly, accurate expert knowledge is not always available.

An alternative solution is using a data-driven method. The advantage is that they do not require any expert knowledge or human intervention. The disadvantage is however that it is not always easy to determine which points are measurement errors and which are anomalies caused by the failure of a component or machine. It can be difficult to learn a rule from the data that can make this distinction accurately. This can result in unwanted effects where good or interesting observations are removed.

In the shallow ML pipeline, measurement errors are removed using a data-driven methodology. This is based on signals from which the wind farm median signal value is subtracted (idiosyncratic component). The threshold depends on the median value of the original signal (meaning the signal from which the wind farm median has not been subtracted). This value is multiplied with a scaling factor, which is set to 1. This threshold is used on the signal from which the wind farm median has been subtracted. All absolute values in this signal that are larger than the threshold are considered measurement errors. Concretely this means that values from the wind turbine signal that deviate from the wind farm median by more than its median value are considered measurement errors. Eq. 3.5 shows the procedure using mathematical notation.

$$\text{Error}_{measurement} = \begin{cases} 1 & \text{if } |\text{Value}_{wfc}| > \text{Value}_{original} \text{ outlier\_factor} \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

where:

$\text{Error}_{measurement}$ = Indicator whether the observation is a measurement error or not.
Value_{wfc} = Signal value from which the wind farm median has been subtracted.
$\text{Value}_{original}$ = Signal value.
outlier_factor = Constant multiplier.

The advantage of the latter procedure is that it is a relatively simple way of determining measurement error thresholds, and it does not require expert knowledge. A disadvantage is that there is somewhat less control over what is removed. Certain exceptional circumstances related to situations in which the wind farm median is 0 (this can occur when a majority of wind turbines are shut down), can cause unexpected results for wind turbines that are operational during that time. If the wind turbines have a normal operational temperature that is one degree higher than the wind farm median, this can result in the removal of these valid values. The situation described above is however extremely rare, and no real examples were found.

The issue described above, together with the fact that it is not easy in the case at hand to create a fully reliable model that removes the measurement errors (and not the relevant anomalous values that are associated with a failure), led to the decision to use a hybrid approach, i.e. a combination of expert knowledge and data analysis, for the autoencoder pipeline.

In Chapter 2 it was shown that the measurement errors show themselves in general as extreme values. These values are so extreme that they can easily be removed by setting conservative thresholds that are based on common sense and an analysis of the data. For example, it is very unlikely that the wind turbine components cool down to temperatures below freezing point given the fact that the lowest ambient temperatures registered at the location of the wind turbines are around freezing point. Furthermore, temperature spikes that are nearly 100 °C above the normal temperatures are also very unlikely given the fact that the signals that are used are 10-minute averages.

By removing only these extreme values, the risk of removing abnormal values that are caused by abnormal behavior of the component is reduced. Using a system that is at least in part based on expert knowledge and manually setting thresholds is feasible because the number of signals that are used is manageable. It removes some uncertainty from the pipeline, which is a good thing if the result needs to be reliable. It is of course possible that some measurement errors that create less extreme values are missed. However, a thorough analysis of the data indicated that their number is most likely small.

Tables A.1, A.2, and A.3 in the Appendix show for respectively WF1, WF2, and WF3 the thresholds that are used for each signal. For each signal, a lower and upper bound is created below or above which the sensor values are assumed to be unrealistic. For WF1 and 3, the most relevant thresholds are the lower bounds since they occur with the highest frequency.

For WF2 both the negative and positive thresholds are relevant. As can be seen in Figure 2.2b in Chapter 2, the data contains measurement errors on both the low and high end. The measurement errors on the high end are detectable by the fact that their values are much higher than the normal values, and many of the high values all have the same value (205.0°C). The latter indicates that this must be a default sensor value.

If a measurement error is found, then the value is set to missing for both the shallow ML, and autoencoder pipelines. Since the number of clear measurement errors is small, the number of new missing values is small.

2.7 Handling missing values

In Chapter 2 it was shown that the data from WF1, WF2, and WF3 contain a limited amount of missing values. Furthermore, the handling of the measurement errors created also a small number of missing values. Most ML models can not handle missing values themselves. This means that a solution needs to be found for them before any modeling can be done. There are several ways this can be done, but the solution depends also on the properties of the mechanism that generates the missing values. These properties have been thoroughly discussed in Chapter 2. There it was indicated that the missing data is most likely MCAR on MAR. Furthermore, the amount of missing data is small.

The wind farm median NBM does not impute the missing values. Instead, the missing values are ignored if their number at a given point in time is limited. If there are many missing values, no wind farm median is calculated since it would most likely be unreliable. Instead, a missing value is generated. Eq. 2.1 is used to decide whether the wind farm median is calculated or not.

In the shallow ML pipeline, linear interpolation is used to impute the missing values. This means that they are filled in by taking the linear interpolation between the last known value before and the first known value after the missing value. A thorough analysis of the imputation showed however that in some cases, mostly when several sequential observations are missing, this technique can have unwanted consequences. The missing values are in those situations sometimes imputed with values that make no sense if one considers the values of the other signals. This pollutes the data, which is detrimental to the training of the models. The problem became only clear when analyzing the results of the pipeline. For this reason, this technique is still used by the shallow ML pipeline.

The deep learning methodology removes the missing values instead of imputing them. This is appropriate since the analysis of the missingness indicated that the missing data is most likely MCAR or MAR. This assessment is based on the fact that the missingness is caused by sensor failures. There is no indication that they are caused by anything related to the wind turbine operation or the component temperatures. This means that the probability that removing the observations with missing values introduces bias is small.

Other imputation techniques could have been used. To avoid that the imputed values are nonsensical when compared to the values of the other signals, model-based imputation can be used. However, this requires the availability of such a model, which is exactly the goal of the NBM step. Multiple imputation and other more complex imputation systems are less suitable for the task at hand due to their (computational) complexity and the fact that they also add uncertainty to the pipeline.

2.8 Scaling of the data

Input signals that have large differences in scaling are not ideal for machine learning. Some algorithms can handle this out of the box, but for many algorithms, this is not the case. For this reason, it is advisable to scale the input signals. de Amorim et al. [2023] shows that the choice of scaling technique can have an impact on the results. Nevertheless, the selection of a specific technique is seldom thoroughly explained in papers.

There are several different techniques that can be used to scale the data, e.g. standard scaler, min-max scaler, maximum absolute scaler, robust scaler, and quantile transformer. Each technique has advantages and disadvantages. An advantage of the standard scaler method is that it can transform positive and negative values. A disadvantage is that it is influenced by outliers. The same is true for the min-max scaler and the maximum absolute scaler. The robust scaler and quantile transformer are robust for outliers [de Amorim et al., 2023].

$$x_{stand} = \frac{x - \bar{x}_{train}}{\sigma_{x_{train}}} \tag{3.6}$$

where:

x = A time series or signal x.
x_{train} = x values in the training data.
$\overline{x}_{train}$ = The mean of x_{train}.
$\sigma_{x_{train}}$ = Standard deviation of x_{train}.

$$x_{norm} = \frac{x - min(x_{train})}{max(x_{train}) - min(x_{train})} \tag{3.7}$$

where:

x_{train} = x values in the training data.
min(...) = Minimum value of a series or array.
max(...) = Maximum value of a series or array.

The wind farm median NBM does not require scaling of the signals. The shallow ML and autoencoder pipelines do. The predictors (not the targets) of the shallow ML pipeline are standardized when modeling. Standardizing consists of subtracting the training data mean from the samples and dividing by the training data standard deviation (see Eq. 3.6). This centers the distributions of the signals around 0 and gives them a standard deviation of around 1. The means and standard deviations of the training datasets are used to standardize the test dataset signals, to avoid information leakage.

The autoencoder pipeline normalizes (which is not the same as standardizing) all signals before modeling. This is done by calculating the minimum and maximum of each signal on (only) the training dataset. The min is subtracted from the signal value and the difference is divided by the difference between the maximum and the minimum of the signal (see Eq. 3.7).

The reason for using scaling methods that are not robust for outliers in both pipelines is the following. Firstly, these techniques are most often used in the literature, and their impact on the training of machine learning algorithms is relatively well understood. Secondly, the largest outliers caused by measurement errors have already been removed during preprocessing. They will not influence the scaling anymore. Thirdly, the data that is used for modeling describes physical processes that can only achieve values within certain ranges that are physically possible. This means that very large (or small) outliers (which are not measurement errors) are not possible.

3 Methodologies for modeling the normal behavior

3.1 The wind farm median as benchmark

The wind farm median of the different temperature signals can be considered as an *implicit NBM*. Implicit refers to the fact that no mathematical model is constructed, i.e. no predictors are selected, and no model is trained, ... This NBM is based on the assumption that in general, a majority of the wind turbines (in a large wind farm) are in a healthy state, meaning they do not suffer from major failures. This healthy state corresponds to the normal behavior state given the environmental conditions. Some wind turbines may contain components that are damaged, however, as long as they make up less than 50% of the wind farm, they will not influence the median. The wind farm median was also used

Datetime	Signal A			Wind farm median	Error		
	WT1	WT2	WT3		WT1	WT2	WT3
2012-01-01 00:00:00	2.0	3.0	4.0	3.0	-1.0	0.0	1.0
2012-01-01 01:00:00	3.0	4.0	5.0	4.0	-1.0	0.0	1.0
2012-01-01 02:00:00	1.0	2.0	3.0	2.0	-1.0	0.0	1.0

Table 3.3: Table with an overview of how the wind farm median and the wind farm prediction error is calculated for a fictive wind farm with three wind turbines.

as NBM in Chesterman et al. [2021] and Chesterman et al. [2022]. The wind farm median NBM is what is equal to what is called in previous sections the *common component*.

Table 3.3 shows how the wind farm median is calculated. The median is calculated for each timestep in the dataset. This means that for timestamp t_1 all the values of a certain signal of all the wind turbines in the wind farm are collected and the median is calculated. The result is a time series of median values for each signal. The difference between the corresponding signal of each wind turbine and this signal is calculated. This is the wind farm median error and is used to estimate whether a wind turbine behaves normally or not. If a wind turbine suffers from a bearing failure then it is expected that the bearing temperature will rise just before the actual failure. This will show itself as a positive deviation from the wind farm median.

3.2 Shallow machine learning pipeline

The shallow ML pipeline contains multiple levels (see Figure 3.5). The first level uses several machine learning models, e.g. elastic net, random forest, light gradient boosting machine, and multilayer perceptron, in parallel for learning the normal behavior of a single target signal. Each model is an NBM on its own, which means that they model each the normal behavior of the target signal.

$$L(\lambda_1, \lambda_2, \beta) = |\mathbf{y} - \mathbf{X}\beta|^2 + \lambda_2|\beta|_2 + \lambda_1|\beta|_1 \tag{3.8}$$

where:

X = Predictors.
y = Target.
β = Model coefficients.
$|\beta|_2 = \sum_{j=1}^{p} \beta_j^2$, L_2-norm regularisation term.
$|\beta|_1 = \sum_{j=1}^{p} |\beta_j|$, L_1-norm regularisation term.
L = Loss function.

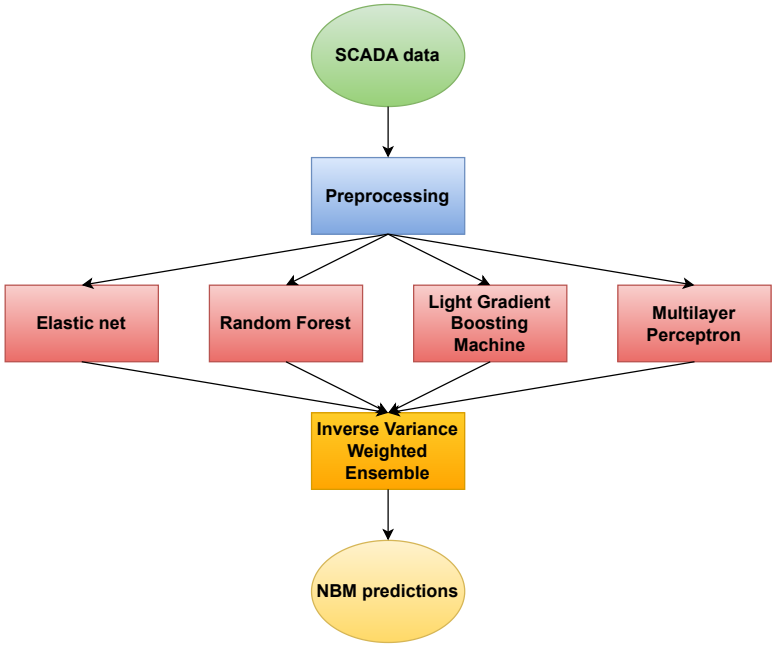


Figure 3.5: Schematic overview of the shallow ML-pipeline.

The elastic net regression model is a linear regression model to which L_1 and L_2-regularization terms have been added. The elastic net method was presented first in Zou and Hastie [2005]. The loss function is described by Eq. 3.8. The goal is to find $\hat{\beta}$ that minimizes the loss function $\hat{\beta} = \arg\min_\beta L(\lambda_1, \lambda_2, \beta)$. The L_1 corresponds to the regularization term used in the least absolute shrinkage and selector operator (LASSO) method, and the L_2 to the one used in the Ridge method. The L_1 will make certain coefficients or weights of predictors 0 which results in a feature selection. The L_2 will shrink coefficients to 0 but will not make them exactly equal to 0.

$$\hat{f}_{rf}^{B}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x; \Theta_b) \tag{3.9}$$

where:

$T(x;\Theta_b)$ = A tree.
Θ_b = The split variables, cutpoints at each node, terminal-node values of tree b.
B = The number of trees.
$\hat{f}_{rf}^{B}(x)$ = Prediction by the random forest for observations x.

The random forest model was first presented in Breiman [2001]. A random forest is a combination of decorrelated decision trees. The tree-growing process of each tree is based on a subset of randomly selected input signals. By using decorrelated trees it attempts to improve the variance reduction compared to bagging, which is also a tree-based algorithm, but which does not randomly select input signals. Each tree generates a certain value (in case of regression) for an observation. The answers of the different trees are combined through averaging (in case of regression) and this is the result of the random forest. Eq. 3.9 gives a mathematical description of the random forest [Hastie et al., 2017].

gradient boosting machine (GBM) are a different type of tree-based learners. The idea behind boosting is to combine many weak learners to create a strong *committee*. A weak learner performs only slightly better than random [Hastie et al., 2017]. This is a major difference with random forest, which uses many strong learners.

One of the most popular original boosting algorithms is Adaboost.M1 developed in Freund and Schapire [1997]. The weak learners are applied sequentially to the data. The predictions of all of them are combined using weighted majority voting in the case of classification or averaging in the case of regression. This is similar to the random forest (Eq. 3.9). The observations in the dataset are given weights. At the start of the learning, all weights are equal. Based on how well the observations are learned by each learner the weights are modified [Hastie et al., 2017].

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{3.10}$$

where:

$\Omega(f_t)$ = Penalty term $\gamma T + \frac{1}{2}\lambda||w||^2$.
y_i = The true value of observation i (target).
$\hat{y}_i^{(t-1)}$ = The estimate of the value of observation i after $t-1$ iterations (weak learners).
$f_t(x_i)$ = The weak learner that most improves the model.

Over time many different modifications have been developed. Gradient boosting applies steepest descent minimization as a numerical optimization algorithm to find the most optimal solution [Friedman, 2001]. It learns iteratively a new weak learner to improve the performance of the overall model. Eq. 3.10 gives the loss function [Chen and Guestrin,

2016]. The weak learner that is selected in each step is the one that reduces the loss function the most.

A very competitive version is XGBoost which is more scalable [Chen and Guestrin, 2016]. Light GBM is an even more efficient version, of the well-known gradient boosting decision tree (GBDT) algorithm. It was first presented in Ke et al. [2017]. Light GBM was developed to further solve efficiency and scalability issues with other implementations (e.g. XGBoost, pGBRT) when the amount of data is large and the dimensionality is high. This is done through two innovations, i.e. gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB).

$$x_j^{d+1} = \sum_i y_i^d w_{ij}^d - \alpha_j^{d+1} \tag{3.11}$$

where:

y_i = Output of neuron i in layer d.
w_{ij} = Weight assigned to the output of neuron i in layer d.
α_j^{d+1} = Bias term.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.12}$$

where:

x = Input.
f(x) = Output.

$$f(x) = \begin{cases} x & \text{if x} > 0, \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

where:

x = Input.
f(x) = Output.

A multilayer perceptron is a universal approximator. This means that it can approximate any continuous function. The name of the model comes from the fact that it consists of neurons that are called *perceptrons* [Menzies et al., 2015]. Figure 3.6 gives a schematic representation of a perceptron. A perceptron takes as input values of different signals, i.e. x_1, x_2, x_3, x_4, and x_5. It calculates a linear combination of these inputs by weighing them using the weights w_1, w_2, w_3, w_4, and w_5, and subsequently adding them. This result

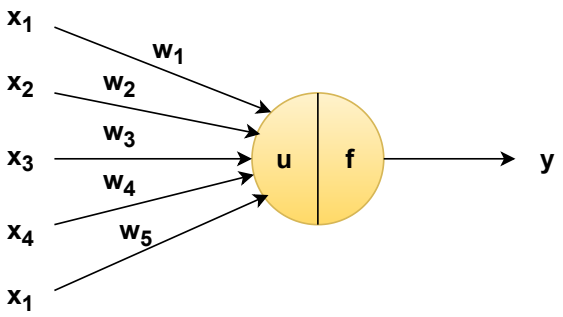


Figure 3.6: Schematic overview of a perceptron.

is then used as input for a function f, also called an activation function, that transforms it [Menzies et al., 2015].

Several functions can be used for this. Examples of common activation functions are the identity function, the logistic or sigmoid function, the rectified linear unit (ReLU) function, ... The identity function does not perform any transformation. The sigmoid function (Eq. 3.12) transforms its input into a value between 0 and 1. The ReLU function (Eq. 3.13) transforms its input into a value between 0 and $+\infty$. The *perceptron* is called a linear classifier because it works well when the groups of data points can be linearly separated. It is trained by adjusting the weights (w_1, w_2, w_3, w_4, and w_5). This is done by minimizing a specific metric [Menzies et al., 2015].

The strength of a multilayer perceptron is that many perceptrons can be combined. This makes it possible to model much more complex problems. This is the idea behind the multilayer perceptron. It is a supervised FNN that consists of an input layer, one or more hidden layers, and an output layer. A neuron or perceptron j in layer $d+1$ receives a total input according to Eq. 3.11. All neurons in all layers except the input layer have an activation function that transforms the output. The multilayer perceptron is trained using the backpropagation algorithm [Menzies et al., 2015].

$$\hat{\mu} = \frac{\sum_i w_i x_i}{\sum_i w_i} \tag{3.14}$$

where:

x_i = Prediction by learner i.
$w_i = \frac{1}{\sigma_i^2}$, weight assigned to predictions learner i.
σ_i^2 = Variance predictions learner i.

$$MAD = \frac{\sum |x_i - \tilde{x}|}{n} \tag{3.15}$$

where:

x_i = Observation value.
$\overline{x}$ = Average value of the observations.
n = Number of observations.
MAD = Median Absolute Deviation

$$std_{robust} = f_{scaling}\, MAD \tag{3.16}$$

where:

$f_{scaling}$ = Scaling factor which is set to 1.4826 if data is normally distributed.

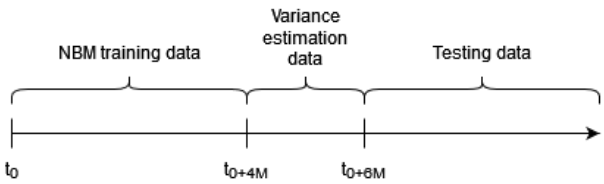


Figure 3.7: Schematic overview of how the data is used for training and testing of the shallow ML methodology. t_0 is the first healthy observation for a wind turbine.

The second (ensemble) level combines the predictions of the individual NBM models using Inverse Variance Weighting (Eq. 3.14). The variance of the errors of the NBM models is estimated using the median absolute deviation (MAD). MAD is used because it is much more robust for outliers (Eq. 3.15, 3.16). For this, a separate dataset (the variance estimation data) containing two months of healthy data is used. Figure 3.7 gives an overview of how the data is divided for the training of the ensemble machine learning model.

3.3 Autoencoder pipeline

Autoencoders are deep learning algorithms that consist of two parts, i.e. an encoder (a function f) and a decoder (a function g). The goal of the encoder is to create a (lower-dimensional) representation of the original data ($h = f(x)$). The decoder uses this representation (h) to recreate x. The autoencoder is given a set of inputs x, and its goal is to reproduce x as well as possible. This is done by minimizing a loss function $L(x, g(f(x)))$. The loss function penalizes $g(f(x))$ for being dissimilar to x. An example of such a function is the mean squared error (MSE) [Goodfellow et al., 2016].

In this sense, the autoencoder can be considered as a non-linear version of a PCA, in which the number of principal components is less than the total number of components. The reduced number of principal components can then be used to recreate the original signal. This recreation will not be perfect due to the loss of information. Strictly speaking, it is not required that h has a lower dimensionality than x. h can in principle have the same number of dimensions, or even more dimensions (*overcomplete autoencoder*). However, in general, the number of dimensions is smaller (*undercomplete autoencoder*). The reason for this is that by reducing the dimensionality, the algorithm is forced to learn the most interesting features in the data, which can give valuable insights.

In the case of anomaly detection, which is the application in this paper, this dimension reduction strategy is valuable. This is because it is assumed that anomalies are rare events. Undercomplete autoencoders tend to ignore unimportant or rare events. Due to this property, we expect the low-dimensional representation h and the reconstruction $g(f(x))$ to ignore these events. The reconstruction error L, which is the difference between the observed and reconstructed values, is searched for anomalies.

One could wonder why a healthy dataset is still used in this setup. The reason for this is that some failures cause very long periods of abnormal behavior. This together with other less significant but still anomalous behavior can result in a situation where the anomalies are no longer rare. This means that it is possible that the autoencoder still learns anomalous behavior. By training it on data that has already been filtered a first time this risk is reduced. It still is useful to use an undercomplete autoencoder nonetheless. The first filtering is not perfect, which means that it is likely that it still contains some anomalous behavior. To avoid that the model learns this, a hidden layer with lower dimensionality is useful.

Several restrictions were put on the structure of the autoencoder. Firstly, the encoder and decoder are each other's mirror images. Secondly, during hyperparameter tuning architectures are tested with 1-6 hidden layers for the encoder (the last hidden layer of the encoder is shared between the encoder and decoder). Thirdly, the number of units per layer is during hyperparameter tuning restricted using Eq. 3.17, 3.18. The minimum number of

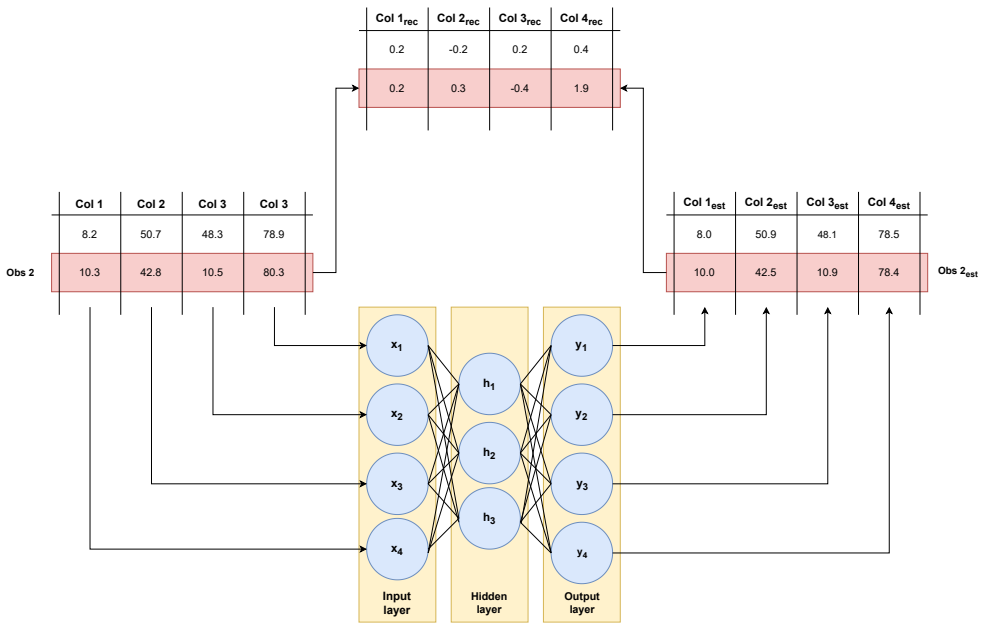


Figure 3.8: Schematic representation of undercomplete autoencoder

units ($units_{min}$) is set to 2 (a unit is a different name for a neuron). Fourthly, the layers of the autoencoder are all dense layers (which means that all neurons of a layer are connected to all neurons of the previous and next layer) with an activation function, which is also a hyperparameter. Fifthly, the activation functions are selected by the hyperparameter tuning from the following possibilities: exponential linear unit (ELU) (Eq. 3.19), ReLU (Eq. 3.20). The last layer of the output layer of the autoencoder uses a sigmoid activation function. Sixthly, the optimal learning rate is selected by the hyperparameter tuning from the following possibilities: 0.01, 0.001, 0.0001. The hyperparameter tuning algorithm is the Keras Hyperband tuner, which is based on the algorithm developed in Li et al. [2018]. This algorithm was selected for efficiency reasons. In the paper, the authors focus on speeding up the random search of the parameter space through adaptive resource allocation and early stopping.

Restrictions on the structure of the autoencoder are used to limit the size of the search space. From experiments, it became clear that adding many layers to the encoder and decoder does not improve the performance. For this reason, the search space was limited to at most 6 layers. The number of units per layer was limited in such a way that the

number of units per layer slowly decreased with each hidden layer. This way was preferred because it results in a gradual dimension reduction. Making the encoder and decoder mirror images was chosen because this is more or less the standard in the literature. The same reasoning is behind the selection of the activation functions and the learning rate. The run time of the hyperparameter tuning is around 3-4 hours on a standard server.

The autoencoder is set up as a multi-input multi-output model. This implies that the normal behavior of several signals is modeled at the same time. Figure 3.8 gives a schematic representation of how this works. The autoencoder has in its input layer a number of neurons that is equal to the number of input signals. The output layer has the same number of neurons. This means that in a single run, the normal behavior for all the signals is predicted. This is a considerable advantage since it reduces the run time of the algorithm. The reconstruction error is then the difference between the original signal values and the predicted values.

$$units_{min,i} = max(units_{max,i-1} - \lfloor \frac{features}{layers} \rfloor, units_{min}) \tag{3.17}$$

where:

features	= Number of features or predictors.
layers	= Number of layers.
$\lfloor \frac{x}{y} \rfloor$	= Floor division of x by y.
units_{min}	= Lower bound of number of units or neurons in layer.
$\text{units}_{max,i-1}$	= Upper bound of number of units or neurons in previous layer.

$$units_{max,i} = max(units_{max,i-1} - 1, units_{min}) \tag{3.18}$$

$$f(x) = \begin{cases} x, \ if \ x > 0 \\ a(e^x - 1), \ otherwise, \ with \ a \geq 0 \end{cases} \tag{3.19}$$

$$f(x) = \begin{cases} x, \ if \ x > 0 \\ 0, \ otherwise \end{cases} \tag{3.20}$$

The autoencoder does not use dropout layers. This type of layer is in some research papers used to avoid overfitting of neural networks [Srivastava et al., 2014]. However, an analysis of the results showed that this modification did not result in improved modeling performance on the validation dataset, while at the same time, it increased the training time. For this reason, dropout layers were not used in the final model architecture.

4 Validation of the NBM models

To be able to compare the performance of the three methodologies discussed above, they are subjected to the same validation procedure. To this end, data from three real operational (off-shore) wind farms is used, i.e. WF1, WF2, and WF3. The properties of the data are described in Chapter 2.

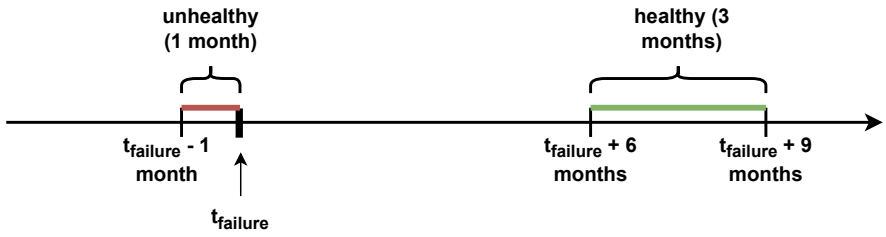


Figure 3.9: Schematic representation of the selection of healthy and unhealthy data

$$UHH - ratio = \frac{MAE_{UH}}{MAE_{H}} \tag{3.21}$$

where:

MAE_{UH} = Mean absolute error for unhealthy data.
MAE_{H} = Mean absolute error for healthy data.

To understand how well the models perform, two metrics are calculated, i.e. the MAE on healthy data and the UHH-ratio. A small value for the first metric indicates that the model performs well on the healthy behavior of the wind turbines. In practice, it means that for the wind farm median NBM the MAE of the idiosyncratic component is calculated. For the shallow ML pipeline, it means that the MAE of the prediction error on healthy data of the ensemble of shallow ML algorithms is calculated. For the autoencoder pipeline, it means that the MAE of the reconstruction error on healthy data is calculated. These values for the three methods are compared.

The second metric measures how useful the model is as NBM. A good NBM is a model that distinguishes healthy and unhealthy behavior well. To quantify this, the MAE on the unhealthy data is compared to that on the healthy data. This is done by calculating the ratio of the two (see Eq. 3.21). An NBM is considered useful if the ratio is significantly larger than 1 for the signals that should be able to capture the impact of damage to

the components. This indicates that the reconstruction error is considerably larger for unhealthy than healthy data. The other signals show preferably a ratio of around 1.

In practice, it means for the wind farm median NBM that the mean absolute value of the idiosyncratic component is calculated for healthy and unhealthy data. The ratio between the two is then calculated to understand how well the wind farm median can distinguish the unhealthy from the healthy data. For the shallow ML and autoencoder pipelines the mean absolute value of respectively the prediction and reconstruction errors is calculated for healthy and unhealthy data. Just as for the wind farm median NBM the ratio between the two is calculated. The results of the three methods are compared.

It is important to note that unhealthy and healthy data are interpreted here differently than previously. Unhealthy data is the data that precedes the failure by less than one month. The reason for using only such a narrow interval for unhealthy data is that it gives us greater certainty that the damage to the component(s) is present in the data. Healthy data is defined here as data that follows a failure between 6 and 9 months. Data less than 6 months after the failure is discarded because it likely contains downtime and startup or testing behavior, which is not relevant to this exercise. Figure 3.9 gives a schematic overview. It is unlikely that new damage has already formed 6-9 months after the last failure.

The latter consideration is the reason why healthy data is selected after a failure and not before. It is often not clear how long before the failure a component is already damaged. This means that one runs a higher risk if one defines healthy data as data that precedes a failure by several months. By using data after the failure, it is more likely that the component is undamaged because it has been replaced recently. This gives more assurance on the state of the component.

5 Comparison of the performance of the wind farm median benchmark, the shallow ML, and autoencoder pipelines

In the previous sections, three methodologies have been discussed. The first method is the wind farm median NBM benchmark. The second method is based on an ensemble of shallow ML algorithms, e.g. elastic net, random forest, light GBM, and multilayer perceptron (MLP). The third method is based on the autoencoder. The different properties of the models have been discussed. In this section, a comparative analysis is done of the three methodologies.

Two of the methodologies are pipelines. This means that they consist of several steps, not just shallow ML or autoencoder algorithms. They contain also several preprocessing

steps. By comparing the complete pipelines, the differences between these steps are also evaluated. The two pipelines are compared to a baseline, i.e. the wind farm median benchmark. The wind farm median can be considered as an implicit NBM. By using this model as a benchmark it will become clear how much better the other pipelines are at modeling the temperature signals.

For this comparison, data from WF1, WF2 and WF3 are used. The analysis will focus for WF1 specifically on the generator failure case (generator short-circuit). The generator short-circuit failures are challenging to detect because the electrical failure can only be found indirectly in the temperatures. For WF2 the comparison will focus on the generator bearing failure case. These failures are more traditional mechanical failures for which temperature analysis should be suitable. This should make them easier to detect. WF3 did not experience any failures during the observation period. For this reason, this wind farm will only be used for the comparison of the modeling performance on healthy data. A more detailed analysis of the failure types can be found in Chapter 2.

5.1 WF1

Figure 3.10 shows the results for WF1. The first thing to note is that the wind farm median NBM has a relatively large MAE (between 3 and slightly over 5 °C). This is as expected since the wind farm median is unable to model any (transient) behavior that is specific to a minority of wind turbines. Secondly, the MAEs for the shallow ML and autoencoder pipelines are substantially smaller (at most 1.5 °C). This indicates that the models can model some of the wind turbine-specific behavior that the wind median can not.

Thirdly, for 4 out of 11 signals, the autoencoder has a smaller MAE on healthy data than the shallow ML. For 6 out of 11 signals, it is the opposite. For example, the MAE of the autoencoder is smaller for the generator phase 1M2, generator phase 3M2, and the temperatures of the front and rear generator bearings. For the generator phase 1M1, generator phase 2M1, generator phase 2M2, the generator phase 3M1, and the other generator phase temperatures, the opposite is true.

However, these results do not tell everything. As it has already been explained, for anomaly detection it is not that important whether the NBM has a low MAE on the healthy data. It is much more important that it can distinguish healthy from unhealthy data. This can be determined by calculating the UHH-ratio.

The results, shown in Figure 3.11, give again a mixed message. Firstly, the UHH-ratio for the wind farm median NBM is for all signals smaller than 1. This means that the MAE of the wind farm median NBM on unhealthy data is never larger than the NBM on healthy data. This indicates that this model has trouble distinguishing the data that is related to the generator failures from data that is healthy. This is not surprising. The

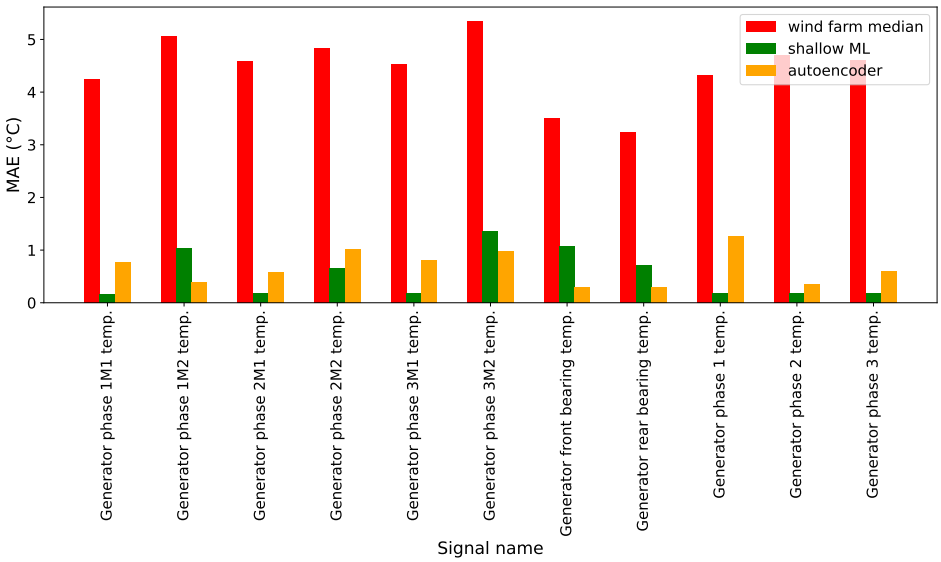


Figure 3.10: MAE on healthy WF1 data for wind farm median (benchmark), shallow ML, and autoencoder pipeline.

generator failures are challenging to detect because they can only under some conditions be seen in the temperatures of the generator phases. The wind farm median benchmark fails to detect it.

Secondly, both the shallow ML and autoencoder pipelines have several UHH-ratio related to generator phase temperatures that are substantially larger than 1. Interestingly, both pipelines do not fully agree on which signals behave suspiciously and which do not. The shallow ML pipeline generates MAE ratios significantly larger than 1 for generator phase 2M1 temp, generator phase 2M2 temp, generator phase 3M2 temp, and generator phase 3 temp. The autoencoder pipeline finds high UHH-ratios for the generator phase 1M2 temp, generator phase 2M1 temp, generator phase 3M2 temp, generator phase 2 temp, and generator phase 3 temp. However, these differences should not be exaggerated. The results of both are more or less in line.

Thirdly, the generator bearing temperature ratios are for both pipelines not significantly larger than 1. This is also an important finding. It is not expected that this generator failure type is visible in the generator bearing temperatures. This means that the pipelines find only UHH-ratios larger than 1 for signals that are relevant to the failure mode.

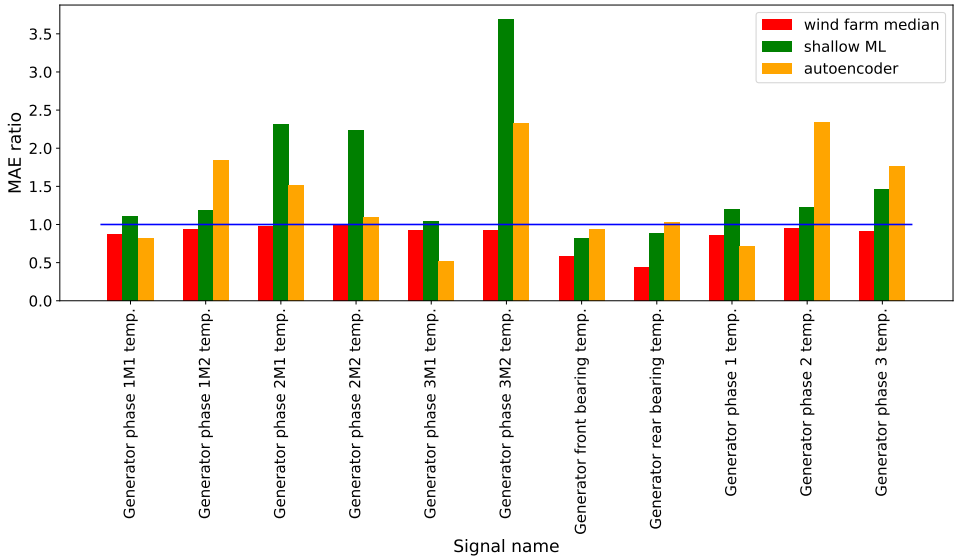


Figure 3.11: MAE ratios on WF1 data for wind farm median (benchmark), shallow ML, and autoencoder pipeline. The blue horizontal line indicates a UHH-ratio of 1.

In conclusion, one can state for WF1 that the shallow ML and autoencoder pipelines outperform the wind farm median benchmark for this failure mode. However, based on these results, it is difficult to decide which of the pipelines performs the best. This is surprising since one would expect that the autoencoder outperforms the shallow ML models given the supposed complexity of the problem at hand which should favor complex highly non-linear models. Possible explanations are perhaps the predictiveness of the generator phase temperatures for the failure mode or the quality of the predictors of the model used to model the generator phase temperatures. They only capture indirectly the problem, which means that the amount of information they contain is limited. This might explain why the autoencoder is unable to outperform the shallow ML models.

5.2 WF2

Figure 3.12 shows the healthy MAE results for WF2. They contain several interesting elements. Firstly, the results for WF2 show again that the wind farm median has for most temperature signals a large MAE. The exception is the generator cooling water

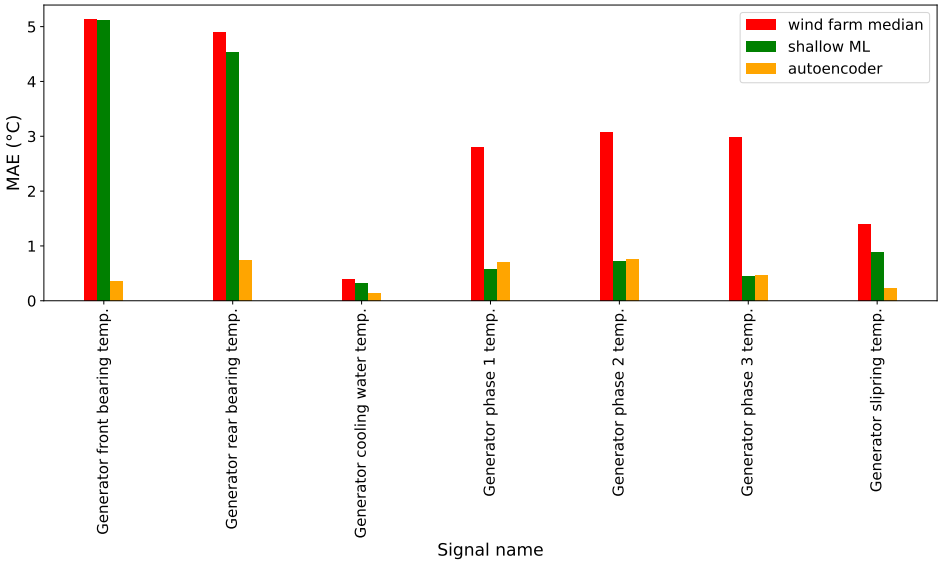


Figure 3.12: MAE on healthy WF2 data for wind farm median (benchmark), shallow ML and autoencoder pipeline.

temperature signal. For most signals both the shallow ML and autoencoder pipelines have healthy MAEs much smaller than that of the wind farm median.

Secondly, for WF2 the advantage of using the autoencoder pipeline is more clear. Figure 3.12 shows that the shallow ML pipeline has problems modeling the temperatures of the generator bearings. It performs barely better than the wind farm median. It is unclear why this exactly is. The loss of fit for the front generator bearing might be explained by the bearing slip issue. However, this does not explain the loss of fit for the rear generator bearing. The autoencoder does not suffer from this problem. The MAE for these signals is lower than 1 °C and is in line with the error of the other signals.

The errors for the other signals are for the shallow ML and autoencoder more or less the same. For both models, they are smaller than 1 °C. This is a good performance all things considered. The healthy MAE results of the autoencoder pipeline for WF2 are more or less in line with those for WF1. This is not the case for the shallow ML pipeline, due to the weak modeling performance for some of the signals of WF2.

Figure 3.13 shows the UHH-ratios for the different signals. This figure contains also several interesting results. Firstly, the UHH-ratios of the wind farm median, shallow ML, and autoencoder models are more or less similar, even though the MAE on the healthy data

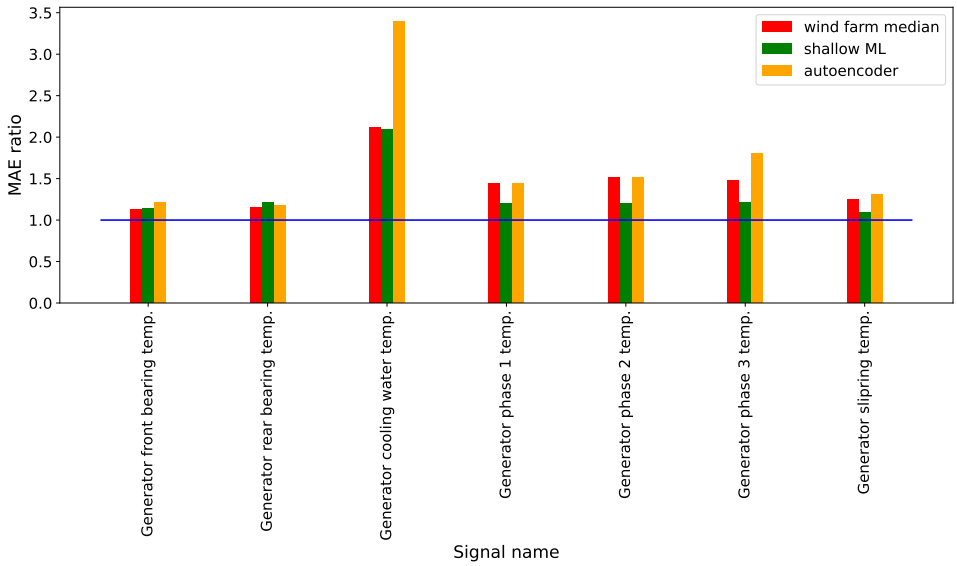


Figure 3.13: UHH-ratios on WF2 data for wind farm median (benchmark), shallow ML and autoencoder pipeline. The blue horizontal line indicates a UHH-ratio of 1.

is much larger for the wind farm median and shallow ML pipeline than for the autoencoder pipeline. This indicates that it is not necessarily the case that the model with the smallest MAE on healthy data (meaning it is the best modeler of healthy data), is also the best at distinguishing healthy from unhealthy data.

The UHH-ratios for the generator bearing temperatures are larger than 1 (between 1.2 and 1.3) for both pipelines. This is as expected since the failures are related to the generator bearings. The UHH-ratios for the bearing signals are however not much larger than 1. This might be the result of the bearing slipping issue of which the consequences are widespread in the temperature signals of mainly the front but also to a certain extent the rear generator bearing.

The UHH-ratio for the generator cooling water temperature is substantially larger than 1. It is larger for the autoencoder than for the shallow ML pipeline. This result can be explained by the overheating of the bearings which influences the temperature of the generator and the cooling water. Interestingly, the UHH-ratio of the cooling water is larger for the bearing temperatures. It seems that the cooling water temperature of the generator can be used for predicting a generator bearing failure. It will however not allow for making a distinction between DE and NDE bearing failures.

5.3 WF3

For WF3, no known failures of the generator are available. For this reason, the MAE on unhealthy data and the UHH-ratio can not be calculated. It is however possible to calculate the MAE on healthy data. These results can still be interesting for understanding how well the healthy data modeling performance of the different NBM models generalizes to other datasets. Just as for WF1 and WF3, the results will be compared for the wind farm median (benchmark), the shallow ML, and autoencoder pipelines.

The results for WF3 (see Figure 3.14) indicate that both the shallow ML and autoencoder pipelines outperform the wind farm median. Furthermore, in general, the autoencoder pipeline outperforms the shallow ML pipeline. This is the case for 10 of the 12 signals. The healthy data MAE for the autoencoder pipeline is always less than or slightly above 1 °C.

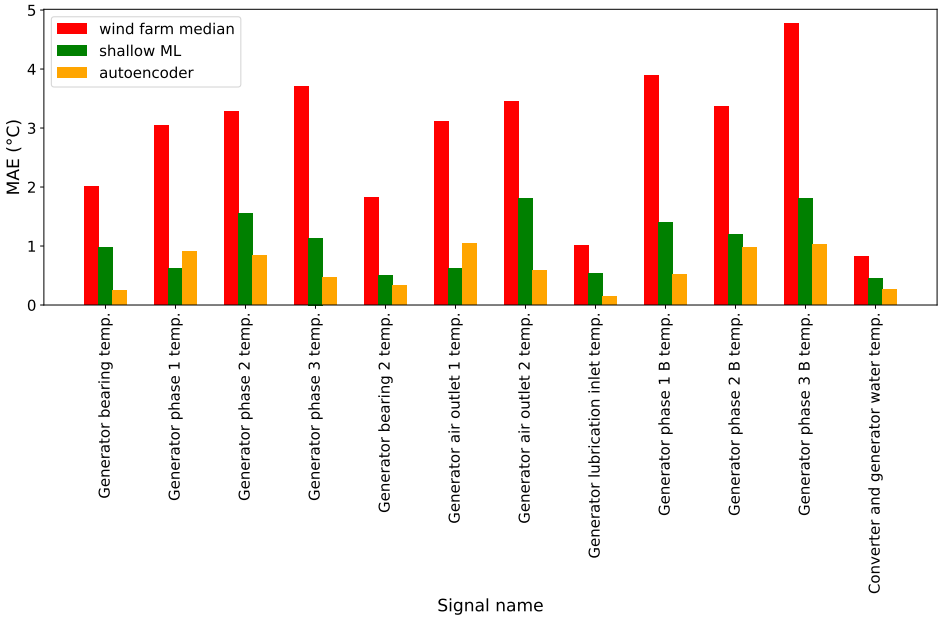


Figure 3.14: MAE on healthy WF3 data for wind farm median (benchmark), shallow ML, and autoencoder pipeline.

From the results described above one can conclude that in general both the shallow ML and autoencoder pipelines have an added value compared to the wind farm median. The healthy data MAEs are often smaller than those of the wind farm median benchmark.

The shallow ML and autoencoder pipelines also tend to be better at distinguishing healthy from unhealthy data (see UHH-ratios), which is a minimum for failure prediction.

5.4 Conclusion based on comparison of results

An overview of the results has shown that for a majority of the signals, the autoencoder pipeline has a lower healthy data MAE than the shallow ML pipeline. In 18 cases the autoencoder pipeline has the smallest MAE, while the shallow ML pipeline has the smallest MAE in 12 cases. Furthermore, in 2 of the cases where the shallow ML pipeline comes out as the best, its MAE is only marginally smaller than that of the autoencoder pipeline. However, the MAE on healthy data does not say everything. An analysis of the UHH-ratio shows that the autoencoder pipeline slightly outperforms the shallow ML pipeline. The autoencoder pipeline finds in 6 of the relevant signals a clearly positive UHH-ratio. The shallow ML pipeline finds this in only 5 signals. One can discuss whether the UHH-ratio larger than 1 found by the shallow ML pipeline for the generator phase 1 temp signal in WF1 is not also sufficiently large to count. However, the ratio is only marginally larger than 1. So for this reason it is ignored.

It needs to be pointed out however that the results for the healthy data MAE and UHH-ratio metrics are stochastic. This means that they contain some uncertainty. This is not shown in the plots. To quantify this uncertainty confidence bounds can be calculated for these metrics using schemes like Bootstrap. However, this would computationally be very expensive and for this reason, it has not been done yet.

6 Concluding remarks concerning the NBM modeling

During this research, many different models and configurations have been tested. Some did produce good results, others did not. In what follows an overview is given. This section discusses the results found in Chesterman et al. [2023a] and extends them with the findings in this thesis.

6.1 Limited evidence for added value of using highly non-linear models as NBM when using data from which the wind farm median has been subtracted

In the course of this research, we conducted experiments with different types and configurations of NBM models. One of the most surprising results of the shallow ML research presented in Chesterman et al. [2023a] was the fact that the more non-linear

models, i.e. lightGBM and MLP, did not outperform a simple elastic net model on data from which the wind farm median had been extracted.

The theory states that the relation between different temperature and context signals is highly non-linear. This is most definitely the case during transient behavior, e.g. start-ups and shutdowns of wind turbines. During start-ups, many drivetrain components slowly heat up. During this process, the relation between the context signals (e.g. active power, rotor speed, ...) and temperatures should be non-linear. So why did the fifth experiment in Chesterman et al. [2023a] not find any clear evidence that the lightGBM and MLP models outperform the elastic net model? Several factors might have contributed to this result.

Firstly, by aggregating the data to the 1-hour level, a part of the transient characteristics might have been removed from the data. This might explain why more complex models, and models that use lagged values of the signals did not outperform the base elastic net model. Secondly, part of the transient characteristics are also removed by subtracting the wind farm median from the signals. All start-ups and cool-downs that are general to the whole wind farm are removed. Only the transient behavior that is specific to a wind turbine remains in the data. Also, this might explain why the more complex models do not outperform the less complex ones. These are currently only hypotheses. Future research will investigate them further.

The results for the autoencoder pipeline did show often improvements in the modeling of normal behavior compared to the shallow ML pipeline. This might be surprising given the results in Chesterman et al. [2023a]. There it was found that the more complex models, i.e. RF, lightGBM, and MLP did not clearly outperform the simple elastic net model. It is however interesting to note that the most optimal network structure identified after hyperparameter tuning is a shallow network with only one hidden layer. This confirms to a certain extent the findings in the paper.

There are several reasons why the autoencoder pipeline somewhat outperforms the shallow ML pipeline. Firstly, the autoencoder can be a better modeler even if it is shallow because of a higher flexibility and better tuning. Secondly, the preprocessing was improved based on the lessons learned from the shallow ML preprocessing. Several less useful steps have been replaced with simpler but more robust new ones. The fact that the autoencoder received more training data as input can have played a role, but it will be limited. The amount of data is such that even for the shallow ML pipeline it was observed that giving more training data to the models had only a limited effect.

6.2 Limited added value of using signal lags or models that can model the time-dependencies

A second surprising result is the fact that there is little evidence that adding lagged signal values to the predictors has added value. The elastic net model with up to three lagged values used in Chesterman et al. [2023a] did not improve the modeling accuracy compared to the model with no lagged values. As already indicated in the previous section, this can most likely be attributed to the aggregation of the data to the 1-hour level and the subtraction of the wind farm median from the signals.

Some testing has been done with LSTM-autoencoders on data from which the wind farm median has not been subtracted but was aggregated to the 1-hour level. The preliminary results did not show any performance improvements compared to the standard autoencoder. This indicates that the aggregation of the data indeed removes most of the non-linear transient behavior from the data. Experiments with LSTM-autoencoders on 10-minute data are a topic for future research.

6.3 Some evidence for inter-turbine differences in the prediction errors

An analysis of the prediction errors shows that there are small differences in normal behavior between the different wind turbines of the wind farm. This shows itself by the fact that for some wind turbines, the prediction error is shifted slightly upwards or downwards. Figure 3.15 shows for WF1 the average steady-state reconstruction errors on healthy data for the different signals. The differences between the wind turbines need to be taken into account during the anomaly detection step. If not this will result in too many or too few anomalies being found for certain wind turbines. This problem can be solved by linear transforming the prediction error (adding or subtracting a quantity from the error to center it around 0), or by using an anomaly detection algorithm that cancels the upward or downward shift. The latter technique will be used in the anomaly detection section.

The reason behind these inter-turbine differences can be found in the fact that wind turbines, just like other machinery, tend to behave slightly differently, even if they are operating normally. This can be explained by small differences in the production or maintenance process. The reason why these differences are visible in the prediction error is that the autoencoder pipeline combines all data from the different wind turbines as if it were a single wind turbine. The predictions it makes are the averages over the different wind turbines. This means that wind turbines that run structurally slightly warmer or cooler than average will show a small upward or downward shift in the prediction errors. This phenomenon can be avoided by training a separate model for each wind turbine.

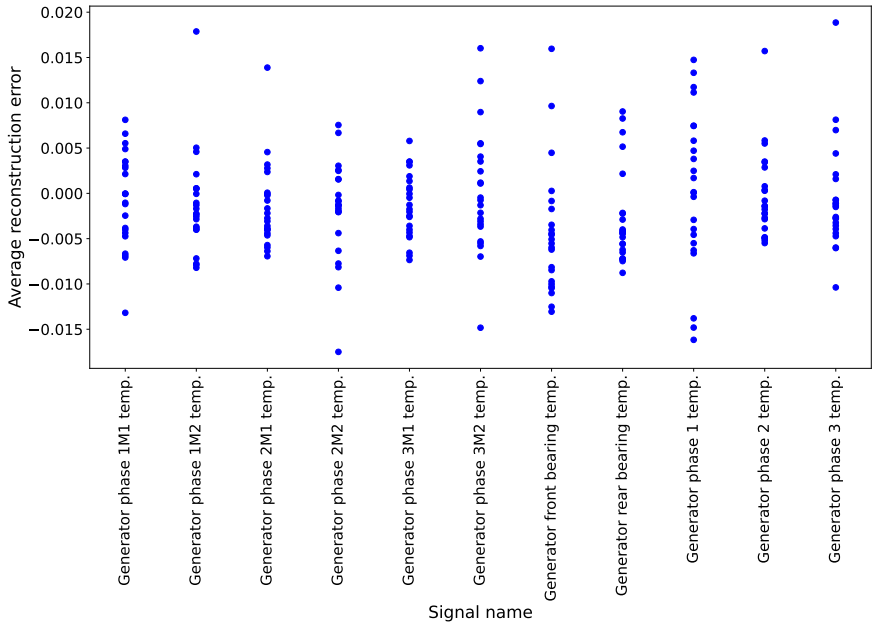


Figure 3.15: Average reconstruction error on healthy data for different signals over different wind turbines. Each blue dot represents a wind turbine.

However, this would reduce the amount of training data for each model drastically, which would cause other problems. It would also increase the waiting time, which is the time it takes to collect sufficient data for the NBM.

6.4 Little evidence for drift over time

A major point of attention is whether the prediction error shows not-damage-related drift over time. If it does this would mean that over time the anomaly rate increases or decreases not because damage starts to build but because the model does not take into account certain time-related changes in the wind turbines. The question is of course what can cause such a drift in the case of wind turbines?

The problem of drift is a gradual loss of fit by the model. This can be for example if over time the normal temperature relations gradually change due to the smoothing of certain production imperfections. If the model is trained on data that comes from the period where the machine was first put into service, then it is not unlikely that the model fit

will reduce gradually over time. To avoid this from happening, a preliminary data analysis was done. Data at the upstart of the wind turbines was discarded if there were clear indications that the signals behaved differently.

However, not all issues could be avoided by a preliminary data analysis. Some issues became only clear after the first modeling results were in and thoroughly discussed with the wind turbine operators. An example is an issue that appeared when the wind turbines of WF1 had to handle reactive power. This increased the temperature in the generator phases. The models were trained on data coming from a period where no reactive power demands were made by the network operator. Because of this, the models underestimated the normal temperatures, which resulted in an elevated false positive rate.

A thorough analysis of the results of the statistical, shallow ML, and autoencoder methodologies showed however that, except for the issue described above, there are no clear indications for drift over time for the WF1, WF2, or WF3 datasets.

6.5 Limited evidence that after a component replacement retraining is required

An analysis of the prediction errors did show however that sometimes after a replacement the NBM model lost fit. In some cases, this could be attributed to a problem with the new component, but this is not always the case. The extent of the problem is fairly limited for the autoencoders. However, it is not unlikely that the replacement of a component can in some cases lead to a loss of fit.

There are three possible solutions to this problem. Firstly, retraining the models each time a component replacement has taken place. This would completely solve the issue, however, it has some disadvantages. After each replacement, it would take some time (several months) before enough data has been collected to train the new model. This is unappealing for an industrial company. Furthermore, it would reduce the amount of training data drastically.

The second solution is modifying the prediction error by adding a correction model to it. This is a simple straightforward solution that requires much less data. However, it still requires a new model. The third solution is using transfer learning. This is a combination of the previous two. It uses a model that has lost its fit or has been trained on another wind turbine as a basis and further trains it to compensate for the deviations. Handling the loss of fit after some replacements is a topic for further research.

6.6 No clear evidence that the autoencoder pipeline models transient behavior worse than steady-state behavior

Under certain conditions, the wind turbine changes its operational state fast and drastically. This is called transient behavior. Modeling this behavior is more complex because the relations between the context variables and the component temperatures are highly non-linear under those conditions. This can be explained by the fact that signals like for example active power change much faster than temperature signals. It takes time for a component to cool down or warm up. This means that time dependencies will also play a role.

However, the fact that the data is aggregated to a 1-hour resolution might diminish their relevance. In this section, it is investigated whether the autoencoder models can model transient behavior well. To this end, the MAE under transient conditions is compared to that under steady-state conditions, by calculating the ratio between the two.

To identify steady-state and transient behavior, a rule-based annotator is developed. This annotator uses a mixture of expert knowledge and data-analysis-based rules. The advantage of a rule-based system is that it is a transparent methodology that can easily be understood. The annotator can classify many different states, e.g. steady-state, transient, full power, full power steady-state, power down, offline, and suspicious power-downs, ... For this research, the most relevant ones are the steady-state and transient identifications.

Determining whether a wind turbine is in steady-state or transient mode is done using a custom definition based on active power. A wind turbine is classified as being in steady-state if the difference between the minimum and maximum of the active power during a 1-hour interval is smaller than 1% of the rated power of the wind turbine. A wind turbine is classified as transient if it is not in steady-state mode.

Figure 3.16 shows the identification of steady-state and transient zones for a specific wind turbine in WF1. The figure shows 15 days of operation. During that time the wind turbine exhibits both steady-state and transient behavior. From the figure, it is clear that the rule-based annotator can detect the steady-state and transient behavior well. The figure uses as example data from only one wind turbine. However, since the rule-based annotator is a simple transparent rule-based system, it is not particularly useful to add extra plots for other wind turbines. The performance would be similar.

The results for the three wind farms indicate that there is no evidence that the autoencoders fit worse on transient than steady-state data. The results show surprisingly the opposite. For WF1, WF2, and WF3 the average transient-vs-steady-state MAE ratios (transient vs. steady-state mean absolute error (TSS-MAE)-ratio) are respectively 0.69, 0.72, and 0.74. For WF1 there is only one signal for which the MAE is larger during transient than steady-state, i.e. the gearbox main bearing temperature. Also for WF2, there is only one signal for which this is true, i.e. the temperature of the rear generator

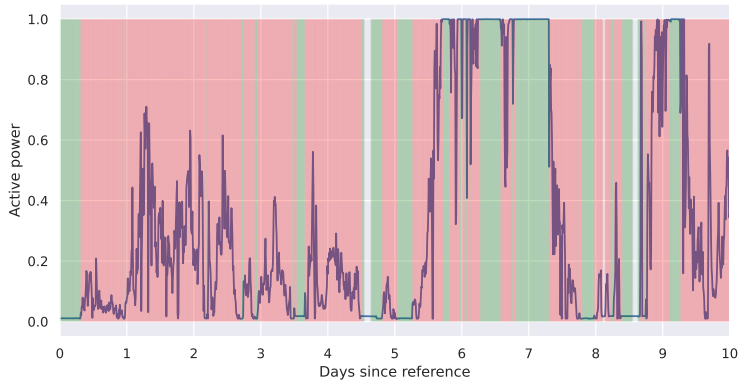


Figure 3.16: The figure shows for several days the active power of the wind turbine (blue curve), the steady-state zones (green), and the transient zones (red) for a wind turbine in WF1.

bearing. For WF3 all signals have a TSS-MAE-ratio smaller than 1. The detailed results can be found in the appendix in Tables 3.4, 3.5, and 3.6.

There are several possible explanations for this phenomenon. Firstly, and most likely, the aggregation of the data to a 1-hour level has removed (most of) the transient characteristics from the data. This would mean that modeling the transient behavior is not more complicated than modeling the steady-state behavior. This is for the moment only a hypothesis based on expert engineering knowledge. Secondly, there is more transient data available for training than non-transient data. This will force the models more towards modeling the transient behavior well compared to the steady-state behavior. This might explain why the performance of the models is better for transient behavior.

Thirdly, it is possible that the worse modeling performance on steady-state behavior may be caused by inter-turbine differences in the component temperatures during the steady-state. It is normal for components of different machines to run on slightly different healthy steady-state temperatures. This can be due to certain production or component quality factors. Due to this, the model will predict during steady-state the average of the temperatures of all wind turbines under the same conditions. Whether the inter-turbine differences play a larger role during steady-state than during transient must be investigated further.

Signal name	Steady-state	Transient
Generator phase 1M1 temp.	0.0070	0.0038
Generator phase 1M2 temp.	0.0063	0.0039
Generator phase 2M1 temp.	0.0051	0.0034
Generator phase 2M2 temp.	0.0074	0.0047
Generator phase 3M1 temp.	0.0058	0.0031
Generator phase 3M2 temp.	0.0124	0.0080
Generator phase 1 temp.	0.0094	0.0054
Generator phase 2 temp.	0.0070	0.0031
Generator phase 3 temp.	0.0067	0.0046
Gearbox A1PREC bearing temp.	0.0135	0.0107
Gearbox A2BPNREC bearing temp.	0.0132	0.0092
Gearbox A2BPREC bearing temp.	0.0069	0.0060
Gearbox BPNREC bearing temp.	0.0109	0.0079
Gearbox main bearing temp.	0.0073	0.0087

Table 3.4: Mean absolute reconstruction error for transient and steady-state behavior for WF1

Signal name	Steady-state	Transient
Generator bearing front temperature	0.0054	0.0033
Generator bearing rear temperature	0.0046	0.0061
Generator cooling water temperature	0.0057	0.0028
Generator phase 1 temp	0.0117	0.0055
Generator phase 2 temp	0.0138	0.0061
Generator phase 3 temp	0.0092	0.0038
Generator slipring temp	0.0046	0.0038
Gear bearing hollow shaft gen. end temp.	0.0060	0.0053
Gear bearing hollow shaft gen. temp.	0.0028	0.0026
Gear bearing hollow shaft middle temp.	0.0092	0.0061
Gear bearing hollow shaft rotor end temp.	0.0036	0.0024
Gear oil L1 temperature	0.0019	0.0018

Table 3.5: Mean absolute reconstruction error for transient and steady-state behavior for WF2

Signal name	Steady-state	Transient
Generator bearing 1 temp.	0.0057	0.0043
Generator phase 1 temp.	0.011	0.0068
Generator phase 2 temp.	0.0109	0.0092
Generator phase 3 temp.	0.0062	0.0041
Generator bearing 2 temp.	0.0057	0.0049
Generator phase 1B temp.	0.0075	0.0073
Generator phase 2B temp.	0.0111	0.0062
Generator phase 3B temp.	0.011	0.0075

Table 3.6: Mean absolute reconstruction error for transient and steady-state behavior for WF3

7 Identifying anomalies in the reconstruction error

In this section, it is discussed how anomalies can be found in the reconstruction errors of the autoencoders. So far a NBM model has been developed that can model the normal or healthy behavior of the wind turbines well. The reconstruction error, which is what the prediction error is called when using autoencoders, is, in general, under healthy conditions small. It is to be expected that when a component is damaged, the behavior of the wind turbine will be less normal, which results in a larger reconstruction error.

The question is of course how large the reconstruction error can be before it is considered abnormal. To this end, an algorithm needs to be used that can analyze it. The literature contains many examples of algorithms that are useful for this. These can be statistics-based, for example, calculating the distribution of the error and determining quantiles, or machine learning-based, which uses certain machine learning algorithms (e.g. Isolation Forest, One-Class SVM, ...) to identify anomalies.

In the course of this research, several anomaly detection algorithms have been tested. These are only a small subset of all possible algorithms. This subset was created by keeping several practical considerations in mind. Firstly, the technique should be unsupervised since there are no labels available that indicate whether data is anomalous or not. Secondly, the amount of data that is required as a minimum to use the algorithm should be as small as possible, so that the waiting time is short. Thirdly, the complexity of the algorithm should be low, so that the overall complexity of the pipeline remains low. Fourthly, the computational demand should be small. Fifthly, the methods should be able to handle the wind turbine-specific differences in the reconstruction errors. For these reasons, the following algorithms were chosen: CUSUM and MAD-IOD.

7.1 Detecting mean-shifts with CUSUM

The CUSUM algorithm was first presented in Page [1954]. It is a technique that is still widely used in the statistical process control (SPC) domain for testing whether a certain process is still within normal operation bounds or not.

The specific CUSUM method that is used is called the 'Tabular CUSUM'. It has only a small number of parameters that need to be set by the user, i.e. K and H. K is usually called the reference value, the allowance, or the slack value [Montgomery, 2009]. As a rule of thumb, K is usually set equal to half the absolute value of the difference between the in-control and the out-of-control mean. H is called the decision interval. If C_i^+ (see Eq. 3.22) or C_i^- (see Eq. 3.23) exceed H then the process is considered out-of-control. A common value for H is 5, which means that a process is considered out-of-control if it deviates from the in-control mean by more than five times the process standard deviation σ [Montgomery, 2009].

The CUSUM algorithm is run separately on each target signal (univariate mode). Multivariate versions of CUSUM exist (see Woodall and Ncube [1985]) and certainly have their merits, however, they will not make it easy to identify which signal is generating the anomalies, which is of interest in this research.

$$C_i^+ = max[0, x_i - (\mu_0 + K) + C_{i-1}^+] \tag{3.22}$$

where:

x_i = Signal value
μ_0 = Target value
K = Reference or slack value
C_{i-1}^+ = Previous one-sided upper CUSUM
C_i^+ = Current one-sided upper CUSUM

$$C_i^- = max[0, (\mu_0 + K) - x_i + C_{i-1}^-] \tag{3.23}$$

where:

x_i = Signal value
μ_0 = Target value
K = Reference or slack value
C_{i-1}^- = Previous one-sided lower CUSUM
C_i^- = Current one-sided lower CUSUM

The target value μ_0 is in the case at hand equal to 0, since it can be assumed that under normal conditions the reconstruction error is around 0. The reference value K is

set to 0.01. This value is determined through trial and error because of the lack of a real out-of-control mean. Both the C_i^+ and the C_i^- are plotted for the analysis of the reconstruction error. This is due to the following observation. In cases where one signal has temperatures higher than expected, it tends to increase the normal predictions for the other signals, which generates in those signals a negative reconstruction error. This means that a failure might show itself as a combination of high C_i^+ and C_i^-.

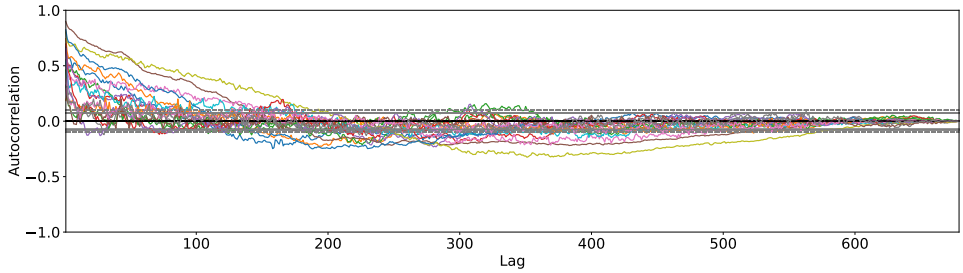
A major challenge when using the CUSUM algorithm on time series is the limitation that the CUSUM assumes that the samples are independent of each other. Bagshaw and Johnson [1975] show that CUSUM is not robust for deviations of the independence assumption and that it has a large impact on the average run length (ARL). Lu and Reynolds [2001] pointed out that the presence of autocorrelation can result in a much higher false alarm rate. Many data signals in the industry are time series that are sampled at a high frequency. This increases the probability of having autocorrelation in the data. This is also the case for the data used in this research. The reconstruction error contains strong autocorrelation.

Several solutions have been suggested to cope with autocorrelation. On the one hand, there are model-based (not to be confused with the definition of model-based methods in Helbing and Ritter [2018]) solutions that focus on modeling the autocorrelation by using autoregressive integrated moving average (ARIMA) models. The SPC techniques are subsequently used on the residuals or the out-of-sample predictions. This is a method that is used in for example Alwan and Roberts [1988], and Chesterman et al. [2021]. They fit first a Box-Jenkins ARIMA model on the process. Next, they use SPC techniques on the out-of-sample predictions. However, model-based solutions also have their challenges. Kovarik et al. [2015] point out that they require an in-control dataset which is not a straightforward requirement in many industrial contexts.

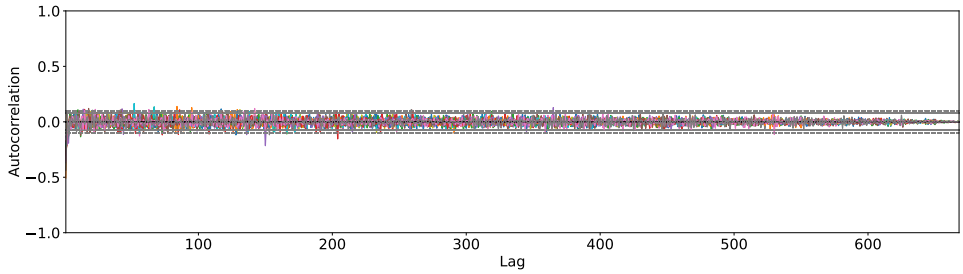
Several techniques have been tried in the course of this research to solve the autocorrelation issue. In Chesterman et al. [2021], it was attempted to remove the autocorrelation by first fitting an ARIMA model to the data. The residuals of the ARIMA were, after some corrections using OLS, used as input for the CUSUM. Although the technique reduced the amount of autocorrelation significantly, it did not sufficiently remove it. The CUSUM thresholds (H) still had to be set at values much larger than the usual value of 5σ to achieve acceptable false positive rates.

Experiments have also been done with CUSUM versions that compensate for the autocorrelation (examples of algorithms can be found for example in Lu and Reynolds [2001] and Orlando O. Atienza and Ang [2002]). The preliminary results were however disappointing, and these techniques were not further explored.

A different approach that has been tried, and which is used in this thesis, is taking the first difference of the reconstruction error (see Ex. 3.24). This technique has a



(a) Before first-differencing.



(b) After first-differencing.

Figure 3.17: Autocorrelation for 700 lags for different signals of a wind turbine of WF1 before and after first-differencing the signals. Each line corresponds to a signal. The figure does not contain a legend since it would clutter the figure, and the exact signal is also not that important.

low computational cost and is straightforward. The technique reduces the amount of autocorrelation significantly. Differencing is often used to make time series stationary. It stabilizes its mean, which eliminates trends and seasonality. There is evidence that the reconstruction errors are non-stationary. This is clear from the autocorrelation plot in Figure 3.17a, where the amount of autocorrelation only decreases slowly. In section 2 it was found that the data contains seasonal and daily patterns. Also, the reconstruction error tends to show certain trends during a prolonged period of time. Both make the time series non-stationarity. The fact that taking the first difference of the reconstruction error removes much of the autocorrelation indicates that most of the autocorrelation in the data is caused by a linear trend.

Since first differencing does not remove seasonal trends, it can be expected that the first difference scheme will not always succeed in removing all the autocorrelation. How successful it is depends on the importance of the seasonal trend. To remove these, the difference needs to be taken between the current observation and the observation x lags back, where x corresponds to the length of the cycle.

Figure 3.17a shows the autocorrelation in different signals of a wind turbine of WF1 before the transformation. The autocorrelation is significant for many lags. Figure 3.17b shows the same signals for the same wind turbine of the same wind farm after the transformation. The autocorrelation is strongly reduced. The result of the transformation is the same for the other wind turbines. With this solution for the autocorrelation, CUSUM could be used for failure prediction with the standard parameter values.

$$\Delta x = x_t - x_{t-1} \tag{3.24}$$

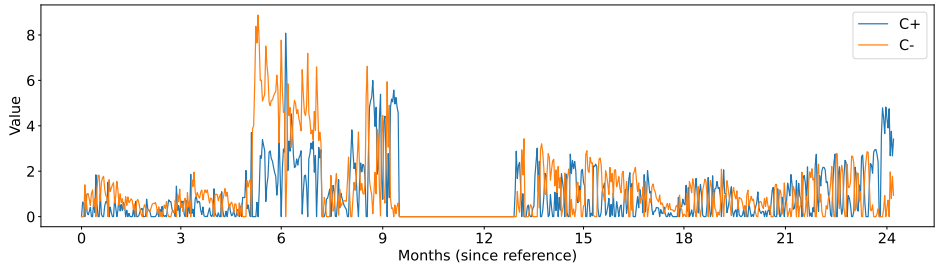
where:

x_t = Value of time series at time t.

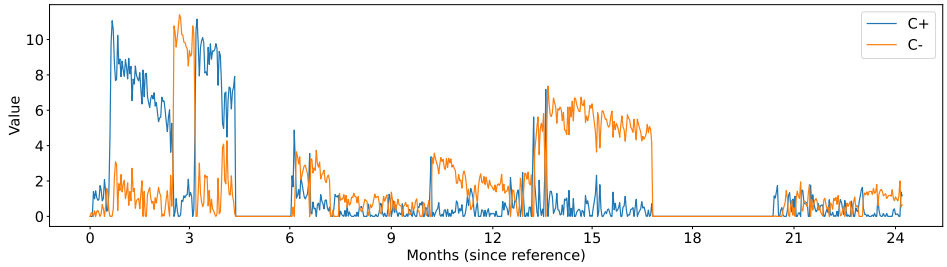
Figures 3.18a and 3.18b show two examples of correct detections of generator failures. The C_+ scores increase strongly a couple of months before the actual failure (the 0-line between months 9 and 13 in figure 3.18a). Also the C_- increases. The fact that both increase indicates that the reconstruction error behaves erratically as the failure approaches.

Figures 3.19a and 3.19b show the results for two wind turbines of WF1 that experienced gearbox failures. The first figure shows the CUSUM scores for the gearbox A1PREC bearing temperature signal. The second figure shows the scores for the gearbox A2BPNREC bearing temperature. In both cases, the failure is detected well in advance.

To quantify the performance of the CUSUM algorithm, the following procedure is used. In general, it is assumed that if the CUSUM score, positive or negative, becomes higher than 5 (5 SD, but since the time series is standardized it is just 5), there is strong evidence

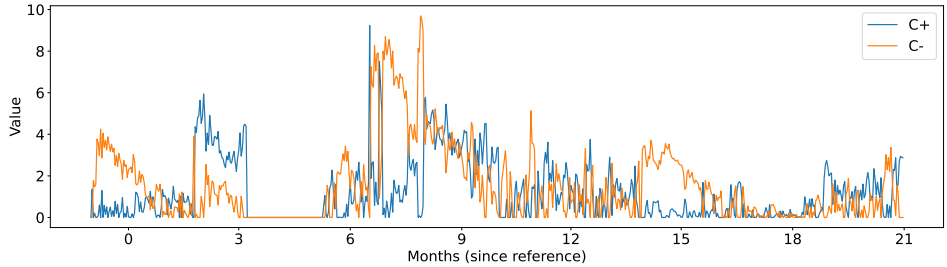


(a) CUSUM values for generator phase 1M1 signal of a wind turbine in WF1. Generator failure occurred just after month 9.

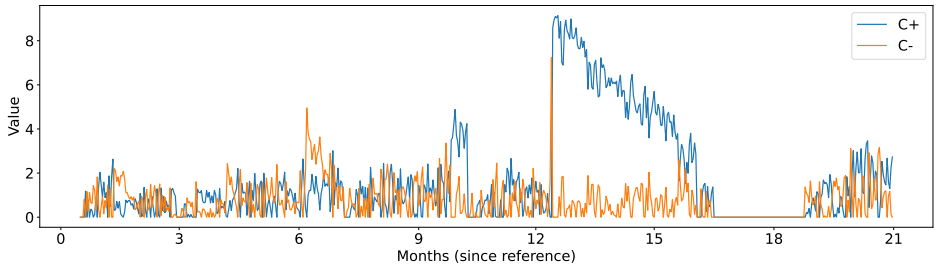


(b) CUSUM values for generator phase 3 signal of a wind turbine in WF1. Generator failure occurred between months 4 and 5.

Figure 3.18: CUSUM values for two generator signals of 2 wind turbines that experienced a generator failure.



(a) Gearbox A1PREC bearing temperature signal of wind turbine. Failure occurred just after the start of month 3.



(b) Gearbox A2BPNREC bearing temperature signal of wind turbine. Failure occurred between months 16 and 17.

Figure 3.19: CUSUM values for 2 gearbox signals of 2 wind turbines that experienced a gearbox failure.

for a change in the mean of the process [Montgomery, 2009]. This standard threshold is also used here. If one of the CUSUM scores exceeds 5 during the three-month period that precedes the failure then it will be assumed that the algorithm has accurately detected the failure. If it only exceeds 5 for a short while during that period, it will be assumed that the algorithm only marginally detected the failure. If during the same period, the CUSUM scores never exceed 5 then it will be assumed that the algorithm has missed the failure. The validation is done on the testing dataset.

On WF1 where there is information on 6 generator and 6 gearbox failures which are well documented and understood, three of the six gearbox failures are identified well in advance (at least a month). Two failures are identified marginally, meaning that the CUSUM algorithm does indicate that something is going wrong, but only weakly. One failure is not identified in advance. For the generator failures, five out of six are identified well. One failure is missed. This indicates that the CUSUM anomaly detection scheme works well for generator failures but is only somewhat mediocre for gearbox failures. A possible explanation is that the autoencoder model is less able to differentiate between unhealthy and healthy behavior due to less relevant information being available (less relevant sensors). However, to be conclusive, further research needs to be done.

Validating the methodology on data from WF2 is somewhat more complicated. This is because this wind farm suffered from severe problems with the generator bearings. Due to this many bearings were replaced preventively during maintenance campaigns. The data does not contain a label on whether the replacement was preventive or an actual failure. This information is nevertheless relevant since it is not unreasonable to assume that the damage on the bearings, and the detectability of the issue in the temperatures, will be more severe for actual failures than for preventive maintenance. The inclusion of preventive maintenance replacements in the validation will tend to underestimate the performance of the methodology. So they are filtered out. This is done using an automatic replacement labeler that is a rule-based classifier (see method 3 in the section "The failure modes of WF2" in Chapter 2). Using the CUSUM algorithm, 83.78% of the failures are accurately detected in advance. This is also a good result on real data.

7.2 Detecting point anomalies with MAD-IOD

The MAD-IOD algorithm is a combination of two statistical algorithms, i.e. the Iterative Outlier Detection (IOD), which is the Iterative Outlier Removal (IOR) algorithm modified for detecting anomalies, and the MAD. MAD is a statistic that has been developed to identify outliers. It is based on the median as a measure of central tendency. The reason for using the median instead of the mean is the fact that it is much less sensitive (more robust) to outliers. A good metric to understand the robustness of the estimator is the breakdown

point. It indicates the maximum proportion of observations that can be contaminated (set to infinity) without resulting in a false value for the estimator (e.g. null or infinity). The breakdown point of the median is 0.5. For comparison, the breakdown point of the mean is 0, which means that even 1 infinite value will give the mean an infinite value [Leys et al., 2013]. This means that all metrics based on the mean (e.g. the standard deviation) will be heavily impacted by outliers. They will tend to inflate the standard deviation which will result in the masking of certain (less extreme) outliers. This should of course be avoided. For this reason, instead of using the standard deviation, MAD is used which is based on the median and which is an approximation of the standard deviation that is not influenced by outliers. Huber [1981] defines MAD as follows:

$$MAD_n = med(|x_i - med_n|) \tag{3.25}$$

where:

x_i = The n time series observations
med_n = The median of the time series.

To make MAD consistent with the normal distribution, it needs to be divided by the 75th percentile of the standard normal distribution ($Q(0.75) = 0.6745$), which corresponds to multiplying it with 1.4826. If the data is not assumed to be normally distributed then it should be divided by $Q(0.75)$ of that distribution [Leys et al., 2013]. This factor will be called k.

IOR, on which IOD is based, is an iterative procedure to remove outliers presented in Parrinello et al. [2016]. The procedure described by the authors makes use of the mean and the standard deviation (SD). The first iteration goes as follows. First, the mean of the time series of the sample is calculated. Next, the difference between the observations and the mean is determined. All observations for which the value is more than 3 SD away from the mean are considered outliers. These points are excluded from the data. Now the next iteration starts. The mean of the time series or sample is calculated without the observations that were considered outliers in the previous iteration. The rest of the iteration goes exactly as the first one. The iterations stop when there are no new points that are more than 3 SD away from the mean.

In Parrinello et al. [2016] the mean and SD are used. Both metrics have a very low breakdown point which makes them sensitive to outliers. This is not ideal, since the algorithm is going to be used for detecting outliers. For this reason, in the IOD algorithm, the mean and SD are replaced by respectively the median and $k\,MAD$. Once the SD has been calculated, three upper bound thresholds are determined, i.e. median + 3 SD, median + 4 SD, and median + 5 SD. This results in respectively the anomaly scores

1, 2, and 3. Next, the moving averages over different window sizes are calculated. These window sizes are 1 day, 10 days, 30 days, 90 days, and 180 days.

The advantage of this algorithm is that it is simple, fast, and robust. Because it uses the median, it also automatically handles structural deviations in the reconstruction error, which might be caused by small differences between the turbines. The results of this algorithm are used in the failure diagnosis step. These results are also used in Chesterman et al. [2023a], and Chesterman et al. [2023b]. The results that are discussed in this section are applied to the autoencoder reconstruction errors.

Of the 6 generator failures 5 are detected clearly and 1 is detected marginally. This is a good result since the generator failure can only be detected indirectly. Of the 6 gearbox failures 4 are detected clearly, 1 is detected marginally, and 1 is missed. This is also a good result for the validation on real data. The performance on the generator failures is similar to that of the CUSUM and slightly better on the gearbox failures.

Figures 3.20 and 3.21 show two examples of correct predictions of a generator failure. This failure type has been discussed in detail in Chapter 2. The actual failure in Figure 3.20 happens shortly after the start of month 9. The failure in Figure 3.21 happens somewhere around month 4 or 5. In the months preceding the failure, the anomaly scores of all window lengths increase dramatically.

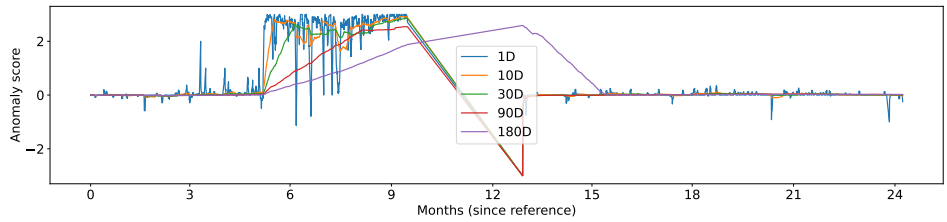


Figure 3.20: MAD-IOD scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for generator phase 2M1 for a wind turbine in WF1. The failure happened just after month 9. Month 0 is just a reference. The exact date to which it corresponds cannot be disclosed.

Figures 3.22 and 3.23 show two examples of where the MAD-IOD scores correctly predict generator bearing failures for several turbines of WF2. As can be seen in the plots, anomaly scores are calculated for different windows, e.g. 1 day, 10 days, 30 days, ... This is done so that different types of anomalies can be identified. Some anomalies form slowly, others fast. By using windows with a different length, there is a bigger chance that at least one of the curves picks up the anomaly. As the failure approaches, the anomaly score starts

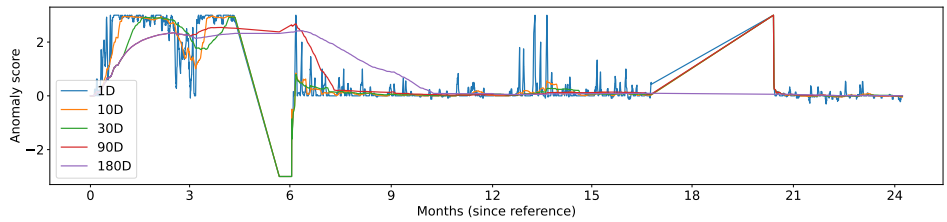


Figure 3.21: MAD-IOD scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for generator phase 3M1 for a wind turbine in WF1. The generator failure happened between months 4 and 5. Month 0 is just a reference. The exact date to which it corresponds cannot be disclosed.

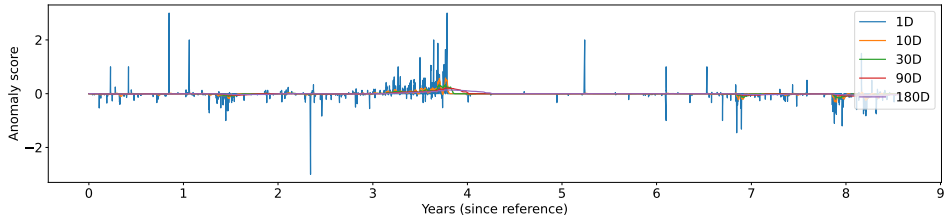


Figure 3.22: MAD-IOD scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for a wind turbine in WF2. The failure happened just before the start of year 4. Year 0 is just a reference. The exact date to which it corresponds cannot be disclosed.

to increase. Interestingly, the score increase starts already a considerable time before the actual failure.

An analysis of the general results of the MAD-IOD on the data of WF2 shows that it can identify 81.08% of the generator bearing failures. The replacements used for the analysis are the same as the ones used for the analysis of the performance of the CUSUM algorithm. The performance is slightly less than for the CUSUM algorithm (the MAD-IOD identifies one failure less than the CUSUM). However, due to the uncertainties of the WF2 data and the good performance of the MAD-IOD on the data for WF1, it is decided to keep on using the MAD-IOD results in the next steps.

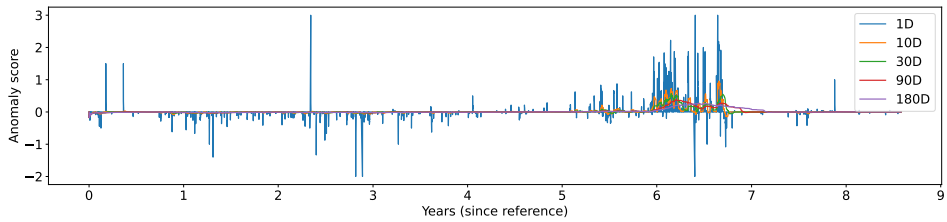


Figure 3.23: MAD-IOD scores for the different window sizes (1 day = blue, 10 days = orange, 30 days = green, 90 days = red, and 180 days = purple) for a wind turbine in WF2. The failure happened just before the start of year 7. Year 0 is just a reference. The exact date to which it corresponds cannot be disclosed.

7.3 Concluding remarks on the identification of anomalies in the reconstruction error

The results from both the CUSUM and MAD-IOD are good given the fact that the validation happens on real data. The MAD-IOD outperforms the CUSUM slightly on the failures of WF1, while it performs slightly worse on the failures of WF2. The robustness of the MAD-IOD is however considerably better than that of the CUSUM. This is because the CUSUM is sensitive to autocorrelation.

This problem is handled by taking the first difference from the reconstruction errors. This results in the removal of the autocorrelation caused by linear trends. This means however that it will not remove autocorrelation created by for example seasonal trends. This is not a major issue for the results based on data from WF1 and WF2. However, analysis done after the submission of the thesis shows that this does not always work sufficiently on data from other wind farms.

The impact of the autocorrelation on the CUSUM shows itself often as constantly increasing trends in the CUSUM scores. This results in a too-high false positive rate which makes the technique unusable. It is important that when one makes an automated system, one can trust that the results are in most cases reliable. This is not the case with the CUSUM algorithm, which makes it less useful than the MAD-IOD.

One aspect of the performance assessment that has not been discussed so far, are the false positives. This is not because the false positives are not considered important by the wind turbine operator, to the contrary. Discussions with the wind turbine operator indicate that minimizing the false positives is more important than maximizing the true positives since each positive starts a whole chain of expensive events, e.g. ordering a

new component, renting a barge with a crane, ... Starting this chain needlessly should be avoided since it would rapidly decrease the confidence of the user in het CM.

There is however a problem with quantifying the number of false positives on real data from wind turbines. The temperature signals are influenced by many events, not just the failure of interest, but also other failures. This means that anomalies that are found in a temperature signal that can not be associated with the failure of interest are not necessarily false positives that tell us something about the performance of the model. These anomalies can be the result of other failure events. In this sense, it is normal that the anomaly detector identifies them.

The issue is caused by the fact that the methodology captures many more problems than the ones covered by the scope of the research. To be able to accurately assess the false positive rate of the methodology in relation to the failures of interest, further analysis needs to be done which requires a thorough analysis of the identified anomalies, and a labeling of them. A classification model/layer needs to be added to the pipeline that labels the anomaly detections automatically. Once this is done an accurate assessment of the false positives can be made. This is the topic for future research. In the meantime, to fulfill the request of the wind turbine operator, conservative anomaly detection thresholds are used. These are set in such a way that the number of anomalies that are found is reduced. This has of course also an impact on the number of true positives. However, for the moment this is the only available solution.

4 | Automated failure diagnosis for wind turbines

The increase in available data has been both a blessing and a curse. On the one hand, it has made it possible to do condition monitoring of wind turbines in much greater detail than before. On the other hand, the increase in the number of signals that can generate anomalies has made it more difficult for humans to interpret the patterns, and understand what is going wrong. A solution to this problem needs to be found. How a failure can be recognized in the data depends on several factors, e.g. the failure type, the location of the sensors, ... Some failure types create similar patterns in the temperature signals. Distinguishing them from each other is only possible by taking other signals and data sources, e.g. status codes into account. However, combining many information sources and making a decision on the failure type that causes the anomalies is not a simple task for humans.

In this chapter, two methodologies are presented for automatic failure diagnosis. Their purpose is to identify the characteristics or properties of several failure types. Both methodologies use data mining (more specifically pattern mining) algorithms to identify suspicious patterns in the data. The first methodology is based on contrast set mining, and the second one is based on frequent association rule mining. There are however differences

in the input sources, the preprocessing, and the postprocessing of the results. This is in part done on purpose to learn which techniques perform best.

The validation data for the two methodologies is also different. The first methodology is tested on data from WF2, while for the second one data from WF1 is used. There are several reasons for this. Firstly, at the time of the methodology design, not all the data was available for validation. Secondly, the data that is used at a particular time also depends on the demands of the industrial partners with whom we collaborate. Thirdly, initially, it was assumed that the failure types in both wind farms were sufficiently similar to use them both as validation cases. However, after analysis of the data and discussions with the wind turbine operator, it appeared that this was not the case and that the generator failures for WF1 could only be detected indirectly in some cases. This complicated of course the analysis and the validation.

Comparing the performance of the two methodologies is made more difficult by three factors. Firstly, the fact that different wind farms (the wind turbine types are not the same) are used for the validation. Secondly, the fact that the failure types are different. Thirdly, it is unclear whether a bearing replacement for WF2 is truly a failure or preventive maintenance.

For this reason, each method will be judged on its own merits and not on how it performs compared to the other method. The validation will also be less quantitative and more qualitative. This means that the focus will be more on how the methodologies behave than on the exact number of correct identifications. These results can be the basis for future research, where a standardized approach is used on many wind farms for which the nature of the failures or replacements is better understood.

This chapter is built up as follows. The first part discusses the background of this research. The focus will be on failure diagnosis research that uses data mining techniques to identify patterns and construct rules. The second part discusses a methodology that uses variants of treatment learning and contrast-set mining, both examples of supervised pattern mining, to identify failure-related fingerprints. These fingerprints are then used as rules in a rule-based classifier. The third part discusses a methodology that uses a combination of frequent association rule mining, ontologies, and clustering to create interpretations of the anomalies that are identified by the autoencoder-MAD-IOD methodology discussed in Chapter 3. This methodology performs failure diagnosis in an unsupervised way. The fourth and last part contains a general conclusion and some final thoughts.

1 Background

The failure diagnosis methodology that is presented in this chapter relies heavily on pattern mining algorithms. The idea is that these algorithms can identify the anomaly and status

code combinations that are specific for a certain failure type. These combinations can be seen as a fingerprint, that ideally shows few similarities with the fingerprint of another failure. An advantage of using patterns is that they can be easily interpreted and compared to expert knowledge. This is not the case for pure *Black Box*-methods.

Pattern mining algorithms have not been used often for failure diagnosis purposes for wind farms. In Liu and Zhang [2020] an extensive overview of wind turbine bearing failure modes and state-of-the-art research is given. Vibration, electrical, microscopic, and lubricant signals have been used for the detection of problems with the gearbox bearings. For generator bearing failures vibration, electrical, and SCADA signals have been used. The application of pattern mining algorithms to failure diagnosis cases for wind turbines is still largely an unexplored area.

The fact that pattern mining algorithms have not been used often for failure diagnosis in wind turbines does not mean that these algorithms have not been thoroughly researched in general. The pattern mining research domain has since its inception in the 90s produced a large variety of different algorithms. These differ in how they prune the search space and the type of patterns they mine, e.g. frequent, rare, high-utility,... Several of these algorithms have been used in research on the condition monitoring of wind turbines.

In what follows an overview is given of several different kinds of pattern mining algorithms, i.e. frequent itemset mining, rare itemset mining, high-utility itemset mining, contrasting itemset mining (or contrast-set mining), and rule-based classification.

1.1 Frequent itemset mining

A first group of research papers is based on algorithms that search for frequent patterns (also called itemsets). To make it more clear what is meant by *frequent* and *pattern* or *itemset* several definitions are introduced. These are based on Goethals [2010]. Let F be a set of items i.

Definition 1.1 (Transaction)**.** A transaction T over F is a tuple (TID, I) with TID a unique transaction identifier and I a set of items $i \in F$.

Definition 1.2 (Transaction database)**.** A transaction database (TXDB) over F is a set of Ts over F with each T having a unique TID.

Definition 1.3 (Supporting)**.** A T supports a set X if X in T, meaning that all elements of X can be found in T.

Definition 1.4 (Cover)**.** The cover of an itemset X in $TXDB$ is the set of Ts that support X. More formally, $cover(X) = \{T : X \in T\}$

Definition 1.5 (Absolute support)**.** The (absolute) support of an itemset *X* is the number of *T*s in its cover. More formally: $sup(X) = |cover(X)|$, with $|...|$ indicating the cardinality of the set *X*.

Definition 1.6 (Relative support or frequency)**.** The frequency of a set *X* is the support of *X* divided by the number of *T*s in the *TXDB*. The frequency is sometimes called the relative support. More formally, $freq(X) = \frac{sup(X)}{|TXDB|}$, with $|...|$ indicating the cardinality of the set *TXDB*.

Definition 1.7 (Frequent itemset)**.** An itemset *X* is frequent if the frequency or relative support of *X* is larger than a minimum frequency threshold t_{min}^{rel}. More formally, *X* is frequent $\iff sup(X) \geq t_{min}^{rel}$.

Frequent itemset mining algorithms look for patterns that fulfill the frequent itemset condition. The user needs to specify a value for t_{min}^{rel}. The lower this value is set, the more patterns that will fulfill the condition. This can quickly result in an explosion of patterns. The search space is exponential in the number of items that occur in the transaction database (TXDB). An important property that is used by most algorithms to prune the search space is the fact that the support of patterns is monotone decreasing with the number of items in the set (see Property 1) [Goethals, 2010].

Property 1 (Support monotonicity): Given a database *TXDB* over F, and two sets $X, Y \subseteq F$. Then $X \subseteq Y \Rightarrow sup(Y) \leq sup(X)$.

This property implies that all supersets of an infrequent itemset are also infrequent, or all subsets of a frequent itemset are also frequent. This is very useful because it can be used to reduce the search space dramatically.

Definition 1.8 (superset)**.** For two itemsets *X* and *Y*, *X* is a superset of *Y* if all items of *Y* are in *X* but not all items of *X* are in *Y*.

Definition 1.9 (subset)**.** For two itemsets *X* and *Y*, *X* is a subset of *Y* if all items of *X* are in *Y* but not all items of *Y* are in *X*.

Several algorithms have been developed to identify frequent itemsets or patterns. One of the first was the Apriori algorithms developed in Agrawal and Srikant [1994]. This algorithm uses breadth-first search to search the search space. Another algorithm that was developed a couple of years later is equivalence class clustering and bottom-up lattice traversal (ECLAT), which was presented in Zaki et al. [1997]. This algorithm used depth-first search to search the search space, which gave it some advantages compared to the Apriori algorithm.

Another important algorithm is FP-growth, which was presented in Han et al. [2004]. This algorithm is used in this thesis. For this reason, it will be discussed in more detail. FP-growth uses just like ECLAT depth-first search. However, it has several new developments that greatly increase the efficiency compared to Apriori. The latter algorithm has two costly steps.

Firstly, the generation of the candidate itemsets. These are itemsets that need to be checked whether they are frequent or not. This set of candidate itemsets can grow very fast. In Han et al. [2004] the example is given for 10,000 frequent itemsets consisting of 1 item (length 1). The set of candidate itemsets of length 2, each consisting of a combination of 2 of the frequent itemsets of length 1, contains 10,000,000 itemsets. Searching for longer itemsets becomes quickly infeasible. Even storing the candidate sets becomes prohibitively expensive. A second problem of the Apriori algorithm is that it requires repeated scanning of the transaction database and matching of the candidate itemsets. This is necessary to calculate the support and the frequency of the itemsets. If the number of itemsets is large, and the transaction database is also large, this is computationally very expensive.

One of the issues of frequent itemset mining is that it tends to generate a large amount of frequent patterns that contain often redundant information. This is because all subsets of frequent itemsets are also frequent. This is due to the support monotonicity property. All this redundancy makes it difficult to analyze the results. A solution is to find a concise representation of these patterns. The most popular representations are according to Fournier-Viger et al. [2017] closed itemset mining, maximal itemset mining, and generator itemset mining. The concepts are defined below. The definitions are based on Fournier-Viger et al. [2017].

Definition 1.10 (closed itemset)**.** Closed itemsets are frequent itemsets that do not have any supersets with the same support. More formally: $CI = \{X | X \in FI \wedge \nexists Y \in FI$ such that $X \subset Y \wedge sup(X) = sup(Y)\}$, with FI the set of frequent itemsets and CI the set of closed itemsets.

Definition 1.11 (maximal itemset)**.** Maximal itemsets are frequent itemsets that do not have supersets that are frequent itemsets. More formally: $MI = \{X | X \in FI \wedge \nexists Y \in FI$ such that $X \subset Y\}$, with FI the set of frequent itemsets and MI the set of maximal itemsets.

Definition 1.12 (generator itemset)**.** Generator itemsets are frequent itemsets that do not have subsets that have the same support. More formally: $GI = \{X | X \in FI \wedge \nexists Y \in FI$ such that $Y \subset X \wedge sup(X) \neq sup(Y)\}$, with FI the set of frequent itemsets and GI the set of generator itemsets.

Most often in the literature closed or maximal itemsets are searched. But even with those representations, it is often necessary to post-process the itemsets because the number of itemsets that are found is simply too large.

There is substantial research that uses frequent itemset mining algorithms for identifying anomalies. These algorithms are often used in conjunction with machine learning-based anomaly detection algorithms for the identification of instances or transactions that contain an abnormal concentration of frequent patterns.

For example, in Feremans et al. [2022] a new pattern-mining-based algorithm is presented that can be used for time series classification. An advantage of this methodology is according to the authors the fact that the results are easily interpretable. The methodology uses symbolic aggregation approximation (SAX) to transform the time series into a symbolic representation. Frequent itemset mining is used to set up a pattern dictionary which in turn is used for creating a pattern-based embedding of the time series. This embedding serves as input for L1- and L2-regularized classification models.

In Feremans et al. [2020] a methodology is presented for mixed-type time series. Frequent pattern mining is used to extract (sequential) patterns from the data. These patterns are filtered by removing the redundant or overlapping ones using the Jaccard similarity metric. The remaining patterns are used to create a pattern-based embedding of the time series. Anomalies can be detected in these time series by using anomaly detection algorithms like for example Isolation Forest,...

In Kuchar and Svatek [2017] patterns are searched in the data using algorithms like for example Apriori and FP-growth. The anomaly score calculations are based on frequent pattern isolation (FPI), which is based on the Isolation Forest algorithm. For each instance or transaction, the data is checked for frequent patterns, and a contribution factor, based on the support and the length of the pattern, is assigned. Also, a penalty term, based on the number of items in the instance that are not part of any frequent pattern, is calculated. The FPI is then calculated as the mean of the contribution and penalty scores.

In He et al. [2014] pattern mining-based anomaly detection is used for network traffic analysis. The network traffic, which is a continuous stream, is handled using a sliding window. A profile is built of the captured traffic using frequent pattern mining algorithms. The profiles, which are mined on the training dataset, are clustered together using k-means clustering. Profiles are also mined on new data that arrives. These new profiles are compared to the ones mined on the training dataset. If they do not match with any of the stored normal profiles an alarm is raised.

In Lin et al. [2010] a methodology is presented that is based on maximal frequent pattern outlier factor (MFPOF), which is a maximal frequent pattern mining algorithm, and an outlier detection algorithm (OODFP). This combined methodology is used for outlier detection in online high-dimensional time series. The results show that the accuracy

of the methodology is slightly lower than for FindFPOF presented in He et al. [2005] (see further). However, the fact that mining maximal frequent itemsets is faster than mining all the frequent itemsets is an advantage. FP-growth is also used for anomaly detection in Zhang et al. [2010], where an algorithm called LFP is presented, and in Tang et al. [2009], where FP-growth is used for detecting anomalies in data streams.

In He et al. [2005] an anomaly detection technique, called frequent pattern outlier factor (FPOF), is presented, which can be used in conjunction with pattern mining algorithms to identify outliers. Frequent patterns are extracted from the data using an association rule mining algorithm. Once the patterns have been identified, the FPOF is used to identify outliers. As a metric, the contradictiveness of a pattern is used. This is a function of its support and length. The top-k outliers are mined. For each outlier, the k frequent patterns with the highest contradictiveness are returned.

1.2 Rare itemset mining

The second group of research papers searches for rare or infrequent itemsets. The underlying assumption is that frequent itemsets correspond to normal behavior, while rare itemsets correspond to abnormal or anomalous behavior. One could interpret rare itemsets as itemsets that have a support that is lower than a user-defined threshold t_{min}^{rel}. However, mining these patterns using frequent itemset mining algorithms would become quickly computationally unfeasible. This is why these algorithms are not really suitable for mining rare itemsets. They require that the minimum support threshold t_{min}^{rel} is set very low, which leads to the generation of large amounts of patterns. This is called the *rare item problem* [Koh and Rountree, 2005]. The number of patterns that are found is one of the main challenges of rare itemset mining [Fournier-Viger et al., 2017].

To keep the number of generated patterns under control, several different techniques and definitions of what a rare pattern is have been developed [Fournier-Viger et al., 2017]. The AprioriInverse algorithm developed in Koh and Rountree [2005] aims at mining *sporadic patterns*. To this end, *perfectly rare itemsets*, which are itemsets that have a support between a minimum and maximum threshold, and no subsets with a support larger than the maximum threshold, are searched. AprioriRare is another algorithm that can be used for rare itemset mining. This algorithm searches for minimal rare itemsets (mRI) [Fournier-Viger et al., 2017]. It was first presented in Szathmary et al. [2007]. The definition of mRI is based on the definition in that paper.

Definition 1.13 (minimal rare itemsets)**.** An itemset is a minimal rare itemset if it is rare but all its proper subsets are frequent.

In Rahman et al. [2016] a new methodology, called rare sequential pattern mining (RSPM), is presented for identifying rare patterns. Rare sequential patterns are identified

using sequential generators. The technique is used for the detection of anomalies in SCADA logs of intelligent electronic devices. The results show that the algorithm can find rare patterns and keep them in the correct order.

In Hemalatha et al. [2015] an outlier factor is designed that is based on the ratio of the number of rare itemsets that match a transaction and the total number of rare itemsets.

In Chen and Zhan [2008] a method is developed based on rare linear patterns. The anomaly score or degree of a pattern is based on its support, which is assumed to be lower than the support of normal patterns. The lower the pattern support is, the higher the anomaly degree becomes. A pattern is considered an anomaly if its anomaly degree is larger than a certain minimum threshold. The k patterns with the highest anomaly degrees are considered anomalies.

1.3 High-utility itemset mining

A third group of research focuses on identifying high-utility itemsets. High-utility itemset mining is a relatively new addition to the pattern mining family. It attempts to identify high-utility itemsets or patterns in a quantitative transaction database (QTXDB). This is a type of TXDB that does not simply state which items are part of the transaction, but also the number of times an item is part of the transaction. This is different from the ordinary TXDB, where the transactions are just itemsets, meaning each item can only appear once.

To fully understand the principle behind high-utility itemset mining, several definitions need to be introduced. The definitions are based on Fournier-Viger et al. [2019]. Let there be a set I of all items $I = \{i_1, i_2, ..., i_m\}$.

Definition 1.14 (quantitative transaction database)**.** A quantitative transaction database D is a set of transactions denoted as $D = \{T_0, T_1, ..., T_n\}$, where each transaction T_q is a set of items (i.e. $T_q \subset I$), and has a unique identifier q called its TID. Each item $i \in I$ is associated with a positive number $p(i)$, called its *external utility*. Furthermore, every item i appearing in a transaction T_c has a positive number $q(i, T_c)$, called its *internal utility*.

The utility of an itemset indicates the importance of the itemset in the database. This is calculated using a utility function. A utility measure can be defined as follows [Fournier-Viger et al., 2019].

Definition 1.15 (utility measure)**.** The utility of an item i in a transaction T_c is denoted as $u(i, T_c)$ and is defined as $p(i)q(i, T_c)$. The utility of an itemset X in a transaction T_c is denoted as $u(X, T_c)$ and is defined as $u(X, T_c) = \sum_{i \in X} u(i, T_c)$ if $X \subseteq T_c$. Otherwise $u(X, T_c) = 0$. The utility of itemset X in a database D is denoted as $u(X)$ and defined as $u(X) = \sum_{T_c \in g(X)} u(X, T_c)$, where $g(X)$ is the set of transactions containing X.

The goal is to identify high-utility itemsets or patterns. These can be defined as follows:

Definition 1.16 (high-utility itemset)**.** An itemset X is a high-utility itemset if its utility $u(X)$ is no less than a user-specified minimum utility threshold *minutil* set by the user (i.e. $u(X) \geq minutil$). Otherwise, X is a low-utility itemset.

There is a lot more that can be said about high-utility itemset mining. However, this would lead us too far. Comprehensive overviews of the utility-oriented pattern mining field are given in Gan et al. [2019] and Fournier-Viger et al. [2019].

An example of research where high-utility pattern mining is used for anomaly detection is Gan et al. [2021]. In this paper, a new pattern mining algorithm, called discovery of utility-aware outlier sequential rules (DUOS), is designed to search for utility-oriented patterns. The algorithm searches for patterns that contribute the most to a certain utility threshold or objective function. A utility is assigned to the individual items, which can be used to calculate the overall utility of a rule or sequence. The algorithm searches specifically for sequence patterns that are rare. A pattern is considered infrequent if its support is between a user-defined minimum and maximum bound. A pattern is a high-utility rule if it is infrequent, has a utility higher than a minimal utility threshold, and a confidence higher or equal to a minimum confidence threshold. The results show that the algorithm can detect rare high-utility anomalies in sequences of real datasets.

1.4 Contrasting itemset mining

A fourth group of research makes use of what is called contrast set mining. This type of mining is a special case of association rule mining. It focuses on finding patterns that best discriminate between two or more groups [Webb et al., 2003]. The results can teach us something about the differences between the groups. In principle, it would be possible to augment a TXDB with group labels and learn an association rule miner on top of it. The miner will return antecedents that are linked to certain consequents or groups. However, as pointed out by Bay and Pazzani [1999], this would tell us something about the patterns that coincide often with a certain group, but not about the differences between the groups.

If the user is interested in the differences, it would be difficult to distill them from the association rule results, for several reasons. Firstly, the number of rules that need to be checked is often huge. Secondly, the association rule learner does not enforce consistent contrasts. This means that a rule with a certain antecedent for one group does not necessarily have its counterpart for other groups. This makes it more computationally demanding to compare the supports or confidences of a certain antecedent over the different groups because an extra scan of the TXDB is required. Thirdly, even if the rules can be matched for the different groups, a statistical test of some sort is required to

determine whether the difference in support or confidence is significant [Bay and Pazzani, 1999].

These issues have led to the development of algorithms that are specifically designed to find the differences. Several algorithms have been developed. Two will be discussed in detail since they are the basis for the first methodology presented in this chapter, e.g. search and testing for understandable consistent contrasts (STUCCO), and TAR3.

The STUCCO algorithm was first presented in Bay and Pazzani [1999]. It was an answer to the problems the authors encountered when trying to identify patterns that distinguish groups from each other. In their paper, the authors extend the itemset concept to contrast-sets, which they define as follows:

Definition 1.17 (contrast-set)**.** Let $A_1, A_2, ..., A_k$ be a set of k variables called attributes. Each A_i can take on values from the set $\{V_{i1}, V_{i2}, ..., V_{im}\}$. A contrast-set is a conjunction of attribute-value pairs defined on groups $G_1, G_2, ..., G_m$.

Bay and Pazzani [1999] also defines the concept of contrast-set support:

Definition 1.18 (contrast-set support)**.** The support of a contrast-set with respect to a group G is the percentage of examples in G where the contrast-set is true.

The goal is to find contrast-sets that have a support that is significantly different across the different groups. Bay and Pazzani [1999] describes it formally as follows:

$$\exists i, j : P(contrastset = true|G_i) \neq P(contrastset = true|G_j) \tag{4.1}$$

$$\max_{i,j} |sup(contrastset, G_i) - sup(contrastset, G_j)| \geq mindev \tag{4.2}$$

mindev is a user-defined threshold. If Eq. 4.1 holds then a contrast-set is called significant. If Eq. 4.2 holds then the contrast-set is called large. If both hold then they call it a deviation. The monotonicity property can not be applied when searching for contrast-sets. This means that the number of generated itemsets can potentially be huge. To limit the number of returned itemsets, they use several heuristics to shrink the search space, e.g. minimum deviation size, expected cell frequencies, χ^2 bounds. They also only return the surprising contrast-sets. These are itemsets that are surprising compared to the itemsets that already have been returned. The non-weighted contrast set mining (NWCSM) algorithm presented in the next section is based on the STUCCO methodology.

A second methodology that is a special case of contrast-set mining is called treatment learning. Treatment learning has been used in engineering research in the past. It does the opposite of a classifier. The latter uses collected data and models to assign unseen data

to different categories. The purpose of treatment learning is to do the opposite, namely, take data that has been categorized (or where the truth is known) and try to determine how a classifier has assigned the data to that category.

Treatment learning can be seen as a form of minimal contrast-set association rule learning. The treatments contrast different groups. There are however several differences with contrast-set mining algorithms like STUCCO. Firstly, each class receives a weight, which allows the user to specify which classes are desirable and which are not. Secondly, treatment learning focuses on identifying minimal theories. This is done by quickly pruning unpromising attribute-value combinations [Gay et al., 2010]. In this matter, the *Narrow Funnel Effect* assumption is introduced in Hu [2003]. This assumption is based on the observation that many domains lack complex relationships and that a small number of critical variables control the others in the system.

Several treatment learning algorithms have been developed, e.g. TAR1, TAR2, TAR3, and TAR4. The treatment learning algorithms are sufficiently different from the contrast-set algorithms to require the introduction of several new definitions (these are based on the definitions in Hu [2003]).

Definition 1.19 (treatment)**.** Let $A_1, A_2, ..., A_k$ be a set of k attributes. Each A_i takes on values from the set $\{V_{i1}, V_{i2}, ..., V_{im}\}$. A treatment R_X is a conjunction of attribute-value pairs that have different levels of confidence with respect to each class.

Definition 1.20 (treatment confidence)**.** The confidence of a treatment R_X with respect to a particular class C is the conditional probability of that class given the treatment is true. More formally, $conf(R_x \Rightarrow C) = P(C|R_x)$.

Definition 1.21 (worth)**.** The worth of a dataset D in terms of class distribution is defined as a weighted probability sum across all classes in the dataset. More formally, $worth(D) = \sum_{classes} score(C_i)P(C_i)$.

Definition 1.22 (treatment lift)**.** The lift of a treatment R_X is the ratio of the worth of the treated subset to the worth of the baseline dataset D. More formally, $lift(R_X) = \frac{worth(D \wedge R_x)}{worth(D)}$. $(D \wedge R_x)$ is the treated subset.

Whether a treatment is useful or not is decided using a modified lift metric, which is based on the concept *worth*. The worth of a dataset is determined as the sum of the products of the user-defined class scores and the frequency of the class in the dataset (see Definition 1.21). The lift of a treatment is then calculated as the ratio of the worth of the dataset when the treatment is applied and the worth when it is not applied (see Definition 1.22). The lift score expresses how much the application of a treatment to the dataset improves the distribution of the classes compared to the original dataset.

If the lift of the treatment is larger than 1, it indicates that the application of the treatment to the dataset has improved the distribution of the classes. What does this all mean? First of all, it is important to note that the input data to the algorithm is not formatted as a typical transaction database, but as a dataframe with rows (observations) and columns (attributes). A treated dataset is a dataset that consists only of observations to correspond to the treatment. If the relative support of the classes with a high user-defined score is higher in the treated dataset than in the original dataset, then the lift of the treatment is larger than 1.

The weighted contrast set mining (WCSM) methodology which will be used in the research presented in this thesis is based on TAR3. In the literature, there are not many papers to be found that use treatment learning. Exceptions are Gay et al. [2010] and Chesterman et al. [2023b]. Nevertheless, the methodology has interesting properties. For this reason, it is used here for a test.

1.5 Rule-based classification

The research mentioned above focuses on identifying patterns that have certain characteristics, e.g. frequent, rare, contrasting, or high-utility. A different type of research goes a step further and focuses on the creation of interpretable rule-based classifiers. This is done by first mining rules on the data, and subsequently using these rules in a rule-based classifier.

For example, in Liu et al. [1998] a technique for classification rule mining is developed that integrates the principles of classification rule mining with those of association rule mining. This is done by modifying the association rule miner so that it only mines rules with as head or consequent the target variable. The methodology consists of three steps. In the first step, the continuous attributes are discretized. In the second step, the class association rules (CARs) are mined. In the third and final step, a classifier is built using the CARs. The CARs are sorted based on their precedence. The precedence is determined using the following three rules: x has larger precedence than y if 1) the confidence of x is larger than that of y, 2) in case the confidence of x and y is the same but the support of x is larger than that of y, or 3) in case both the confidence and the support of x and y are the same but x is generated earlier than y. New observations or instances are classified by the rule with the highest precedence that satisfies the case. If not a single rule satisfies it, then the observation is assigned the default case.

In Li et al. [2001] an algorithm called classification based on multiple association rules (CMAR), is presented that uses FP-growth to mine frequent patterns. The classifier itself is based on multiple rules. If all rules assign the same class (for example A) to the instance, then it is classified as class A. However, if there are inconsistencies between the different

rules, the rules are divided into different groups, according to the class they predict. For each group, the strength of the group or the combined effect is calculated. Groups that consist of rules that are highly positively correlated and have large support are considered to have a strong effect. The group strength is measured using a weighted χ^2 measure. The results show that CMAR performs well compared to C4.5 developed in Quinlan [1993] and classification based on associations (CBA).

In Yin and Han [2003] a rule-generation methodology is presented, called predictive rule miner (PRM), which is a modification of the first order inductive learner (FOIL). The methodology consists of three steps: 1) all rules of which the antecedent is a subset of the instance are selected, 2) for each target class the best k rules are selected using the expected accuracy (this is the probability that an instance that satisfies the body of the rule belongs to the class), 3) the class that corresponds best to the instance is decided based on the average expected accuracies of the different classes. Rules are generated using the greedy approach and dynamic programming.

In Liu et al. [2001] a technique is developed that according to the authors solves two problems with association rule mining: 1) association rule mining uses only a single minimum support threshold, which might be problematic in situations with unbalanced class distributions, and 2) classification data contains often a large number of rules which results in a combinatorial explosion in which the rule generator is unable to generate rules with many conditions. The technique presented in the paper solves these problems by using separate minimum support thresholds for each class, and by using decision tree methods (i.e. C4.5 or Naive-Bayes) for the mining of patterns. The classifier is constructed by selecting rules based on their precedence. Other well-known rule-based classification algorithms are for example repeated incremental pruning to produce error reduction (RIPPER) [Cohen, 1995] and C4.5rules [Quinlan, 1993].

1.6 Conclusion

A gap in the failure diagnosis research for wind turbines is the development of purely data-driven techniques that can be used to identify patterns that are associated with certain failure types. This is particularly interesting if the symptoms of a certain failure type are not well understood. The lack of expert knowledge can then be supplanted by data-driven knowledge. If this knowledge would be human-interpretable, this would be even better. Data mining algorithms might be useful for this.

For example, if it would be possible to identify a group of patterns for each failure type, and if these groups are unique, like a fingerprint, then it would be possible to create an automatic failure diagnosis system for wind turbines. So far there are no examples of research that have done this. However, the development of such systems has real

economic relevance, since it would make it possible to use the information in the many available signals more optimally, while at the same time reducing drastically the amount of labor hours that are required for a thorough failure diagnosis.

2 Identifying failure characterizing fingerprints using contrast-set and treatment learning-based algorithms

In this section, a methodology is discussed that can be used to identify patterns that are characteristic of different failure types. The main idea behind the methodology is that pattern mining algorithms that are designed to distinguish two groups from each other, i.e. contrast-set mining and treatment learning, can be used to find patterns that distinguish healthy wind turbine behavior from behavior that is associated with a certain failure type. These patterns can then be considered as a kind of fingerprint of the failure type, which can be used to identify failures in advance using a classifier. Figure 4.1 gives a schematic overview of the methodology.

The methodology is validated on data from WF2. For this wind farm, a long observation period is available. The wind farm experienced many failures of the same type, i.e. generator bearing failures. These are typical mechanical failures for which an analysis of the temperature signals should be well suited. A more detailed analysis of this failure mode can be found in Chapter 2.

It is also important to note that in this research the anomaly scores generated by the autoencoder-MAD-IOD methodology are not used. Instead, raw (but cleaned) data is transformed using SAX (see further). This simplifies the pipeline, and the performance of the pattern mining methodology can be analyzed in isolation of any issues or peculiarities of the autoencoder-MAD-IOD methodology. Once the performance in isolation has been determined, the methodology can be integrated into a pipeline together with the failure prediction framework discussed in the previous chapter, and the combined performance can be assessed. However, this will not be discussed in this thesis and is the topic of future research.

The methodology consists of 4 steps. Firstly, enrichment of the input data. Secondly, preprocessing of the data by transforming it using SAX. Thirdly, identifying contrast-sets and treatments. Fourthly, post-processing of the patterns, and construction of the rule-based classifier that is used for the failure prediction. Each of the steps is discussed in the following subsections.

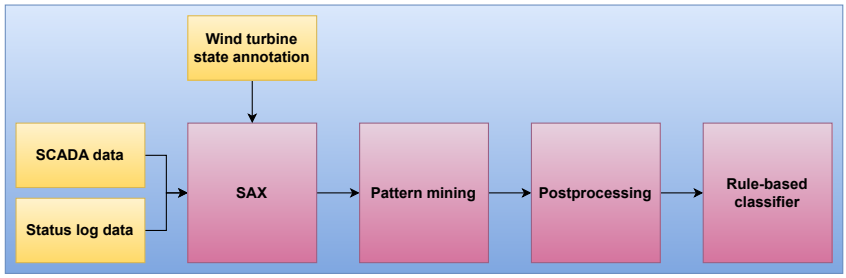


Figure 4.1: Schematic overview of the pipeline of the first methodology consisting of Symbolic Aggregation Approximation (SAX), treatment learning/contrast-set mining, and rule-based classification.

2.1 The methodology and results

Enrichment of the input data with wind turbine operational states

The input data consists of three different information sources: 1) 10-minute SCADA data, which contains component temperatures, and wind turbine context information (e.g. outside temperature, active power, ...), 2) status log data, which contains wind turbine alarms, and 3) generator bearing replacement information, which contains information on the generator bearings (i.e. bearing type, date, wind turbine) that were replaced. From the SCADA data, the following signals are used as input: generator front and rear bearing temperatures, nacelle temperature, wind speed, rotor speed, active power, and generator torque. These were selected based on an extensive literature review (see Chesterman et al. [2023a]).

From the status log data, only the alarms associated with stops and warnings are used. This results in 289 different types of alarms. The data is further enriched with wind turbine state information. These states are deduced from the SCADA data by analyzing the active power behavior of the turbine and the wind farm. A distinction is made between the following states: steady-state, transient, full power, full power steady-state, power down, wind farm power down, offline, suspicious turbine power down, expected turbine power down, expected turbine power up, surprising turbine power up, derated, and maintenance. Table 4.1 gives the definitions of the different states.

State	**Interpretation**
Steady-state	If the active power remains stable during a prolonged period of time (the difference between the maximum and minimum value for a 1-hour window is less than 0.01 times the rated active power).
Transient	If not steady-state.
Full power	If the turbine is producing more than 95% of the rated active power.
Full power steady-state	If the turbine is producing more than 95% of the rated active power and the power production is the same during a prolonged period of time.
Power down	If the turbine is not producing power (less than 5% of the rated active power).
Wind farm power down	If the median of the active power of the whole wind farm is less than 5% of the rated active power.
Offline	If the active power readings are missing.
Suspicious turbine power down	If the turbine is in power down mode, but the farm is not. This indicates that the turbine is not producing any power due to environmental conditions.
Expected turbine power down	If the turbine and the wind farm are both in power down mode. This in general means that there is not sufficient wind.
Expected turbine power up	If the turbine and the wind farm are both not in power down mode.
Surprising turbine power up	If the turbine is not in power down mode, while the wind farm is.
Derated	If the wind turbine produces during a prolonged period of time an active power level that is stable (steady-state) and lies between 5% and 95% of the maximum active power.
Maintenance	If the turbine is for more than two days in power down mode, and during that time the turbine has also been at least part of the time in suspicious power down mode.

Table 4.1: Wind turbine state definitions generated by the wind turbine state annotator.

Transformation of the input data into a symbolic representation using SAX

To transform the data into a symbolic representation, several steps are required. Firstly, the signals in the SCADA data are normalized using wind farm information, e.g. wind farm medians. The wind farm median of each signal is subtracted from the signal. This is done according to the methodology described in Chesterman et al. [2021], and it is the same methodology as the one that is used for the shallow ML anomaly detector (see Chapter 2 and Chapter 3).

Secondly, both the SCADA and status log data are aggregated per day. This makes it possible to remove noise from the temperature signals. The status logs need to have the same resolution before they are merged with the SCADA data. For this reason, they are also aggregated to the 1-day resolution. Thirdly, the generator bearing replacement information, which is made available by the wind turbine operator, is cleaned using the methodology discussed in Chapter 2. This distinguishes true failures from predictive maintenance. Fourthly, the different data sources, i.e. SCADA data, status log data, and wind turbine state information, are merged on the wind turbine and date-time keys.

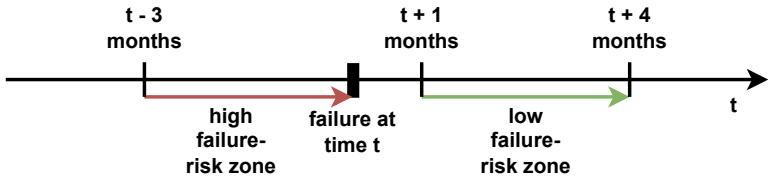


Figure 4.2: Schematic representation of low and high failure risk zones.

Fifthly, the data is divided into failure risk zones. Figure 4.2 shows a schematic representation of how this is done. Data that is generated 1-4 months after a failure is considered low failure-risk data. A run-in time of 1 month is used to make sure that the low failure-risk data is not polluted with start-up and test behavior. By only using data up to 4 months after the failure, the risk is minimized that data is used from components that are again damaged. It is unlikely that a component that has been properly replaced shows already signs of damage four months after the installation. The three months that precede a failure are considered high failure-risk data. Generator bearing damage forms in general slowly over time. By taking the three months of data right before the failure, the data will likely show signs of bearing damage. By dividing the data like this, the failure risk difference is maximized, which we expect to help the pattern mining algorithms find the relevant patterns or rules.

Sixthly, all observations that coincide with a power-down state of the wind turbines are removed. It has been shown in the literature (see Takanashi et al. [2022]) that the

relation between the different signals changes completely in this state. Furthermore, the research conducted here uses abnormal temperature deviations during operation to identify problems with the bearings. This means that power-down states are not useful. No abnormal temperature deviations can form because the bearings are not moving.

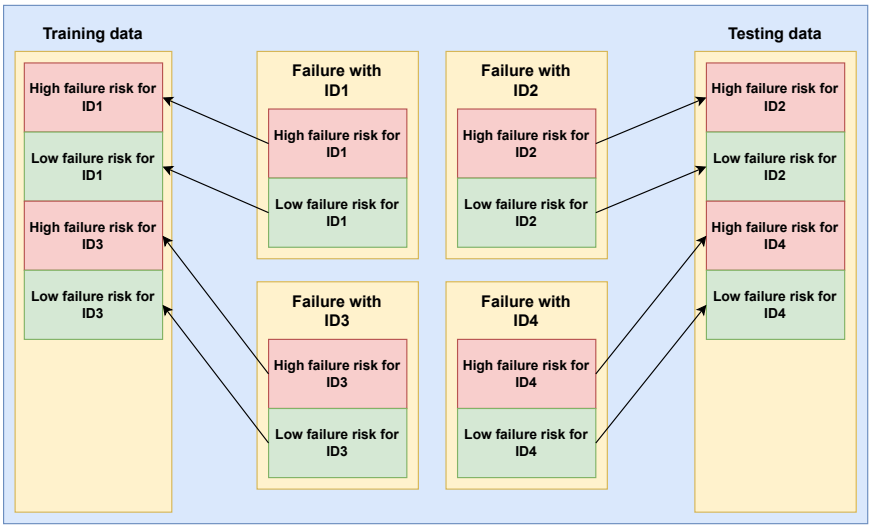


Figure 4.3: Schematic representation of splitting of data in training and testing.

Seventhly, the data is split into a training and testing dataset. Figure 4.3 shows a schematic representation of how this is done. For each replacement type (NDE or DE) there is a separate training and testing dataset. Each replacement is assigned an ID. Half of the IDs are randomly assigned to the training dataset, the other half is assigned to the testing dataset. The high and low failure-risk zones that are associated with the replacement ID are assigned to the respective dataset. This method ensures that the training and testing datasets contain more or less the same amount of data and that the training dataset contains more or less an equal amount of low and high failure-risk data.

The eighth and final preprocessing step is transforming the time series into a symbolic representation using SAX. This technique is presented in among others Papadopoulos and Cipcigan [2009b] and Lin et al. [2007]. It transforms a time series of arbitrary length n into a string of arbitrary length w, with $w << n$. This conversion makes it possible to use techniques developed for example in the bioinformatics research domain. Since the SAX methodology has several steps, and because it plays a crucial role in the pipeline that

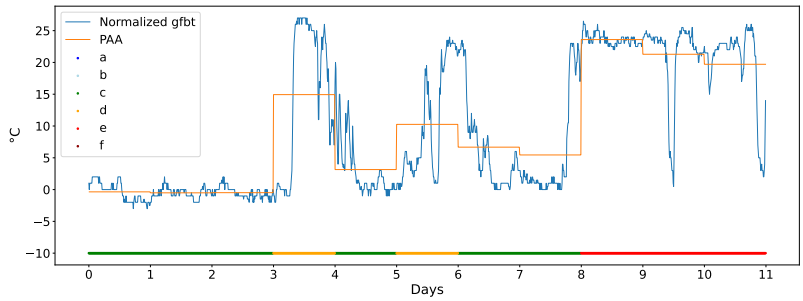


Figure 4.4: Detailed view of several days of data from the front generator bearing temperature sensor of a wind turbine. The temperature signal (blue line) is normalized. The result after application of the PAA step is also shown (orange line). The PAA result is converted into a symbolic representation represented by a color scheme at the bottom of the plot. Green corresponds to "c", yellow to "d" and red to "e".

is presented here, it will be discussed in more detail, using Papadopoulos and Cipcigan [2009b] as a guideline.

The SAX methodology consists of two steps. Firstly, the dimensionality of the problem is reduced using PAA. This means that first the time series is normalized by subtracting the mean from it and dividing it by the standard deviation. In the next step, the time series data is reduced from n dimensions to w dimensions by dividing it into w equal bins and calculating the average of the time series observations in each bin. Eq. 4.3 explains how the reduced time series is calculated. The original time series is represented as $C = \{c_1, c_2, ...c_n\}$, the reduced time series is represented as $\bar{C} = \{\bar{c}_1, \bar{c}_2, ..., \bar{c}_w\}$.

$$\bar{c}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} c_j \tag{4.3}$$

Secondly, the time series is discretized. The goal is to create a symbolic representation where each symbol is equiprobable. The normalized time series have a Gaussian distribution. Based on this distribution breakpoints are determined. These breakpoints are set in such a way that they produce equal-sized areas under the Gaussian distribution. These breakpoints are used to map the PAA values to the alphabet that is used for the symbolic representation. For example, if the PAA value is lower than the first breakpoint, then the value is mapped to "a", if the value is between the first and second breakpoint, it is mapped to "b", ... Figure 4.4 visualizes the impact of each SAX transformation step.

After preprocessing the data, the training and testing data both have a tabular format. The training data for DE failures consists of 2967 instances (observations) and 306 columns. The testing data has 2895 instances. For the NDE failures, the training dataset contains 767 instances and 306 columns, and the testing dataset contains 722 instances. The fact that the number of instances for the NDE failures is significantly smaller than for the DE failures has unfortunately an impact on the performance of the rule-based classifier (see results section).

Identifying treatments and contrasting patterns using treatment learning and contrast-set mining

Once the data has been preprocessed, pattern mining algorithms are used to extract relevant patterns. The goal is to find patterns that best explain the difference between the high and low failure-risk zones. For this, optimized non-weighted and weighted contrast set mining algorithms, called NWCSM and WCSM, which are based respectively on STUCCO [Bay and Pazzani, 1999] and TAR3 [Hu, 2003] are chosen. Since these algorithms are supervised pattern mining algorithms, the patterns are only mined on the training, not on the testing dataset. This is to avoid information leakage, which would artificially inflate the accuracy of the final rule-based classifier on the test dataset.

Both the NWCSM and WCSM can generate rules that have as consequent the high or low failure-risk zone (the *consequent* is the right-hand side of a rule, e.g. a rule of the form $X \implies Y$, which means in words "if X then Y", has as antecedent X and as consequent Y). However, the focus of this research lies on the rules with the high failure-risk zone as consequent. For this reason, the rules with low "failure-risk zone" as consequent are dropped.

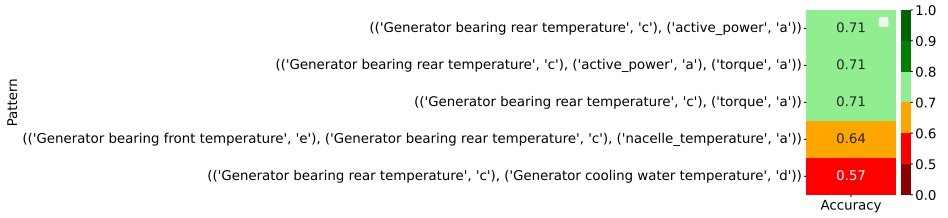


Figure 4.5: WCSM: best rule summary for NDE failures.

The WCSM has been configured to return the 200 rules (with as consequent the high failure-risk zone) with the highest lift. After post-processing, the number of rules is reduced to just 5 for both the DE and NDE generator-bearing failure cases. Figures 4.5 and 4.6 show the results for respectively the NDE and DE generator bearing failures on the test

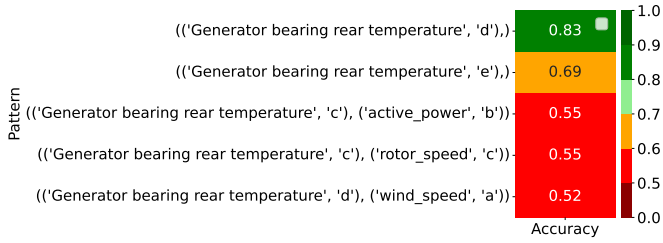


Figure 4.6: WCSM: best rule summary for DE failures.

dataset. The patterns (which are the antecedents or bodies of the rules) are relatively short. The most accurate rules for the NDE case achieve an accuracy of 0.71. The accuracy of the most accurate DE rule is quite high (0.83).

The rules for the DE case are explainable from an engineering point of view. For example, the strongest rules associated with DE failures indicate that the temperature of the rear generator bearing is abnormally high. As has been explained in Chapter 2, these wind turbines suffered from widespread temperature anomalies for the front generator bearing. These mask the failure. This explains why high temperatures in the front generator bearing are not mentioned. However, it was also noted that problems with the DE (or front) bearing can be spotted in the temperatures of the rear generator bearing because both the front and rear generator bearing are part of the same cooling circuit. This is what is shown here in the results.

Figures 4.7 and 4.8 show the results for the post-processed rules identified by the NWCSM. The first thing that can be noted is the fact that these rules are in general longer than those generated by the WCSM. This is because the latter algorithm searches for *Narrow Funnel Effects*, which assumes that the interactions that determine a system are of limited complexity. This results in simpler, but more comprehensible rules. A second thing is that after post-processing more rules from the NWCSM remain (DE: 211 rules, NDE: 44 rules) than is the case for the WCSM (DE: 5 rules, NDE: 5 rules). NWCSM generates more rules with high confidence and relatively high support. Figure 4.7 shows the results for the 20 rules with the best predictive accuracy for the NDE bearing failures. The rules are in general also easily interpretable.

Constructing a rule-based classifier using the contrasting patterns as basis

The next step is designing the rule-based classifier. This classifier is based on the rules that have been identified by NWCSM (DE: 211 rules, NDE: 44 rules) and WCSM (DE: 5 rules, NDE: 5 rules). Using only a single rule for constructing the classifier is suboptimal because

Figure 4.7: NWCSM: top-20 best rule summary for NDE failures.

a lot of information is lost. However, using all the rules is also not useful because it makes the classifier less transparent. Furthermore, some rules may have a high lift score (*lift* is defined as a measure of the prediction performance of an association rule compared to the performance of a random model, see Eq. 4.5) but very low support (*support* is defined as the relative frequency that a pattern occurs in a dataset, see Eq. 4.4). The latter can result in severe overfitting [Cheng et al., 2007], something that was also observed during the analysis of the results for this research.

There are three criteria for determining whether a rule will be used by the classifier or not, i.e. *relevance*, *minimum support*, and *minimum confidence* (confidence expresses the

Model	Bearing type	Minimum confidence	Minimum absolute support
WCSM	**DE**	0.7	10
WCSM	**NDE**	0.7	5
NWCSM	**DE**	0.7	30
NWCSM	**NDE**	0.7	10

Table 4.2: Confidence and support thresholds used for postprocessing of the patterns.

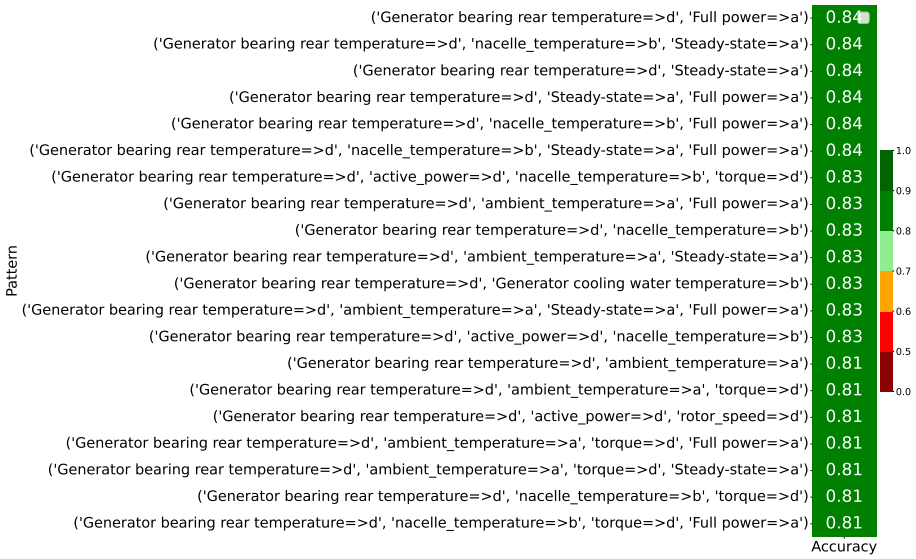


Figure 4.8: NWCSM: top-20 best rule summary for DE failures.

reliability of a rule, or how often the antecedent and consequent occur together compared to the total number of times the antecedent appears in the transactions, see Eq. 4.6). A rule is considered relevant if it contains at least one item in its antecedent that is related to a generator-bearing temperature. An overview of the thresholds for minimum confidence and support can be found in Table 4.2.

$$\text{sup}(X) = \frac{\sharp \text{ transactions containing X}}{\sharp \text{ transactions}} \tag{4.4}$$

where:

$X = \text{A pattern.}$
$\sharp = \text{Count.}$

$$\text{lift} = \frac{P(X \wedge Y)}{P(X)P(Y)} \tag{4.5}$$

where:

X = A pattern. In the case of association rules X is the antecedent of the rule.
Y = A pattern. In the case of association rules Y is the consequent of the rule.
P(.) = Probability function.
$\wedge$ = AND.

$$\text{conf}(R) = \frac{\sharp\ (\text{transaction containing } R_{\text{ante}} \wedge \text{transaction containing } R_{\text{cons}})}{\sharp\ \text{transaction containing } R_{\text{ante}}} \tag{4.6}$$

where:

R = Association rule.
$\sharp$ = Count.
conf(R) = Confidence of rule R.
R_{ante} = Antecedent or body of rule R.
R_{cons} = Consequent or head of rule R.
$\wedge$ = AND.

The rule-based classifier uses multiple rules. This means that multiple rules may indicate at the same time that an instance (or transaction) is a high failure risk. A good classifier can quantify the difference between instances where only a few rules apply, and those where many rules apply. The latter situation should get a higher score than the former since it is more likely that it is associated with a high failure-risk zone. For this reason, the classifier is designed as a confidence-weighted voter combined with an anomaly threshold. If a rule is found in an entry or transaction of the training or testing dataset, then this rule votes that it is part of a high failure-risk zone. However, not all rules are equal. Some rules have a higher confidence than others. To take this into account, each rule has a number of votes that equals its confidence. This guarantees that the rules with the highest confidence can give the most votes. Once each rule has voted, the sum of the votes is taken (Eq. 4.7). If the sum is larger than the anomaly threshold, then the instance is classified as a high failure-risk zone.

$$\text{votes}_{\text{hfr}}(X) = \text{conf}(R_1)\ I_{R_{1\,\text{ante}} \in X} + \ldots + \text{conf}(R_n)\ I_{R_{n\,\text{ante}} \in X} \tag{4.7}$$

where:

$\text{votes}_{\text{hfr}}(X)$ = Sum of votes for high failure-risk zone for entry or transaction X.
$\text{conf}(R_i)$ = Confidence of rule i, with i $\in$ 1,...,n.
$I_{R_{i\,\text{ante}} \in X}$ = 1 if antecedent of rule i is in transaction X, 0 otherwise, with i $\in$ 1,...,n.

The anomaly threshold is determined by a grid search over sensible threshold values on the training dataset. To determine the accuracy of the classifier, a confusion matrix is

Model type	Failure type	Training			Testing		
		Accuracy	TPR	TNR	Accuracy	TPR	TNR
WCSM	**DE**	0.80	0.74	0.86	0.81	0.89	0.72
WCSM	**NDE**	0.77	0.83	0.71	0.79	0.57	1.00
NWCSM	**DE**	0.81	0.82	0.79	0.77	0.89	0.66
NWCSM	**NDE**	0.64	1.0	0.29	0.62	0.83	0.43

Table 4.3: Accuracy, TPR and TNR results for rule-based classifiers based on weighted or non-weighted contrast set mining algorithms for DE and NDE replacements.

constructed. This implies that the number of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) need to be calculated. However, on real data, this is a non-trivial task. It requires several assumptions. Firstly, on when a component is healthy or damaged, and secondly, when the damage that will cause the failure impacts the data, and when not. For this reason, a couple of definitions are introduced.

- TP = if at least one entry or transaction in the high failure-risk zone has a sum of votes larger than the anomaly threshold.
- TN = if no entries or transactions in the low failure-risk zone have a sum of votes larger than the threshold.
- FP = if at least one entry or transaction in the low failure-risk zone has a sum of votes larger than the threshold.
- FN = if no entries or transactions in the high failure-risk zone have a sum of votes larger than the threshold.

The anomaly threshold is set to the value that maximizes the accuracy of the rule-based classifier on the training dataset. The accuracy is calculated using Eq. 4.8. Due to the way that the training dataset is constructed, it contains the same amount of high failure-risk and low failure-risk zones. This means that accuracy measures that can handle unbalanced data are unnecessary.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4.8}$$

Table 4.3 shows the results for the different classifiers. For the classifier based on the WCSM rules (called further weighted classifier), the accuracy on the DE bearing failures is 0.8. If this is compared to the best rule in Figure 4.6 then we see that the accuracy is slightly lower. For the classifier based on the rules generated by the NWCSM (called further non-weighted classifier), the accuracy is 0.77. This is substantially higher than

the accuracy of the best rule that can be found in Figure 4.8. The weighted classifier applied to the NDE bearing failures results in an accuracy on the test dataset of 0.79. This is also a substantial improvement over the most accurate rule shown in Figure 4.5. For the non-weighted classifier, the accuracy is only 0.62, which is significantly lower than for the best rules in Figure 4.7. The results show that in some cases the combination of the different rules by confidence-weighted voting is beneficial compared to using only a single rule. In other cases, it is not. Possible explanations for this are the following: firstly, the relative simplicity of the rule-based classifier methodology that might be unable to combine the rules efficiently, secondly, the amount of overlap between the different rules, and thirdly, the definition of an TP, TN, FP, or FN.

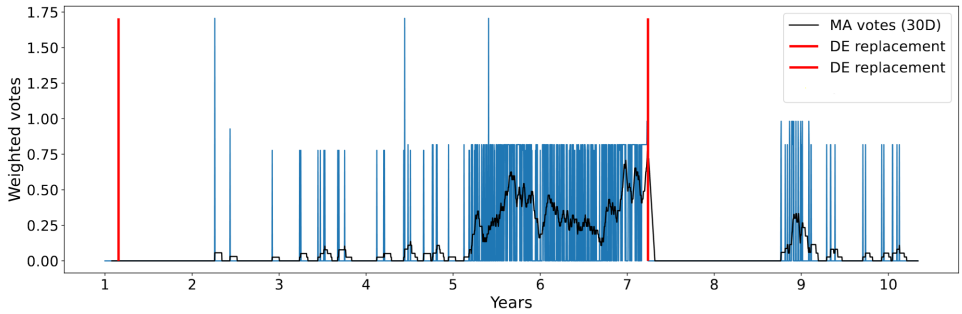


Figure 4.9: WCSM: DE failure prediction.

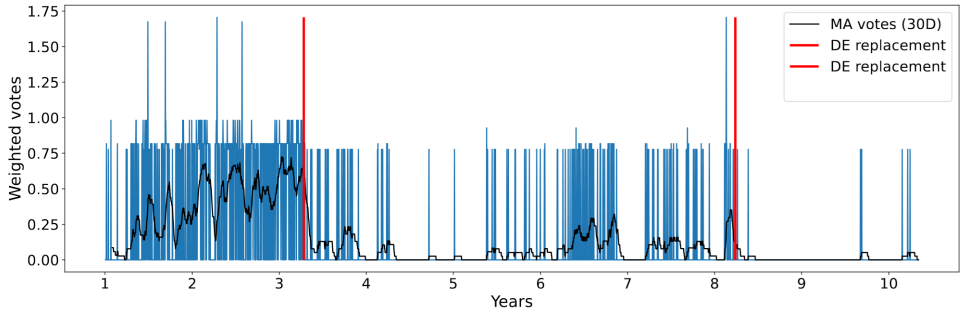


Figure 4.10: WCSM: DE failure prediction.

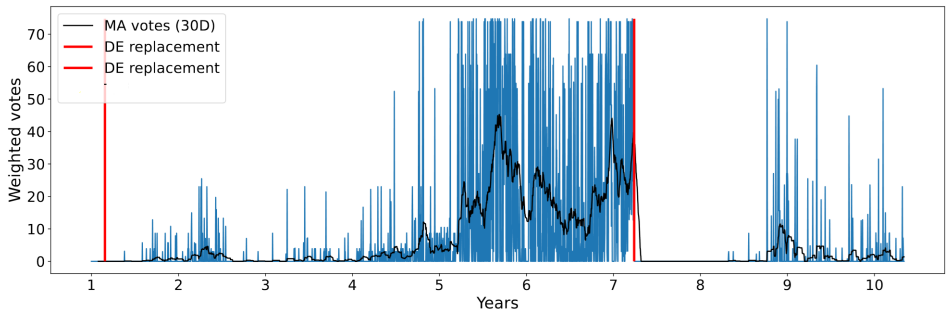


Figure 4.11: NWCSM: DE failure prediction.

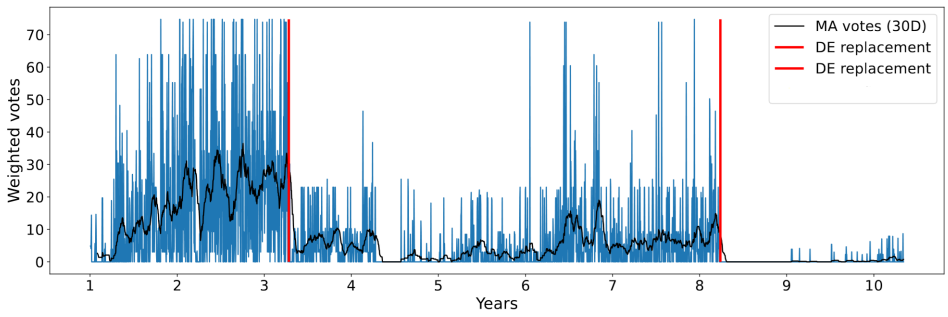


Figure 4.12: NWCSM: DE failure prediction.

Figures 4.9, and 4.10 show several DE generator bearing failure identifications. The moving average of the votes increases until just before the failure. After the failure, when the problem has been fixed, the moving average drops back to 0 or close to it. Figures 4.11 and 4.12 show some of the detections by the non-weighted classifier. The results are for the same turbines as the ones in Figures 4.9 and 4.10. The failure detection pattern is clearer for the non-weighted than for the weighted classifier. For the NDE bearing failures, the results are unfortunately less clear than for the DE bearing failures. This means for example that the increase in the moving average is less pronounced due to only sporadic occurrences of the post-processed rules.

The weaker results for the NDE bearing failures can be attributed to several factors. Firstly, the dataset contains relatively few examples of NDE bearing failures that are clearly separated in time from the DE bearing failures. This means that it is more difficult to distinguish the fingerprint of the NDE bearing failures from those of the DE bearing failures. This probably also explains why the rules for the DE and NDE bearing failures look quite similar. Secondly, the number of observations in the training data for the NDE bearing failures is much lower compared to the DE case. Given the quality of the data, it might be better to treat the DE and NDE bearing failures together instead of separately.

2.2 Some final thoughts on using treatments or contrast-sets as basis for rule-based classifiers

Wind farm median as NBM In this analysis no use is made of an advanced NBM model. Instead, the NBM consists of the wind farm median of the signals. The reason for this has been explained above, namely, the fact that possible imperfections in the NBM modeling process are kept out of the analysis of the performance of the rule-based classifier methodology. Furthermore, it simplifies the pipeline, makes it less computationally demanding, and requires less training data. Furthermore, at the time of the development of the methodology, the autoencoder-based NBM models discussed above were not yet developed. Only results from the shallow ML were available. These models generated under certain conditions unwanted results, which complicate the analysis.

Identifying healthy and unhealthy data The selection of healthy and unhealthy data is also something that is up for discussion. The idea of selecting high-risk and low-risk zones is in principle sound. However, it is doubtful that excluding only one month of data after a replacement from the low-failure risk zone is sufficient. Further data analysis has shown that in some cases the downtime can be significantly longer. This means that some of the low-risk zones may be polluted with atypical behavior. For this reason, it was later decided to increase the number of months that are excluded. These findings were implemented in some of the newer runs of the NBM models (see Chapter 3).

Finding the real failures The procedure for determining whether a failure is a real failure or just a preventive maintenance is also up for discussion. The procedure is sound, however, discussions with the wind turbine operators indicated that a better way to detect real failures is by looking at the amount of downtime before the replacement. This methodology is used for the validation of the autoencoder-based pipeline. The differences between the

two procedures are not massive, but they can also not be completely ignored. The rule-based classifier is likely validated on some predictive maintenance replacements, which puts it somewhat at a disadvantage compared to the autoencoder-based methodologies.

Interpretability The methodology described above creates an automated failure diagnostics tool for wind turbine bearings. One of the main characteristics of the methodology is that the results are easily interpretable because of the usage of pattern mining and rule-based classification. This fact is evident from the results presented above. The treatments and contrast-sets are relatively short and describe well the failure, which makes it for the domain expert easy to understand what is going on. In some cases, it was clear that combining several patterns into a single classifier did not result in improved performance. In those cases using a single best rule is preferable since it is very transparent and easily interpretable.

Combining multiple rules In some cases, it appeared that combining multiple rules has added value. This has the potential to make the interpretation more complex. The top-20 results from the WCSM and NWCSM algorithms did not appear to give clear contradictory information, which is positive, because it indicates that the pattern mining algorithms indeed find useful patterns, and it does not make the interpretation problematic. Nevertheless, interpreting multiple rules at the same time is more difficult than a single rule.

Performance The performance of the WCSM in detecting the DE bearing failures is good and is only slightly lower than the performance of the autoencoder-MAD-IOD and autoencoder-CUSUM algorithms. For the NDE bearing failures, the performance is slightly worse in the case of the WCSM compared to that of the autoencoder-based methodologies, while significantly worse for the NWCSM. This indicates that in some aspects the transparent pattern mining-based methodology, or at least the WCSM, can perform more or less on par with complex black-box deep learning-based algorithms. This is an interesting result since the computational complexity of this methodology is also quite favorable. However, more research should go into what the best way is to combine the different patterns. In the research presented above confidence-weighted voting is used. It is unclear whether confidence is the best choice. Perhaps *lift* would result in better performance. Another question is whether voting is the optimal procedure to combine the rules. This can be the topic of further research.

Clarity of the failure prediction signal The anomaly signal generated by the rule-based classifier looks relatively clean. As the failure approaches, the number of raised anomalies increases clearly. After repairing the problem the number of anomalies reduces dramatically,

which is as expected. This makes the methodology also useful from a user standpoint of view. There is only limited room for doubt on where the problematic behavior appears. There are of course situations where the anomaly signal is less clear.

Final verdict The methodology seems to show potential for anomaly detection, failure diagnosis, and root-cause analysis. It has clear advantages compared to the more complex autoencoder-based methodologies at the cost of perhaps a slightly lower performance. It depends on the user to determine how the advantages and disadvantages weigh up against each other.

3 Failure identification for wind turbines using association rule mining and clustering

In this section, a second data mining-based methodology is presented that can be used for the identification of failures for wind turbines. In this methodology, frequent association rules are mined instead of contrast-sets or treatments. In the previous methodology, the objects of interest were several failures identified by the wind turbine operators. These were large, expensive failures, that occur rarely (meaning there are only a few examples). The low number of occurrences can be challenging. A way to circumvent this problem is to broaden the scope to less expensive failures, that may occur more frequently. Forced shutdowns, which are registered in the status logs of the wind turbines, can be used as proxies for such failures. A forced shutdown of a wind turbine occurs when an error is triggered that is severe enough to require a complete shutdown of the wind turbine. A more detailed discussion of forced shutdowns or outages is given in Chapter 2.

The goal of this research is to find patterns that are associated with forced shutdowns. An interpretation of these patterns can then be used to identify the drivetrain component that has caused the shutdown or failure. The methodology is validated on data from WF1 (see Chapter 2). Currently, the wind turbine operator has a system in place that tries to identify the reason for the shutdown based on the alarms that are visible in the status logs just before the shutdown. This information will be used for the verification of the models. However, it needs to be pointed out that this system is not completely reliable and occasionally it will happen that a wrong cause is assigned to the shutdown. To further validate the methodology, it will also be tested on the severe generator short-circuit failures of WF1 described in Chapter 2. These cases are well documented and have a higher reliability concerning the cause that is attributed to them. However, they are much more rare.

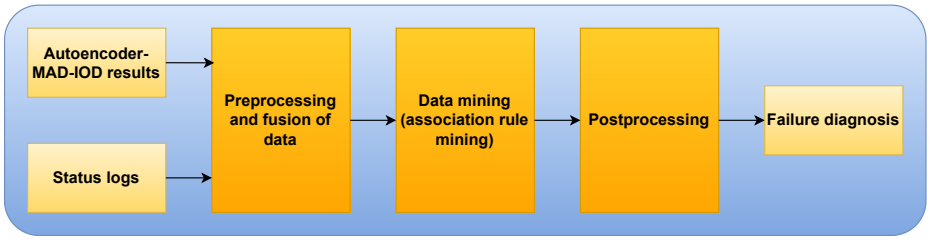


Figure 4.13: Schematic overview of the failure identification methodology using association rule mining and forced shutdowns.

The methodology presented in this chapter is a pipeline consisting of several steps (see Figure 4.13). It builds on the results that are generated by the autoencoder-MAD-IOD pipeline described in the previous chapter. This is different from the previous methodology, where the wind farm median was used as NBM. Even though using the autoencoder-MAD-IOD makes the pipeline more complex and computationally demanding, and introduces some modeling uncertainties, its modeling performance is significantly better than that of the wind farm median NBM. The results from the autoencoder-MAD-IOD pipeline can not be simply used as input for association rule mining. Firstly, the identified anomalies need to be transformed into a format that is suitable for fusion with the data from the status logs. Secondly, association rule mining algorithms are used to identify rules that are associated with forced shutdowns. Thirdly, the identified rules are post-processed to generate a failure diagnosis for the shutdowns.

3.1 Transforming the data into a format suitable for pattern mining

The results from the autoencoder-MAD-IOD pipeline are returned as time series with discrete values between -3 and 3 for windows with lengths of respectively 1 day, 10 days, 30 days, 90 days, and 180 days, for the different signals of the different wind turbines. The status log data is formatted as an event list with for each event a description of what has happened and a datetime. Both formats are not suitable for pattern mining, and they hinder a simple merger of both datasets. Furthermore, the data is, once it is transformed, characterized by a high dimensionality, which increases the computational burden substantially. To solve these two issues, extensive preprocessing is required. This is based in part on several wind turbine ontologies. In the next section, these ontologies are discussed in more detail.

3.2 Wind turbine ontologies: connecting the signals and the status codes to the appropriate system

An important role in this methodology is played by the temperature signal and status code ontologies (see respectively Table 2.8 and Table 2.9) presented in Chapter 2. They describe respectively how the temperature signals and the status codes are linked to the wind turbine components (and sub-components in the case of the temperature signals).

The ontologies are used in this failure diagnosis methodology both during preprocessing and postprocessing. This is unlike for example Domingues and Rezende [2005] and Marinica [2010] where they are only used during postprocessing to make sense of the large amount of rules that are produced by association rule algorithms. The goal of using ontologies here is to construct a hierarchy that resembles the wind turbine structure. The goal is to link the signals and status codes to a specific system or main component of the wind turbine, e.g. generator, gearbox, converter, ... There are several reasons for doing this. Firstly, it allows for the incorporation of technical and expert knowledge into the models. By using a wind turbine ontology, it is possible to add information to the model about the structure of the wind turbine.

Secondly, a failure diagnosis system should be able to give an interpretation of the identified anomalies and status code events, and not simply be returning the anomalies and status codes. The latter presents the operator only with a list of signals showing abnormal behavior and warning or error-related status codes. It does not give an interpretation, which is left to the operator. Interpreting a large amount of data is however not a trivial task for human beings and requires significant investment of time. This is exactly the economic logic that makes an automated failure diagnosis system attractive.

Thirdly, the ontology can be used to reduce the dimensionality of the problem significantly. It makes it possible to combine the different anomaly signals and status codes in a meaningful way by generalizing them. The latter means that the individual signals and status codes are replaced by the wind turbine component to which they are related. Pattern mining can then be performed on these generalizations. This reduces the dimensionality of the problem by reducing the number of unique items, which reduces the computation time significantly. A further reduction of the dimensionality can be achieved during postprocessing.

3.3 Transforming the data into a transaction database

The first preprocessing step transforms the results of the autoencoder-MAD-IOD pipeline into a format that is more suitable for association rule mining. As discussed in the previous chapter, for WF1, there are anomaly scores generated by a generator and a gearbox model.

Both results are used by the failure diagnosis framework. The anomaly scores for moving windows of size 1 day, 10 days, 30 days, 90 days, and 180 days are used. These scores can take up values between -3 and 3. The results of the generator and gearbox models are combined. After this step, the anomaly signals are assigned to a main component using the temperature anomaly ontology (see Table 2.8). This is done separately for each window size, by calculating the mean of the values of the anomaly signals for each datetime timestamp.

$$f(x) = \begin{cases} 1, & \text{if } x \geq q(0.99) \\ 0, & \text{if } q(0.99) > x > q(0.01) \\ -1, & \text{if } x \leq q(0.01) \end{cases} \tag{4.9}$$

The combination of the different anomaly signals via the ontology results in a dataframe containing a column for each main component. The values of the columns are continuous. However, to create items for the transaction database it is better to have discrete values. The discretization is done by using the 0.01 and 0.99 quantiles. If the value of the signal is larger than the 0.99 quantile then it is assigned the value 1. If it is smaller than the 0.01 quantile then it gets the value -1. Otherwise, it gets the value 0 (see Eq. 4.9). The items for the transactions are then created by combining the column name with the recoded value: column_name = recoded_value.

The second preprocessing step transforms the status logs. This is done by transforming the event log into a dataframe with as columns all the unique status codes and as rows the timestamps of the wind turbines. This is accomplished by binning the status logs into 1-hour bins and counting how often a status code was triggered during the bin. Next, the moving sums for each status code for each wind turbine are calculated. This is done for windows of different sizes, i.e. 6 hours, 12 hours, 1 day, and 2 days. This is done to let the algorithm look into the past. In some cases, a status code that has been triggered once or several times several hours before the actual event can be informative. By creating items that look at the number of times a status code occurred in the past hours or days, pattern mining algorithms can include this information.

A point of attention when calculating the occurrences over the different windows is the fact that some status codes tend to be triggered more often and at higher frequencies than others. A naive approach to transforming this information into items would be to simply combine the column name with the occurrence number: column_name = occurrence_number. However, this would cause two problems. Firstly, there would be an explosion in the number of unique items because each different occurrence number would create a new unique item that can be mined. For example "status code x = 10" would be a different item than "status code x = 11", ... One can doubt the relevance of the difference between 10 occurrences and 11 occurrences of a certain status code.

Secondly, the status codes that tend to occur at higher frequencies also tend to have many more different occurrence values. For example, a status code y that has a high maximum trigger frequency, let us say 10 times, will also tend to be triggered in some cases 1, 2, 3, 4, 5, 6, 7, 8, 9 times (some of these numbers may be missing). While status code z which can only be triggered 1 or 2 times due to its nature, will appear only together with occurrence numbers 0, 1, 2. This means that when the naive transformation is used, the number of items related to status code y will be artificially inflated compared to status code z.

To avoid this the occurrence numbers are first recoded. If a status code occurs 0 times in a window it is recoded to 0. If it occurs 1 time it is recoded to 1. If it occurs more often it is recoded to 1+. This will avoid the problems described above. At the same time, it keeps the most important information. It is important to know whether a status code has been triggered or not, and it can also be important to know whether it has been triggered once or several times because this can indicate something about the severity of the problem. The items for the status log transactions are then of the following format: column_name = recoded_occurrence_number.

The third preprocessing step entails the combination or merging of the two transaction databases. Each transaction database consists of transactions represented as sets. The sets are merged using the wind turbine name and datetime as merge keys. The end result is a transaction database containing 353,671 transactions with in total 158 unique items. 158 unique items might seem a lot, but it is much less compared to the situation in which the ontologies and recoding are not used. As a final step, an item is added to all transactions that indicate whether it is part of a forced shutdown zone or not. A forced shutdown zone consists of all transactions that occur during the interval $[t_{fs} - 1\ day, t_{fs}]$, with t_{fs} being the time of the forced shutdown.

3.4 Association rule mining as a way to extract rules that can be associated with forced shutdowns

There are several ways to mine a transaction database as the one constructed above. The goal is that patterns that can be associated with forced shutdowns are found. One way would be to mine frequent (FP-growth) or rare itemsets (AprioriRare) and select only those patterns that contain "forced_shutdown = 1". Another way would be to mine frequent itemsets only on the transactions that contain the item "forced_shutdown = 1". A third way is to mine association rules on the transaction database using FP-growth and force the heads or consequents of the rules to be "forced_shutdown = 1" or "forced_shutdown = 0". The rules with as consequent "forced_shutdown = 1" are then kept. It is the third option that will be followed here.

The advantage of using an FP-growth-based technique is efficiency. This has been explained in the background section of this chapter. The implementation used here comes from the pyFIM library developed by Borgelt. The parameters are set through trial and error, taking into account the computational feasibility, and the minimum absolute support that rules can have. The minimum relative support is set equal to 0.01, the maximum rule length (antecedent + consequent) is set to 6, the minimum confidence is set to 10% and the minimum lift is set to 10.

The reason for using such low confidence together with a high lift is the fact that due to the unbalancedness of the data, using confidence alone can be misleading. The unbalancedness of the data follows from the fact that the forced shutdown zones are rare and contain relatively few observations compared to the complete dataset. This can best be illustrated with an example. Let's say that the shutdown zones consist of 100 observations or transactions. The complete dataset consists of 100,000 transactions, which means that 99,900 transactions are not associated with forced shutdowns.

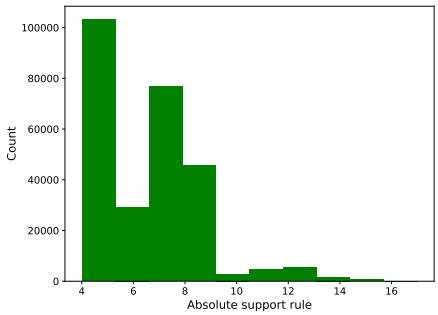
The support of a pattern A is for example 10. Pattern A has an absolute support of 2 in the transactions that are associated with forced shutdowns ($A \rightarrow forced\_shutdown = 1$) and an absolute support of 8 in transactions not associated with forced shutdowns ($A \rightarrow forced\_shutdown = 0$). This means that the confidence of the rule $A \rightarrow forced\_shutdown = 1$ is only 20%. This is low and might indicate that the rule is not interesting. However, this would be misleading, since the A is very rare in the transactions not related to forced shutdowns ($P(A|forced\_shutdown = 0) \approx 0.000008$), while less rare for the transactions related to forced shutdowns ($P(A|forced\_shutdown = 0) = 0.02$). This means that A might actually be very interesting. The lift score is better able to pick up the importance of the rule. The lift score of $A \implies forced\_shutdown = 1$ is approximately 20.02, which is high.

The problem with the confidence metric is also confirmed in Azevedo and Jorge [2007]. There it is pointed out that rules with high confidence are not always useful for the case at hand since such rules may occur by chance. Many different metrics could also have been used instead of lift, e.g. laplacian, conviction, ..., however, lift is the most well-known metric, so this was used.

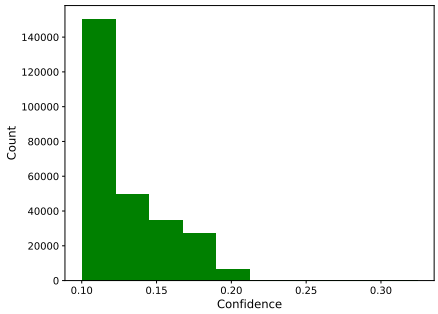
$$sup_{abs} = |\{tx : x \in tx,\ for\ tx \in txdb\}| \tag{4.10}$$

$$sup_{rel} = \frac{sup_{abs}}{|\{tx : tx \in txdb\}|} \tag{4.11}$$

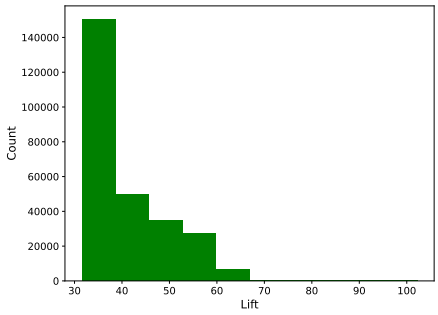
The FP-growth association rule mining algorithm identifies 270,196 association rules. It is always interesting to pay some attention to the properties of these rules. The highest absolute support of the rules is 17. This is low, given that there are 270,196 rules in total.



(a) Absolute support distribution.



(b) Confidence distribution.



(c) Lift distribution.

Figure 4.14: Histograms showing the distribution of the absolute support, confidence, and lift of the rules identified by the FP-Growth association rule miner.

Figure 4.14a shows however that most rules have even lower support. This indicates that most rules are specific (rare) concurrences of events. Itemsets that are so rare are not interesting. However, there is reason to believe that the events that they describe are somewhat less rare than they seem. Several of the rules likely point to a similar physical event, even though the bodies of the rules are not completely the same. This problem will be handled during post-processing.

$$conf(A \rightarrow B) = \frac{sup(A \cup B)}{sup(A)} \tag{4.12}$$

The maximum confidence is 0.32. This is also low. It means that the rule with the highest confidence has only in 32% of the cases a consequent equal to "forced shutdown = 1". Or said otherwise, the antecedent pattern of the rule coincides only in 32% of the cases with a forced shutdown. Figure 4.14c also shows that many rules have a lower confidence. The reason for this is that the minimum confidence parameter of the FP-growth algorithm was set low. This decision was taken to avoid that confidence would be the determining factor for returning a rule or not.

$$lift(A \rightarrow B) = \frac{P(A|B)}{P(B)} = \frac{conf(A \rightarrow B)}{sup(B)} \tag{4.13}$$

To avoid the problem with the confidence metric, the lift score is used instead. The maximum lift score that can be found in the identified rules is 102.23, which is high. Figure **??** shows that most rules have a lift between 10 and 40, but there is also a substantial amount of scores with a higher lift. These scores indicate that for all the returned rules the antecedent and consequent are strongly statistically dependant. This is what we want since it means that the patterns in the antecedent are predictive of forced shutdowns.

270,196 associations rules are of course too much to be usable or interpretable. An analysis of the rules also shows that there is a lot of redundancy in the antecedents. Many antecedents are sub-patterns or super-patterns of each other and are only different in a single item. These differences are not always relevant from a physical or engineering standpoint. In the next section, these characteristics of the rules will be used to reduce their number substantially.

3.5 Going from frequent itemsets to failure diagnosis: making sense of the FP-growth generated patterns.

In this section, the rules that have been identified by the FP-growth association rule miner are analyzed to distill from it more general rules that can be used to perform a failure diagnosis. One of the main issues with the generated rules is that there are too many,

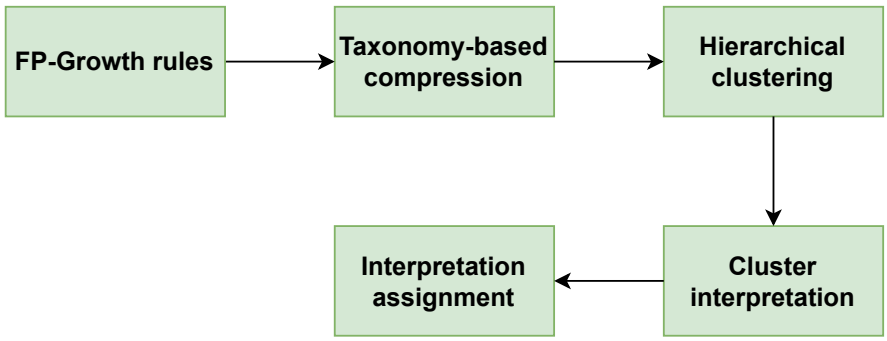


Figure 4.15: Schematic overview of the methodology that transforms the FP-growth association rules into failure diagnosis

and they contain often redundant and overly specific information. This is not useful for a failure diagnosis system. In what follows, a methodology is discussed that uses the wind turbine ontologies and clustering to combine several similar rules into one. Figure 4.15 gives an overview of the different steps.

The first step compresses the identified association rules. The rules that are interesting for the case at hand are those with as consequent "forced shutdown $= 1$", which indicate that the antecedents, which are patterns, are related to forced shutdowns. Since all the remaining consequents are "forced shutdown $= 1$", the only interesting part of the rules is the antecedent. The antecedents indicate which patterns are associated with forced shutdowns. It is in those that we are interested. An analysis of these patterns indicates however that there is a large amount of redundancy from a physical or engineering standpoint of view. A lot of patterns tell more or less the same story. This means that the patterns can be removed by a more high-level pattern that tells the same in just a single pattern.

Identifying these higher-level patterns can be accomplished using the wind turbine ontologies discussed above. They make it possible to link the different items to the different main components of the drivetrain of the wind turbine. A new dataframe is created in which each main component is represented as a column. Each pattern is represented as a row. The values in the row express votes for a main component. The votes are based on the severity of the component problem that the item expresses. A status code warning is 1 vote, a status code stop is 2 votes, a recoded anomaly score of 1 is 2 votes and a recoded anomaly score of -1 is 1 vote.

cluster	water cooling	generator	yaw	pitch	brake	blade	gearbox	converter
1	0.74	0.18	**2.54**	0.94	0.39	0.85	0.56	0.14
2	0.19	0.12	0.87	1.14	0.28	0.83	0.3	**2.48**
3	**4.37**	0.55	0.4	0.98	0.48	0.6	0.0	0.0
4	0.62	0.3	0.91	0.66	0.27	**3.87**	0.15	0.43
5	0.3	**1.69**	1.02	0.76	0.32	0.76	0.49	0.66
6	0.45	0.18	0.58	0.98	**2.33**	0.62	0.24	0.33
7	0.47	0.18	0.55	**4.32**	0.36	0.78	0.29	0.22
8	0.1	0.31	1.0	0.82	0.02	0.76	**4.12**	0.12

Table 4.4: Table shows for each cluster the component score. The higher the score the more linked the cluster is to that specific component issue. The highest score for each cluster is put in bold.

The assignment of votes can be explained as follows. A status code that triggers a stop is more severe than one that just triggers a warning. For the recoded anomaly scores the reasoning is as follows. An anomaly larger than the 0.99 quantile indicates an abnormally high temperature for one of the temperature signals. This can point to a serious problem. An anomaly smaller than the 0.01 quantile, which indicates an abnormally low temperature, can however also point to a problem. This is due to an artifact of the autoencoder. Often abnormally large positive reconstruction errors in certain signals coincide with abnormally large negative reconstruction errors in other signals. This is because the signals with abnormally high temperatures are correlated with the other signals. When those signals exhibit abnormally high temperatures, they pull to a certain extent the estimate of the normal temperatures for the other signals upwards. This creates abnormally large negative reconstruction errors for those signals. This means that the recoded anomaly scores equal to -1 can be useful.

This transformation recodes each pattern as several "failure votes" for each main component. Some patterns will have the same failure vote structure as other patterns. These patterns express the same information. This means that they are redundant. Only one pattern should be kept. By removing the others the pattern database is reduced in size. At the start, it contained 270,196 patterns. This number has now been reduced to 3,341. This is still a considerable amount but is more manageable. It allows for a visual inspection of the rules, and it makes a pattern-matching strategy where the occurrence of each rule in the TXDB is identified more feasible.

The question is what these remaining patterns mean. Which component is likely to fail if they appear? To understand this, the patterns are clustered using hierarchical clustering.

The threshold is set in such a way that there are as many clusters as there are wind turbine components. The clustering happens based on the cosine metric. The interpretation of each cluster is done by calculating the average number of votes per component for each cluster. The component with the largest number of votes is then associated with the cluster. Table 4.4 gives an overview of the average component scores for each cluster. The interpretation of the results of the table is as follows. Cluster 1 is mostly associated with a high score for the yaw system. The scores for the other components are much lower. For cluster 2 the highest score is for the converter. The interpretations for the other clusters are analogous.

Once each cluster has its interpretation, all the patterns in the complete transaction database can be interpreted. Each cluster contains multiple patterns in the compressed database. To identify the cluster that is the best fit for a pattern in the complete transaction database, the dissimilarity (euclidean distance) between the pattern and the patterns in the compressed database is calculated. The average per cluster is computed and the cluster is selected for which the average dissimilarity is the smallest. Table 4.5 shows 10 randomly selected transactions from the transaction database. It also shows the component issue that the methodology associates with it.

These results show that the issue makes sense when one takes into account the items that appear in the transaction. The most interesting examples are those that combine multiple items from different components. In those cases, the algorithm needs to decide to assign it to a certain component. A stop is a more significant event than a warning. This means that if there is a component with a stop and one with a warning, then the issue needs to be attributed to the component with the stop. This is for example the case in the first, seventh, and eighth examples. In all cases, the conflict is resolved correctly.

An interesting way to validate the methodology is to look at the different component scores through time just before a forced shutdown. The focus of this part of the analysis will be on forced shutdowns/outages for which comments from the wind turbine operator on the nature of the failure are available. These often give an indication of which component has been repaired or replaced. If the methodology works properly, the predicted interpretation should be related to the component that is mentioned in the comment. The following figures show that the methodology indeed predicts the issue of several forced shutdowns correctly.

Figure 4.16 shows the component scores that the model assigns to a wind turbine for the 30 days preceding the actual failure. From 5 days before the actual failure onwards, the converter issue score is the highest. The score is around -0.4 or -0.3, which is high. Some converter issues are detected sooner, others later, but in general it is always a couple of days in advance. Figure 4.17 shows the results for a line-side inverter failure (part of

Transactions	**Issue**
Pitch warnings 6H = 1 & Pitch warnings 12H = 1 & Pitch warnings 1D = 1 & Pitch warnings 2D = 1 & Yaw stops 6H = 1 & Yaw stops 12H = 1 & Yaw stops 1D = 1 & Yaw stops 2D = 1+	Yaw
Generator front bearing (90D) = 1	Generator
Yaw stops 2D = 1	Yaw
Generator warnings 1D = 1 & Generator warnings 2D = 1	Generator
Generator phase 2 (1D) = 1 & Generator phase 3 (1D) = 1 & Generator phase 3 (10D) = 1 & Generator phase 3 (30D) = 1 & Generator phase 3 (90D) = 1	Generator
Gearbox warnings 1D = 1 & Gearbox warnings 2D = 1	Gearbox
Generator warnings 2D = 1+ & Blade stops 2D = 1+	Blade
Gearbox warnings 1D = 1 & Gearbox warnings 2D = 1+ & Yaw stops 1D = 1 & Yaw stops 2D = 1	Yaw
Generator phase 2 (1D) = 2 & Generator phase 2 (10D) = 1	Generator
Generator phase 2 (30D) = 1	Generator

Table 4.5: Table shows 10 transactions randomly selected from the transaction database combined with the issue that was assigned to it by the methodology.

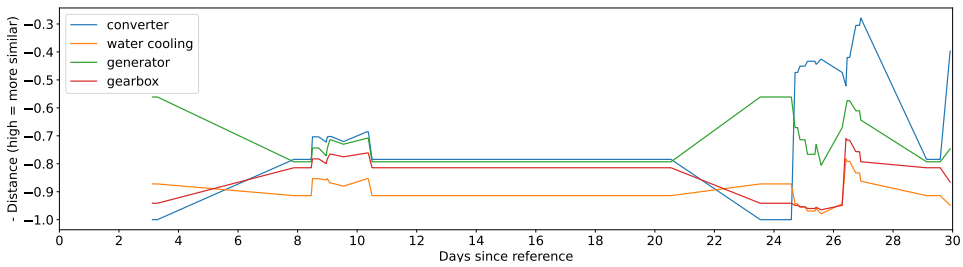


Figure 4.16: Example correct identification of converter failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time.

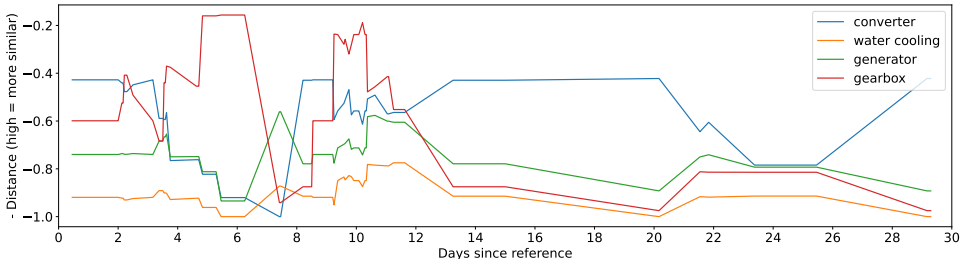


Figure 4.17: Example correct identification of a line-side inverter failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time.

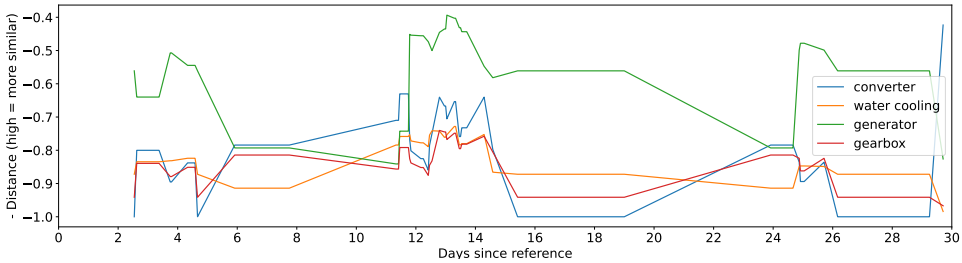


Figure 4.18: Example correct identification of a generator failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time.

the converter). 18 days before the actual failure the converter issue score becomes the highest. This can also be considered a correct detection.

Figure 4.18 shows that during the 28 days preceding the failure, the generator component score is the highest. The failure is related to a problem with the generator phases. One symptom involves the overheating of the phases, which should be observable in the temperature signals. Figure 4.19 shows the results for a turbine that experienced a high shaft current failure. Also here the generator component score is the highest.

A possible remark on the results is that always one issue is the highest. This is indeed correct. It is however important to use this methodology together with a failure detection/prediction methodology like the autoencoder-MAD-IOD. First, the failure prediction methodology should be used to identify anomaly zones. In the next step, these

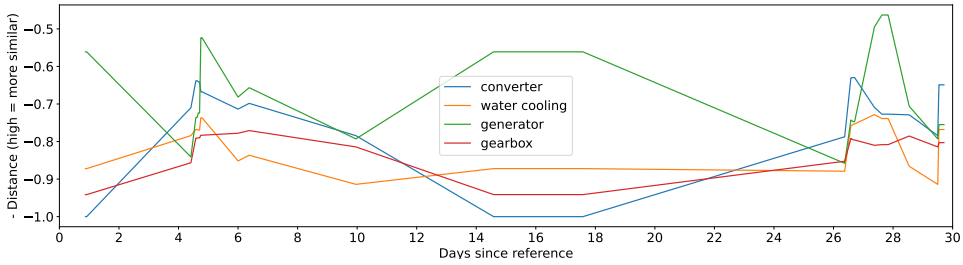


Figure 4.19: Example correct identification of a generator high shaft current failure. The reference day is $t_{shutdown} - 30\ days$. This means that time is counted in days since that reference time.

zones are analyzed by applying the failure diagnosis methodology discussed in this section. The issue with the highest score in the anomaly zone is the failure mode.

3.6 Some final thoughts on using frequent association rule mining, ontologies, and clustering for failure diagnosis.

Although the methodology has potential, since it can interpret some of the shutdowns correctly, it has several drawbacks. Firstly, it has many steps, which makes it rather complex to debug but also to train. Several steps have hyperparameters (e.g. association rule mining, clustering) that need tuning. The parameters have so far been selected by manually exploring the search space. This is not a full-fledged hyperparameter tuning, which means that it is likely that the selected parameters are not optimally tuned. A more automated and structured search of the search space will be developed in future work. Furthermore, the large number of steps also means that the methodology is relatively computationally intensive.

Secondly, the analysis is also not always as clear as in the examples shown above. Sometimes there is some oscillation in the component scores. In those cases, the issue score is not consistently the highest. Using a moving average on top of the scores might solve this problem to a certain extent. Some problems are however not classified correctly. Sometimes the methodology confuses generator and gearbox issues. This can be explained to a certain extent by the fact that these components are located close to each other. Some issues are only identified very late, which makes the detection less relevant. Thirdly, in some cases, a single component score seems to remain consistently the highest for a

prolonged period of time without a clear reason. This indicates that the methodology is prone to bias towards certain components.

However, not all is bad. Using the ontology of a wind turbine has potential for post-processing the association rules. The number of rules can be reduced substantially in a meaningful way. The ontologies used in this research are very small and simple. An interesting research path might be to expand the ontologies. In this way, it should be possible to steer the post-processing. Whether it is also useful to use the ontologies during preprocessing should be investigated further. The reason for using it here also during preprocessing is to reduce the computational cost of pattern mining. The impact of this step on the amount of information that is lost is currently unclear. It is also unclear whether the trade-off between the computational cost and the amount of information is positive.

Using hierarchical clustering to find groups in the compressed pattern database has potential. The groups can be associated with a certain component quite easily. However, selecting the number of groups is non-trivial. For now, the number has been set equal to the number of components on which information is available. It is however unclear whether this is the best solution. Furthermore, the clusters are not always clearly matched to a single component. In some cases this results in a less clear interpretation.

One of the biggest sources of doubt on the methodology is the step where the interpretation is assigned to the patterns in the transaction database. For now, the average Euclidean distance is used, but it is unclear whether this is the best solution. Perhaps a different metric should be more appropriate. Furthermore, selecting the interpretation by taking the one with the smallest dissimilarity is probably also not optimal. Maybe it is better to look at evolutions in the dissimilarity scores, but this would require another layer of analytics, which would make the methodology even more complex.

When comparing this methodology to the methodologies based on NWCSM or WCSM, it seems that the latter methodologies have some clear advantages. They are less complex, less computationally intensive, and less sensitive to biases. Furthermore, their performance seems to be quite good. It needs to be pointed out that those methodologies were only validated on serious wind turbine failures and not on forced shutdowns like the methodology discussed in this section. Even though it seems that the NWCSM or WCSM methodologies seem to have a clear advantage, there are still things that can be learned from the methodology presented here. The idea to post-process the identified patterns using ontologies can be applied to the results or the treatment learning and contrast-set mining. This might be a topic for further research.

5 | Conclusion and future work

The goal of this thesis is the development of a methodology that can be used for failure prediction and diagnosis on industrial machines, i.e. wind turbines, while taking into account several requirements of the industry. Currently, there is a lot of research going on in this domain due to its economic relevance. With the improving accessibility of data from wind turbines, the demand for analysis of this data has also increased. However, searching for patterns in many signals at the same time is difficult and labor-intensive. This problem can be solved through automation.

The first chapter of the thesis is an introduction to the research problem. It starts with a general description of the problem. This is followed by an in-depth discussion of the different parts of a wind turbine, e.g. nacelle, generator, gearbox, ... Next, an overview is given of the most frequently occurring failure types of wind turbines. This is followed by an overview of the typical data that is produced by a wind turbine and the validation techniques that will be used in the thesis. The chapter ends with a discussion of the research questions, the research objectives, the scope, the research strategy, the contributions, and the structure of the thesis.

The second chapter focuses on the properties of the input data. First, the wind turbine types and the characteristics of the wind turbines are presented. This is followed by an overview of the signals in the data and the status codes. Next, a discussion follows of the data quality, and several other properties of the data, e.g. autocorrelation, and spectrum,

... This is followed by an overview of the different failure cases that are used for validation, and a discussion of two ontologies, i.e. a component temperature ontology and a status code ontology. The chapter ends with a conclusion.

The third chapter of the thesis discusses the development of a pipeline that can be used for the detection of anomalies in the SCADA temperature signals. Several pipelines are presented and compared. The pipelines contain two main parts. The first part concerns the prediction of the normal behavior using NBM models. The second part focuses on the detection of anomalies in the prediction error.

For the first part, three different methodologies or pipelines are compared. The first one uses the wind farm median as an NBM. This is used as a benchmark to compare the other solutions. The second methodology uses an ensemble of shallow ML models. This showed in general clear performance improvements over the wind farm median. This came however at the cost of more complexity, higher computational demands, and higher data requirements. The third methodology uses autoencoder models to model the normal behavior. This showed slightly better results than the shallow ML. An extra advantage is that it is faster to train because it models all signals in one go. It did however require more training data than the shallow ML.

In the second part of the chapter two different anomaly detection methodologies are tested, i.e. a CUSUM-based algorithm, and the MAD-IOD algorithm. The validation showed that both algorithms perform well and can detect several different failure types. The MAD-IOD seemed to outperform the CUSUM algorithm slightly on the generator short-circuit failures of WF1, while the CUSUM performed slightly better on the generator-bearing failures of WF2.

The fourth chapter of the thesis discusses the development of a methodology that can be used for automatic failure diagnosis. For this two methodologies are presented that make use of data mining algorithms. The first methodology uses models that are based on treatment learning and contrast-set mining to find patterns that best explain the difference between the healthy and unhealthy wind turbine data. The results showed that the WCSM algorithm (based on treatment learning) can identify rules that can be combined in a classifier to predict failures with an accuracy that is only slightly worse than that of the autoencoder methodology. This is good since the rule-based classifier is much more transparent than the autoencoder. This feature can be used to create failure fingerprints, which are useful for failure diagnosis.

The second methodology uses FP-growth-based association rule mining combined with clustering and wind turbine ontologies to identify different types of forced shutdowns or outages. By using forced shutdowns the problem of having only a few failure examples is circumvented. The wind turbine ontologies are used during pre- and postprocessing to reduce the dimensionality of the data and facilitate the interpretation of the patterns.

The validation of the results showed that the methodology can predict the nature of some forced shutdowns well in advance. This setup has however several disadvantages compared to the first methodology. Firstly, it more often misclassifies forced shutdowns. Secondly, the many steps make it quite complex and difficult to tune. This methodology requires further research to better understand the performance.

The conclusion of this thesis is that some methodologies work well, and some work less well on real data. When taking into account the industry requirements, it seems that a pipeline consisting of the following methodologies is the most interesting for failure prediction and diagnosis. Firstly, for the prediction of the normal behavior of the wind turbine, the autoencoder-based pipeline seems to be the best. Secondly, for the detection of anomalies in the reconstruction error, both the MAD-IOD and the CUSUM seem suitable. Thirdly, for the failure diagnosis, it seems that the methodology that is based on the WCSM algorithm to find patterns that are related to the different failure types is most useful.

There are several paths for future research. Firstly, the methodologies have so far been validated on several different failure types for two different wind turbine types. However, further analysis is required to see whether these results can be confirmed for other failure and wind turbine types. Secondly, there are several parts of the WCSM failure diagnosis methodology that can be improved. One option is to work on the postprocessing of the patterns using the wind turbine ontologies and hierarchical clustering like it was done in the second methodology of Chapter 4. Another possible point of improvement is the combination of several patterns or rules to construct a rule-based classifier. Thirdly, experiments with ontologies have been done in this thesis, however, the ontologies were small. More extensive ones should be constructed, since these would contain more information and will most likely be more useful. Ontologies based on internationally recognized standards like RDS-PP can be interesting to research.

A | Appendices

Signal	Lower bound	Upper bound
Generator phase 1M1 temperature	10.0	170.0
Generator phase 1M2 temperature	10.0	170.0
Generator phase 2M1 temperature	10.0	170.0
Generator phase 2M2 temperature	10.0	170.0
Generator phase 3M1 temperature	10.0	170.0
Generator phase 3M2 temperature	10.0	170.0
Generator phase 1 temperature	10.0	160.0
Generator phase 2 temperature	10.0	160.0
Generator phase 3 temperature	10.0	160.0
Front generator bearing temperature	10.0	80.0
Rear generator bearing temperature	10.0	80.0
Gearbox A1PREC bearing temperature	5.0	80.0
Gearbox A2BPNREC bearing temperature	5.0	80.0
Gearbox A2BPREC bearing temperature	5.0	80.0
Gearbox BPNREC bearing temperature	5.0	80.0
Gearbox main bearing temperature	0.0	80.0
Active power	0.0	*
Generator speed	0.0	*
Nacelle temperature	0.0	*
Outside/ambient temperature	0.0	*
Rotor speed	0.0	*
Torque	0.0	*
Wind speed	0.0	*

Table A.1: Upper and lower measurement error thresholds for WF1. The * indicates that the value has been hidden for confidentiality reasons.

Signal	Lower bound	Upper bound
Generator front bearing temperature	0.5	120.0
Generator rear bearing temperature	0.5	130.0
Generator cooling water temperature	0.5	80.0
Generator phase 1 temperature	0.5	140.0
Generator phase 2 temperature	0.5	140.0
Generator phase 3 temperature	0.5	140.0
Generator slipring temperature	0.5	150.0
Gearbox hollow shaft bearing (gen.) temperature	0.5	85.0
Gearbox hollow shaft bearing temperature	0.5	85.0
Gearbox hollow shaft bearing (middle) temperature	0.5	85.0
Gearbox hollow shaft bearing (rotor) temperature	0.5	85.0
Gearbox oil L1 temperature	0.5	70.0
Gearbox oil temperature basis	0.5	80.0
Active power	0.0	*
Generator speed	0.0	*
Nacelle temperature	0.0	*
Outside/ambient temperature	-5.0	*
Rotor speed	0.0	*
Torque	0.0	*
Wind speed	0.0	*

Table A.2: Upper and lower measurement error thresholds for WF2. The * indicates that the value has been hidden for confidentiality reasons.

Signal	Lower bound	Upper bound
Generator bearing 1 temperature	0.0	70.0
Generator bearing 2 temperature	0.0	80.0
Generator phase 1 temperature	0.0	140.0
Generator phase 2 temperature	0.0	140.0
Generator phase 3 temperature	0.0	140.0
Generator phase 1B temperature	0.0	160.0
Generator phase 2B temperature	0.0	160.0
Generator phase 3B temperature	0.0	160.0
Generator air outlet 1 temperature	0.0	170.0
Generator air outlet 2 temperature	0.0	150.0
Generator lubrication inlet temperature	0.0	50.0
Generator water cooling temperature	0.0	50.0
Active power	0.0	*
Generator speed	0.0	*
Nacelle temperature	0.0	*
Outside/ambient temperature	-5.0	*
Rotor speed	0.0	*
Torque	0.0	*
Wind speed	0.0	*

Table A.3: Upper and lower measurement error thresholds for WF3. The * indicates that the value has been hidden for confidentiality reasons.

Curriculum Vitae

Education

- PhD Computer Science – Artificial Intelligence for Industrial Applications, Sep 2021 – now Vrije Universiteit Brussel – Brussels, Belgium
- Postgraduate Studies in Big Data and Analytics in Business and Management, Jan-June 2020 KU Leuven – Leuven, Belgium
- Data Analysis – Module 8: Data Mining, May-June 2019 Ghent University – Ghent, Belgium
- Master of Science in Statistical Data Analysis (magna cum laude), 2015-2018 Ghent University – Ghent, Belgium Thesis: "The P2Pool mining pool – An Analysis of a Distributed Cryptographically Secured Database"
- The Data Science Toolbox, 2017 Barcelona Graduate School of Economics – Barcelona, Spain
- Bayesian Machine Learning in Social Sciences, 2017 Barcelona Graduate School of Economics – Barcelona, Spain
- Statistical Methods in Risk Management, 2016 London School of Economics – London, United Kingdom
- Factor Models in Time Series with Applications in Macroeconomics and Finance, 2016 London School of Economics – London, United Kingdom
- Master of Science in Economics (cum laude), 2012-2013 Ghent University – Ghent, Belgium Thesis: "Institutional dynamics and economic growth in transition countries. An analysis of the second-round evolutions in post-communistic countries" (econometric analysis)

- Preparatory Course Master of Science in Economics, 2011-2012 Ghent University – Ghent, Belgium
- Master of Arts in History (cum laude), 2010-2011 Ghent University – Ghent, Belgium Thesis: "The development of civil class conscience in 15th century Bruges"
- Bachelor of Arts in History, 2001-2010 Ghent University – Ghent, Belgium

Professional history

- Vrije Universiteit Brussel – Brussels, Belgium, Sep 2021-now PhD researcher: Artificial Intelligence for Industrial Applications
- Picanol NV – Ieper, Belgium, July 2018-Aug 2021 Data Science/Engineering + software development
- Administra Accountancy and Fiscality – Nieuwpoort, Belgium, Oct 2013 – June 2018 Accounting

Teaching experience

WPO Wetenschappelijk Rekenen (4 years), 2021-2024

Master students

Van Der Veken Lotte, KU Leuven Evaluating the prevalence and predictors of metabolically (un)healthy obesity in children: a systematic review and meta-analysis

Journal publications (peer-reviewed)

Chesterman, X., Verstraeten, T., Daems, P-J., Nowe, A. Helsen, J. (2023), Overview of normal behavior modeling approaches for SCADA-based wind turbine condition monitoring demonstrated on data from operational wind farms, Wind Energy Science, 8, 6, p. 893–924.

Conference proceedings (peer-reviewed)

- Chesterman, X., Verstraeten, T., Daems, P-J., Nowe, A. Helsen, J. (2023), Pattern mining based data fusion for wind turbine condition monitoring, Journal of Physics: Conference Series, Vol. 2507, 13 p.

- Chesterman, X., Verstraeten, T., Daems, P-J., Perez Sanjines, F. R., Nowe, A. Helsen, J. (2022), The detection of generator bearing failures on wind turbines using machine learning based anomaly detection, Journal of Physics Conference Series, Vol. 2265, 11 p.
- Chesterman, X., Verstraeten, T., Daems, P-J., Nowe, A. Helsen, J. (2021), Condition Monitoring of Wind Turbines and Extraction of Healthy Training Data Using An Ensemble of Advanced Statistical Anomaly Detection Models, Proceedings of the Annual Conference of the PHM Society. PHM Society, 12 p.

Conference presentations

- Wind Europe 2023, Copenhagen, Denmark, 25-27 April 2023
- TORQUE 2022, Delft, The Netherlands, 1-3 June 2022
- 14th Annual Conference of the Prognostics and Health Management, Nashville, United States of America (Online), 1-4 Nov 2022
- Wind Energy Science Conference (WESC), Online, 25-28 May 2021

Bibliography

Agrawal, R. and R. Srikant
1994. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, P. 487–499, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Alwan, L. C. and H. V. Roberts
1988. Time-series modeling for statistical process control. *Journal of Business Economic Statistics*, 6(1):87–95.

Azevedo, P. J. and A. M. Jorge
2007. Comparing rule measures for predictive association rules. In *Machine Learning: ECML 2007*, J. N. Kok, J. Koronacki, R. L. d. Mantaras, S. Matwin, D. Mladenič, and A. Skowron, eds., Pp. 510–517, Berlin, Heidelberg. Springer Berlin Heidelberg.

Bagshaw, M. and R. A. Johnson
1975. The effect of serial correlation on the performance of cusum tests ii. *Technometrics*, 17(1):73–80.

Bajric, R., N. Zuber, G. Skrimpas, and N. Mijatovic
2015. Feature extraction using discrete wavelet transform for gear fault diagnosis of wind turbine gearbox. *Shock and Vibration*, 2016:1–10.

Bangalore, P., S. Letzgus, D. Karlsson, and M. Patriksson
2017. An artificial neural network-based condition monitoring method for wind turbines, with application to the monitoring of the gearbox. *Wind Energy*, 20(8):1421–1438.

Bangalore, P. and L. B. Tjernberg
2014. Self evolving neural network based algorithm for fault prognosis in wind turbines: A case study. In *2014 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, Pp. 1–6.

Bangalore, P. and L. B. Tjernberg
2015. An artificial neural network approach for early fault detection of gearbox bearings. *IEEE Transactions on Smart Grid*, 6(2):980–987.

Bay, S. and M. Pazzani
1999. Detecting change in categorical data: Mining contrast sets. P. 302–306.

Beretta, M., J. J. Cárdenas, C. Koch, and J. Cusidó
2020. Wind fleet generator fault detection via scada alarms and autoencoders. *Applied Sciences*, 10(23):1–15.

Beretta, M., A. Julian, J. Sepulveda, J. Cusidó, and O. Porro
2021. An ensemble learning solution for predictive maintenance of wind turbines main bearing. *Sensors*, 21(4):1–20.

Bermúdez, K., E. Ortiz-Holguin, C. Tutivén, Y. Vidal, and C. Benalcázar-Parra
2022. Wind turbine main bearing failure prediction using a hybrid neural network. *Journal of Physics: Conference Series*, 2265(3):032090.

Bilendo, F., A. Meyer, H. Badihi, N. Lu, P. Cambron, and B. Jiang
2023. Applications and modeling techniques of wind turbine power curve for wind farms—a review. *Energies*, 16(1).

Black, I. M., D. Cevasco, and A. Kolios
2022. Deep neural network hard parameter multi-task learning for condition monitoring of an offshore wind turbine. *Journal of Physics: Conference Series*, 2265(3):032091.

Black, I. M., M. Richmond, and A. Kolios
2021. Condition monitoring systems: a systematic literature review on machine-learning methods improving offshore-wind turbine operational management. *International Journal of Sustainable Energy*, 40(10):923–946.

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

Brigham, E.
1988. *The Fast Fourier Transform and its Applications*. Prentice-Hall, Inc.

Brockwell, J. and R. Davis
2002. *Introduction to Time Series and Forecasting*. Springer Cham.

Campoverde, L., C. Tutivén, Y. Vidal, and C. Benaláazar-Parra
2022. SCADA data-driven wind turbine main bearing fault prognosis based on principal component analysis. *Journal of Physics: Conference Series*, 2265(3):032107.

Cao, W., Y. Xie, and Z. Tan
2012. Wind turbine generator technologies. In *Advances in Wind Power*, R. Carriveau, ed., chapter 7. Rijeka: IntechOpen.

Carroll, J., A. McDonald, and D. McMillan
2016. Failure rate, repair time and unscheduled o&m cost analysis of offshore wind turbines. *Wind Energy*, 19(6):1107–1119.

Castellani, F., D. Astolfi, and F. Natili
2021. Scada data analysis methods for diagnosis of electrical faults to wind turbine generators. *Applied Sciences*, 11(8):1–14.

Cevasco, D., S. Koukoura, and A. Kolios
2021. Reliability, availability, maintainability data review for the identification of trends in offshore wind energy applications. *Renewable and Sustainable Energy Reviews*, 136:110414.

Chen, B., L. Xie, Y. Li, and B. Gao
2020. Acoustical damage detection of wind turbine yaw system using bayesian network. *Renewable Energy*, 160:1364–1372.

Chen, H., H. Liu, X. Chu, Q. Liu, and D. Xue
2021. Anomaly detection and critical scada parameters identification for wind turbines based on lstm-ae neural network. *Renewable Energy*, 172:829–840.

Chen, T. and C. Guestrin
2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM.

Chen, X. and Y. Zhan
2008. Multi-scale anomaly detection algorithm based on infrequent pattern of time series. *Journal of Computational and Applied Mathematics*, 214:227–237.

Chen, Z., J. M. Guerrero, and F. Blaabjerg
2009. A review of the state of the art of power electronics for wind turbines. *IEEE Transactions on Power Electronics*, 24(8):1859–1875.

Cheng, H., X. Yan, J. Han, and C.-W. Hsu
2007. Discriminative frequent pattern analysis for effective classification. In *2007 IEEE 23rd International Conference on Data Engineering*, Pp. 716–725.

Chesterman, X., T. Verstraeten, P.-J. Daems, A. Nowe, and J. Helsen
2023a. Overview of normal behavior modeling approaches for scada-based wind turbine condition monitoring demonstrated on data from operational wind farms. *Wind Energy Science*, 8(6):893–924.

Chesterman, X., T. Verstraeten, P.-J. Daems, A. Nowe, and J. Helsen
2023b. Pattern mining based data fusion for wind turbine condition monitoring. *Journal of Physics Conference Series*, 2507:012001.

Chesterman, X., T. Verstraeten, P.-J. Daems, A. Nowé, and J. Helsen
2021. Condition monitoring of wind turbines using machine learning based anomaly detection and statistical techniques for the extraction of 'healthy data'. *Proceedings of the Annual Conference of the PHM Society*.

Chesterman, X., T. Verstraeten, P.-J. Daems, F. P. Sanjines, A. Nowé, and J. Helsen
2022. The detection of generator bearing failures on wind turbines using machine learning based anomaly detection. *Journal of Physics: Conference Series*, 2265(3):032066.

Clifton, A., S. Barber, A. Bray, P. Enevoldsen, J. Fields, A. M. Sempreviva, L. Williams, J. Quick, M. Purdue, P. Totaro, and Y. Ding
2023. Grand challenges in the digitalisation of wind energy. *Wind Energy Science*, 8(6):947–974.

Cohen, W.
1995. Fast effective rule induction. Pp. 115–123.

Cornel, D., F. Gutiérrez Guzmán, G. Jacobs, and S. Neumann
2021. Condition monitoring of roller bearings using acoustic emission. *Wind Energy Science*, 6(2):367–376.

CTE Wind France
2024. The pile group solution. https://www.cte-wind.com/fr/solution/pile-foundation-30/. Accessed: 2024-10-20.

Cui, Y., P. Bangalore, and L. B. Tjernberg
2018. An anomaly detection approach based on machine learning and scada data for condition monitoring of wind turbines. *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, Pp. 1–6.

Dannenberg, L.
2014a. Offshore wind energy. In Schaffarczyk [2014], Pp. 406–454.

Dannenberg, L.
2014b. Rotor blades. In Schaffarczyk [2014], Pp. 162–201.

Dao, C., B. Kazemtabrizi, and C. Crabtree
2019. Wind turbine reliability data review and impacts on levelised cost of energy. *Wind Energy*, 22(12):1848–1871.

de Amorim, L. B., G. D. Cavalcanti, and R. M. Cruz
2023. The choice of scaling technique matters for classification performance. *Applied Soft Computing*, 133:109924.

Dienst, S. and J. Beseler
2016. Automatic anomaly detection in offshore wind scada data. *Wind Europe summit 2016*, Pp. 1–7.

Dinwoodie, I., D. McMillan, M. Revie, I. Lazakis, and Y. Dalgic
2013. Development of a combined operational and strategic decision support model for offshore wind. *Energy Procedia*, 35:157–166. DeepWind'2013 – Selected papers from 10th Deep Sea Offshore Wind RD Conference, Trondheim, Norway, 24 – 25 January 2013.

Domingues, M. A. and S. O. Rezende
2005. Post-processing of association rules using taxonomies. In *2005 portuguese conference on artificial intelligence*, Pp. 192–197.

Du, S., Y. Wan, C. Zhang, and S. Zhang
2023. Anomaly root cause analysis for wind turbines based on denoising autoencoder and sparse estimation. In *2023 IEEE 12th Data Driven Control and Learning Systems Conference (DDCLS)*, Pp. 449–454.

Emmanuel, T., T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona
2021. A survey on missing data in machine learning. *Journal of Big Data*, 8(140):1–37.

Faber, T.
2014. Tower and foundation. In Schaffarczyk [2014], Pp. 253–272.

Feremans, L., B. Cule, and B. Goethals
2022. Petsc: pattern-based embedding for time series classification. *Data Mining and Knowledge Discovery*, 36:1015–1061.

Feremans, L., V. Vercruyssen, W. Meert, B. Cule, and B. Goethals
2020. A framework for pattern mining and anomaly detection in multi-dimensional time series and event logs. *New Frontiers in Mining Complex Patterns: 8th International Workshop, NFMCP 2019, Held in Conjunction with ECML-PKDD 2019, Würzburg, Germany, September 16, 2019, Revised Selected Papers*, Pp. 3–20.

Fernandes, E.
2023. Wind turbine main shaft market has huge growth in industry | size, share, trends forecasts by 2029. https://www.linkedin.com/pulse/wind-turbine-main-shaft-market-has-huge-growth. Accessed: 2024-07-04.

Fournier-Viger, P., J. C.-W. Lin, R. Nkambou, B. Vo, and V. S. Tseng
2019. *High-Utility Pattern Mining: Theory, Algorithms and Applications*. Springer Cham.

Fournier-Viger, P., J. C.-W. Lin, B. Vo, T. T. Chi, J. Zhang, and H. B. Le
2017. A survey of itemset mining. *WIREs Data Mining and Knowledge Discovery*, 7(4):e1207.

Freund, Y. and R. Schapire
1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(SS971504):119–139.

Friedman, J. H.
2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232.

Froese, M.
2019. Ge unveils nacelle of massive haliade-x wind turbine. https://www.windpowerengineering.com/ge-unveils-nacelle-of-massive-haliade-x-wind-turbine/. Accessed: 2024-06-30.

Gan, W., L. Chen, W. Shicheng, J. Chen, and C.-M. Chen
2021. Anomaly rule detection in sequence data. *CoRR*, abs/2111.15026:1–14.

Gan, W., J. C.-W. Lin, P. Fournier-Viger, H.-C. Chao, V. Teng, and P. Yu
2019. A survey of utility-oriented pattern mining. *IEEE Transactions on Knowledge and Data Engineering*, 33:1306–1327.

Garlick, W., R. Dixon, and S. Watson
2009. A model-based approach to wind turbine condition monitoring using scada data.

In *Keith J Burnham and Olivier C L Haas (eds.). 20th International Conference on Systems Engineering*, Pp. 1–8.

Gay, G., T. Menzies, M. Davies, and K. Gundy-Burlet
2010. Automatically finding the control variables for complex system behavior. *Automated Software Engineering*, 17:439–468.

Goethals, B.
2010. Frequent set mining. In *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, eds., Pp. 321–338. Berlin, Heidelberg: Springer-Verlag.

Goodfellow, I., Y. Bengio, and A. Courville
2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Gruber, T.
2009. *Ontology*. Springer-Verlag.

Gruber, T. R.
1993. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220.

Guarino, N.
1998. Formal ontology and information systems. Pp. 3–15.

Guillom
2005. Darrieus wind turbine. https://en.wikipedia.org/wiki/Darrieus_wind_turbine. Accessed: 2024-07-23.

Han, J., J. Pei, Y. Yin, and R. Mao
2004. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8:53–87.

Hashemi, N.
2024. Cs wind offshore integrates cnc onsite's precision milling machine for flanges up to 10-meter. https://offshore-channel.com/cs-wind-offshore-integrates-cnc-onsites-precision-milling-machine-for-flanges-up-to-10-meter
Accessed: 2024-10-17.

Hastie, T., R. Tibshirani, and J. Friedman, eds.
2017. *The Elements of Statistical Learning*. 233 Spring Street, New York, USA: Springer Science+Business Media, LLC.

He, Z., X. Xu, J. Huang, and S. Deng
2005. Fp-outlier: Frequent pattern based outlier detection. *Computer Science and Information Systems*, 2:103–118.

He, Z., X. Xu, J. Huang, and S. Deng
2014. Anomaly detection system by mining frequent pattern using data mining algorithm from network flow. *International Journal of Engineering Research Technology (IJERT)*, 3:1–7.

Heier, S., ed.
1996. *Wind energy conversion systems*. Chichester, United Kingdom: John Wiley Sons, Ltd.

Helbing, G. and M. Ritter
2018. Deep learning for fault detection in wind turbines. *Renewable and Sustainable Energy Reviews*, 98:189–198.

Helsen, J.
2021. Review of research on condition monitoring for improved om of offshore wind turbine drivetrains. *Acoustics Australia*, 49(2):251–258.

Hemalatha, C. S., V. Vaidehi, and R. Lakshmi
2015. Minimal infrequent pattern based approach for mining outliers in data streams. *Expert Systems with Applications*, 42:1998–2012.

Hu, Y.
2003. Treatment learning: Implementation and application.

Huber, P.
1981. *Robust Statistics*. John Wiley Sons.

Hutchinson, M. and F. Zhao
2023. *Global Wind Report 2023*. Brussels: Global Wind Energy Council.

Industrial Quick Search
2024a. Helical gears. https://www.iqsdirectory.com/articles/gear/helical-gears.html. Accessed: 2024-06-24.

Industrial Quick Search
2024b. Planetary gears. https://www.iqsdirectory.com/articles/gear/planetary-gears.html. Accessed: 2024-06-24.

Industrial Quick Search
2024c. Spur gears. https://www.iqsdirectory.com/articles/gear/spur-gears.html. Accessed: 2024-06-24.

Ke, G., Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu
2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., volume 30. Curran Associates, Inc.

Keet, M., ed.
2020. *An Introduction to Ontology Engineering v1.5*.

Kim, K., G. Parthasarathy, O. Uluyol, and Y. Patel
2011. Use of scada data for failure detection in wind turbines. *ASME 5th Int. Conf. Energy Sustain.*,, P. 2071–2079.

Koh, Y. S. and N. Rountree
2005. Finding sporadic rules using apriori-inverse. In *Advances in Knowledge Discovery and Data Mining*, T. B. Ho, D. Cheung, and H. Liu, eds., Pp. 97–106, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kovarik, M., L. Sarga, and P. Klímek
2015. Usage of control charts for time series analysis in financial management. *Journal of Business Economics and Management*, 16:138–158.

Kuchar, J. and V. Svatek
2017. Spotlighting anomalies using frequent patterns. *Proceedings of Machine Learning Research*, 71:33–42.

Küçük, D. and D. Küçük
2018. Ontowind: An improved and extended wind energy ontology. *CoRR*, abs/1803.02808.

Kusiak, A. and W. Li
2011. The prediction and diagnosis of wind turbine faults. *Renewable Energy*, 36:16–23.

Kusiak, A. and A. Verma
2012. Analyzing bearing faults in wind turbines: A data-mining approach. *Renewable Energy*, 48:110–116.

Lantz, E., O. Roberts, J. Nunemaker, E. DeMeo, K. Dykes, and G. Scott
2019. Increasing Wind Turbine Tower Heights: Opportunities and Challenges. Technical report, National Renewable Energy Laboratory.

Lewis, M.
2022. Check out this offshore wind farm's massive gravity-based foundations. https://electrek.co/2022/11/07/offshore-wind-farm-gravity-based-foundations/. Accessed: 2024-06-30.

Leys, C., C. Ley, O. Klein, P. Bernard, and L. Licata
2013. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, 49(4):764–766.

Li, H., W. Peng, C.-G. Huang, and C. Guedes Soares
2022. Failure rate assessment for onshore and floating offshore wind turbines. *Journal of Marine Science and Engineering*, 10(12).

Li, J., X. Lei, H. Li, and L. Ran
2014. Normal behavior models for the condition assessment of wind turbine generator systems. *Electric Power Components and Systems*, 42(11):1201–1212.

Li, L., K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar
2018. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52.

Li, W., J. Han, and J. Pei
2001. Cmar: Accurate and efficient classification based on multiple class-association rules. Pp. 369–376.

Lima, L. A. M., A. Blatt, and J. Fujise
2020. Wind turbine failure prediction using scada data. *Journal of Physics: Conference Series*, 1618(2):022017.

Lin, F., W. Le, and B. Jin
2010. Research on maximal frequent pattern outlier factor for online high-dimensional time-series outlier detection. *Journal of Convergence Information Technology*, 5:66–71.

Lin, J., E. Keogh, L. Wei, and S. Lonardi
2007. Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144.

Liu, B., W. Hsu, and Y. Ma
1998. Integrating classification and association rule mining. *KDD'98: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, Pp. 80–86.

Liu, B., Y. Ma, and C.-K. Wong
2001. Classification using association rules: Weaknesses and enhancements. *Data Mining for Scientific and Engineering Applications. Massive Computing*, Pp. 1–11.

Liu, J., G. Yang, X. Li, Q. Wang, Y. He, and X. Yang
2023. Wind turbine anomaly detection based on scada: A deep autoencoder enhanced by fault instances. *ISA Transactions*, 139:586–605.

Liu, X., S. Lu, Y. Ren, and Z. Wu
2020. Wind turbine anomaly detection based on scada data mining. *Electronics*, 9(5):1–16.

Liu, Z. and L. Zhang
2020. A review of failure modes, condition monitoring and fault diagnosis methods for large-scale wind turbine bearings. *Measurement*, 149:107002.

Lu, C.-W. and M. R. Reynolds
2001. Cusum charts for monitoring an autocorrelated process. *Journal of Quality Technology*, 33:316 – 334.

Lund and Sorensen A/S
2024. Pt100 temperature sensors. https://ls-windpower.com/product/pt100-temperature-sensors/. Accessed: 2024-07-04.

Lydia, M., S. S. Kumar, A. I. Selvakumar, and G. E. Prem Kumar
2014. A comprehensive review on wind turbine power curve modeling techniques. *Renewable and Sustainable Energy Reviews*, 30:452–460.

Marinica, C.
2010. *Association Rule Interactive Post-processing using Rule Schemas and Ontologies - ARIPSO*. Theses, Université de Nantes.

Maron, J., D. Anagnostos, B. Brodbeck, and A. Meyer
2022. Artificial intelligence-based condition monitoring and predictive maintenance framework for wind turbines. *Journal of Physics: Conference Series*, 2151(1):1–9.

Marti-Puig, P., A. Blanco-M., M. Serra-Serra, and J. Solé-Casals
2021. Wind turbine prognosis models based on scada data and extreme learning machines. *Applied Sciences*, 11(2):590.

Marykovskiy, Y., T. Clark, J. Day, M. Wiens, C. Henderson, J. Quick, I. Abdallah, A. M. Sempreviva, J.-P. Calbimonte, E. Chatzi, and S. Barber
2024. Knowledge engineering for wind energy. *Wind Energy Science*, 9(4):883–917.

Mazidi, P., L. Bertling, and M. A. Sanz-Bobi
2017. Performance analysis and anomaly detection in wind turbines based on neural networks and principal component analysis. Pp. 1–9.

McKinnon, C., J. Carroll, A. McDonald, S. Koukoura, D. Infield, and C. Soraghan
2020. Comparison of new anomaly detection technique for wind turbine condition monitoring using gearbox scada data. *Energies*, 13(5152):1–19.

Menzies, T., E. Kocagüneli, L. Minku, F. Peters, and B. Turhan
2015. Chapter 24 - using goals in model-based reasoning. In *Sharing Data and Models in Software Engineering*, T. Menzies, E. Kocagüneli, L. Minku, F. Peters, and B. Turhan, eds., Pp. 321–353. Boston: Morgan Kaufmann.

Meyer, A.
2021. Early fault detection with multi-target neural networks. *CoRR*, abs/2106.08957.

Miele, E. S., F. Bonacina, and A. Corsini
2022. Deep anomaly detection in horizontal axis wind turbines using graph convolutional autoencoders for multivariate time series. *Energy and AI*, 8:100145.

Montgomery, D.
2009. *Introduction to Statistical Quality Control*. John Wiley & Sons, Incorporated.

Nejad, A. R., J. Keller, Y. Guo, S. Sheng, H. Polinder, S. Watson, J. Dong, Z. Qin, A. Ebrahimi, R. Schelenz, F. Gutiérrez Guzmán, D. Cornel, R. Golafshan, G. Jacobs, B. Blockmans, J. Bosmans, B. Pluymers, J. Carroll, S. Koukoura, E. Hart, A. McDonald, A. Natarajan, J. Torsvik, F. K. Moghadam, P.-J. Daems, T. Verstraeten, C. Peeters, and J. Helsen
2022. Wind turbine drivetrains: state-of-the-art technologies and future development trends. *Wind Energy Science*, 7(1):387–411.

Nielsen, J. S., R. P. van de Pieterman, and J. D. Sørensen
2014. Analysis of pitch system data for condition monitoring. *Wind Energy*, 17(3):435–449.

Orlando O. Atienza, L. C. T. and B. W. Ang
2002. A cusum scheme for autocorrelated observations. *Journal of Quality Technology*, 34(2):187–199.

Page, E. S.
1954. Continuous inspection schemes. *Biometrika*, 41(1-2):100–115.

Papadopoulos, P. and L. Cipcigan
2009a. Wind turbines' condition monitoring: An ontology model. Pp. 1 – 4.

Papadopoulos, P. and L. Cipcigan
2009b. Wind turbines' condition monitoring: An ontology model. Pp. 1 – 4.

Parrinello, C. M., M. E. Grams, Y. Sang, D. Couper, L. M. Wruck, D. Li, J. H. Eckfeldt, E. Selvin, and J. Coresh
2016. Iterative outlier removal: A method for identifying outliers in laboratory recalibration studies. *Clinical Chemistry*, 62(7):966–972.

Peter, R., D. Zappalá, V. Schamboeck, and S. J. Watson
2022. Wind turbine generator prognostics using field SCADA data. *Journal of Physics: Conference Series*, 2265(3):032111.

Pfaffel, S., S. Faulstich, and K. Rohrig
2017. Performance and reliability of wind turbines: A review. *Energies*, 10:1904.

Quinlan, R.
1993. *C4.5 Programs for machine learning*. Morgan Kaufmann Publishers, Inc.

Rahman, A., X. Yue, K. Radke, and E. Foo
2016. Finding anomalies in scada logs using rare sequential pattern mining. *Network and System Security. NSS 2016. Lecture Notes in Computer Science*, 9955:1–9.

Randall, R., ed.
2011. *Vibration-based Condition Monitoring: Industrial, Aerospace and Automotive Applications*. Chichester, United Kingdom: John Wiley Sons, Ltd.

Renström, N., P. Bangalore, and E. Highcock
2020. System-wide anomaly detection in wind turbines using deep autoencoders. *Renewable Energy*, 157:647–659.

Roelofs, C. M., M.-A. Lutz, S. Faulstich, and S. Vogt
2021. Autoencoder-based anomaly root cause analysis for wind turbines. *Energy and AI*, 4:100065.

Ruviaro, M., F. Runcos, N. Sadowski, and I. M. Borges
2012. Analysis and test results of a brushless doubly fed induction machine with rotary transformer. *IEEE Transactions on Industrial Electronics*, 59(6):2670–2677.

Schaffarczyk, A., ed.
2014. *Understanding Wind Power Technology: Theory, Deployment and Optimization*. Chichester, United Kingdom: John Wiley Sons, Ltd.

Schlechtingen, M. and I. F. Santos
2012. Condition monitoring with wind turbine scada data using neuro-fuzzy normal behavior models. *Turbo Expo: Power for Land, Sea, and Air*, 6:717–726.

Schlechtingen, M. and I. F. Santos
2014. Wind turbine condition monitoring based on scada data using normal behavior models. part 2: Application examples. *Applied Soft Computing*, 14(1):447–460.

Schlechtingen, M., I. F. Santos, and S. Achiche
2013. Wind turbine condition monitoring based on scada data using normal behavior models. part 1: System description. *Applied Soft Computing*, 13(1):259–270.

Siegfriedsen, S.
2014. The drive train. In Schaffarczyk [2014], Pp. 202–252.

Simmons, C. H., N. Phelps, and D. E. Maguire
2012. Chapter 31 - cams and gears. In *Manual of Engineering Drawing (Fourth Edition)*, C. H. Simmons, N. Phelps, and D. E. Maguire, eds., Pp. 253–265. Oxford: Butterworth-Heinemann.

SKF
2024. Double row tapered roller bearings. https://www.skf.com/uk/products/rolling-bearings/roller-bearings/tapered-roller-bearings/double-row-tapered-roller-bearings. Accessed: 2024-07-04.

Sohoni, V., S. C. Gupta, and R. K. Nema
2016. A critical review on wind turbine power curve modelling techniques and their applications in wind based energy systems. *Journal of Energy*, 2016(1):8519785.

Spiteri Staines, C., C. Caruana, and J. Licari
2015. Review of power converters for wind energy systems.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov
2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Strang, G.
1994. Wavelets. *American Scientist*, 82(3):250–255.

Studer, R., V. Benjamins, and D. Fensel
1998. Knowledge engineering: Principles and methods. *Data Knowledge Engineering*, 25(1):161–197.

Sun, P., J. Li, C. Wang, and X. Lei
2016. A generalized model for wind turbine anomaly identification based on scada data. *Applied Energy*, 168:550–567.

Szathmary, L., A. Napoli, and P. Valtchev
2007. Towards rare itemset mining. *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, 1:305–312.

Takanashi, M., S. ichi Sato, K. Indo, N. Nishihara, H. Hayashi, and T. Suzuki
2022. Anomaly prediction for wind turbines using an autoencoder with vibration data supported by power-curve filtering. *IEICE Transactions on Information and Systems*, E105.D(3):732–735.

Tang, X., G. Li, and G. Chen
2009. Fast detecting outliers over online data stream. *2009 International Conference on Information Engineering and Computer Science*, Pp. 1–4.

Tautz-Weinert, J. and S. J. Watson
2017. Using scada data for wind turbine condition monitoring – a review. *IET Renewable Power Generation*, 11(4):382–394.

Turnbull, A., J. Carroll, and A. McDonald
2021. Combining scada and vibration data into a single anomaly detection model to predict wind turbine component failure. *Wind Energy*, 24(3):197–211.

Udo, W. and M. Yar
2021. Data-driven predictive maintenance of wind turbine based on scada data. *IEEE Access*, 9:162370–162388.

Untrakdrover
2012. Floating wind turbine. https://en.wikipedia.org/wiki/Floating_wind_turbine. Accessed: 2024-07-23.

van Buren, S.
2021. *Flexibel Imputation of Missing Data, Second Edition*. Chapman Hall.

van Buren, S. and K. Groothuis-Oudshoorn
2011. mice: multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(2):1–67.

Verma, A., D. Zappalá, S. Sheng, and S. J. Watson
2022. Wind turbine gearbox fault prognosis using high-frequency scada data. *Journal of Physics: Conference Series*, 2265(3):032067.

Webb, G., S. Butler, and D. Newlands
2003. On detecting differences between groups. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Wilcoxon
2024. Wind power: Vibration sensors for monitoring wind turbine health. https://wilcoxon.com/industries-and-applications/wind-power/. Accessed: 2024-07-04.

Woodall, W. H. and M. M. Ncube
1985. Multivariate cusum quality-control procedures. *Technometrics*, 27(3):285–292.

Yang, W., P. J. Tavner, C. J. Crabtree, Y. Feng, and Y. Qiu
2014. Wind turbine condition monitoring: technical and commercial challenges. *Wind Energy*, 17(5):673–693.

Yin, X. and J. Han
2003. Cpar: Classification based on predictive association rules. Pp. 331–335.

Zaher, A., S. McArthur, D. Infield, and Y. Patel
2009. Online wind turbine fault detection through automated scada data analysis. *Wind Energy*, 12(6):574–593.

Zaki, M., S. Parthasarathy, M. Ogihara, and W. Li
1997. New algorithms for fast discovery of association rules. In *Proc. 3rd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'97, Newport Beach, CA)*, Pp. 283–296, Menlo Park, CA, USA. AAAI Press.

Zappalá, D. and P. Tavner
2022. *2.11 - Wind Turbine Reliability - Maintenance Strategies*, volume 1-9, Pp. 353–370. Elsevier, 2nd edition.

Zhang, W., J. Wu, and J. Yu
2010. An improved method of outlier detection based on frequent pattern. *2010 WASE International Conference on Information Engineering*, Pp. 3–6.

Zhao, H., H. Liu, W. Hu, and X. Yan
2018. Anomaly detection and fault analysis of wind turbine components based on deep learning network. *Renewable Energy*, 127:825–834.

Zou, H. and T. Hastie
2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2):301–320.