Anomaly Detection in Wind Turbine SCADA Data for Power Curve Cleaning

Rory Morrison[1], Xiaolei Liu[1,*], and Zi Lin[2]

[1] *James Watt School of Engineering, University of Glasgow, Glasgow, G12 8QQ.*

[2] *Department of Mechanical and Construction Engineering, Northumbria University, Newcastle upon Tyne, NE1 8ST, UK.*

[*] *Corresponding author, xiaolei.liu@glasgow.ac.uk*

Abstract

Wind turbine power curve cleaning, by way of removing curtailment, stoppage, and other anomalies, is an essential step in making raw data useable for further analysis, such as determining turbine performance, site characteristics, or improving forecasting models. Typically, data comes as SCADA (Supervisory Control and Data Acquisition) data, so contains not only environmental and turbine performance data but also the control action imposed on the turbine by the operator. Many different anomaly detection (AD) methods have been proposed to clean power curves; however, few papers have explored filtering explicit and obvious anomalies from the SCADA prior to running AD. This paper actively explores this filtering impact by comparing the performances of 4 different AD methods with/without filtering. These are: iForest, Local Outlier Factor, Gaussian Mixture Models, and k-Nearest Neighbours. Each approach is evaluated in terms of prediction error, data removal rates, and ability to maintain the underlying wind statistical characteristics. The results show the effectiveness of filtering with every technique showing improvement compared to its unfiltered counterpart. Furthermore, Gaussian Mixture Models are shown to provide favourable accuracy whilst maintaining wind variability, however, with the wide range of performances of methods, a user's choice may be different depending on their needs.

Keywords— Wind turbine, power curve, data cleaning, anomaly detection

Nomenclature

Abbreviations

AD	Anomaly Detection
BIC	Bayesian Information Criterion
CPU	Central Processing Unit
$DBSCAN$	Density Based Spatial Clustering of Applications with Noise
$FIML$	Full Information Maximum Likelihood
GMM	Gaussian Mixture Modelling
$iForest$	Isolation Forest
IQR	Interquartile Range
kNN	k Nearest Neighbours
LOF	Local Outlier Factor
MAR	Missing At Random
$MCAR$	Missing Completely At Random
$MNAR$	Missing Not At Random
NN	Neural Network

$RMSE$ Root Mean Squared Error

$SCADA$ Supervisory Control and Data Acquisition

WT Wind Turbine

$WTPC$ Wind Turbine Power Curve

Symbols - Isolation Forest.

$c(n)$ Average of $h(x)$ for n instances

$E(h(x))$ Average path length across all iTress

$h(x)$ Averaged path length

n Number of instances

$s(x, n)$ Anomaly score

Symbols - Gaussian Mixture Models

μ_p Mean of a given variable

IQR Interquartile range, $Q_75 - Q_25$

k Number of mixtures assumed

p Number of variables

p_{lower}, p_{upper} Lower and upper bounds of the box plot

Q_25, Q_75 Lower and upper quartiles, equivalent to 25^{th} and 75^{th} quartiles.

Symbols - k Nearest Neighbours

k Number of nearest neighbours

Symbols - Local Outlier Factor

lrd local reach distance

N_{MinPts} Number of nearest neighbours to consider

o A nearest neighbour of p when considering $MinPts$ of nearest neighbours

$reach - dist$ Reach distance

x The instance being studied

Symbols

δIQR_u Percentage difference in wind speed IQR

γ Elimination rate

e_m Prediction error as percentage of P_r

$h(u_i)$ Predicted power for instance i with windspeed u

$IQR_{u,b}, IQR_{u,a}$ IQR of wind speed before and after AD

n Number of instances in test set

N_b, N_a Number of instances before and after AD

p_i Actual power value for instance i

P_r Rated power of the wind turbine

u Wind speed

Units

GW Gigawatts

km Kilometer

kW Kilowatts

m/s Meters/second

MW Megawatts

RPM Revolutions per Minute

1 Introduction

The European Union and United Kingdom are committed to extensive targets to increase offshore wind energy capacity as part of the greening of the energy sector. This is to support the commitments of many nations to the Paris Climate Agreement. As of November 2020, the European Union has committed to increasing their 12GW of capacity to 60GW by 2030 and 300GW by 2050 [1]. Similarly the United Kingdom pledged to increase their 10GW capacity to 40GW by 2030 [2]. It is certainly an exciting time to be involved in wind energy academia as these targets will need to be supported by research to overcome the plethora of challenges facing such ambitious targets, such as continuing the reduction of levelized cost of energy, turbine life extension, increasing reliability, and improving forecasting models, to name a few.

The common thread to these research topics is their reliance, in part, on *SCADA* data from already deployed wind turbines. SCADA (Supervisory Control and Data Acquisition) data is continuously generated by each wind turbine (WT) when deployed. It documents production (power output), turbine parameters (rotor RPM, blade pitch angle, braking, bearing temperatures etc), supervisory action imposed by the operator on the WT, and environmental conditions (air temperature, wind speed, ice indication etc). If we wish to use SCADA data to explore the relationship WTs have with the environment then anomalies must be removed and the *power curve* cleaned. The wind turbine power curve (WTPC) is simply a plot of wind speeds (u) against power produced by the WT.

1.1 Anomalies in SCADA

Anomalies are defined as instances that do not fit the patterns of the rest of the data. This misfitting makes it appear that the instances in question have been generated by an altogether different mechanism to that generating the "normal" data [3]. It is important that these anomalies are identified and removed so as to not bias the relationships being studied. For WT SCADA data, anomalies are typically categorized into 3 types [4] and any anomaly detection (AD) method must be designed with these in mind. These are described below and illustrated with **Figure 1**:

- Anomaly type 1: These anomalies are characterized by no power output whilst above cut in wind speed i.e. parking/downtime imposed by the operator. In general AD terms, these instances would be characterized as *contextual* anomalies [3]. An instance is a contextual anomaly if it would be considered normal in a different context. The values of each feature are normal in isolation, but abnormal when considered together.

- Anomaly type 2: These anomalies are characterized as steady and continuous positive power output at a power less than the turbine's rated power, P_r, i.e. curtailment imposed by the operator. These can be characterized as *contextual* anomalies as well.

- Anomaly type 3: These anomalies are randomly scattered across the feature space. Reference [4] suggested these may be caused by sensor malfunction or noise in signal processing. It is also possible these instances are generated by stop-to-operation transitions or vice versa. These are best described as *point* anomalies. Point anomalies are single instances that clearly do not conform to the nature of the rest of the data.

Note that "Contextual" anomalies are exactly that, driven by context so will differ depending upon the end use of the SCADA data. For applications that seek to understand turbine performance, we would consider any significant deviation from the manufacturer's specified WTPC as anomalous. For example, an instance where no power was generated at wind speeds above cut-in, such as curtailment by the operator, we would consider an anomaly. For condition monitoring applications, the same curtailed instance would not be considered anomalous as the operating mode is included in the context. This paper focuses on applications concerned with turbine performance so considers deviation from the WTPC as anomalous, be it from operator influence or otherwise.

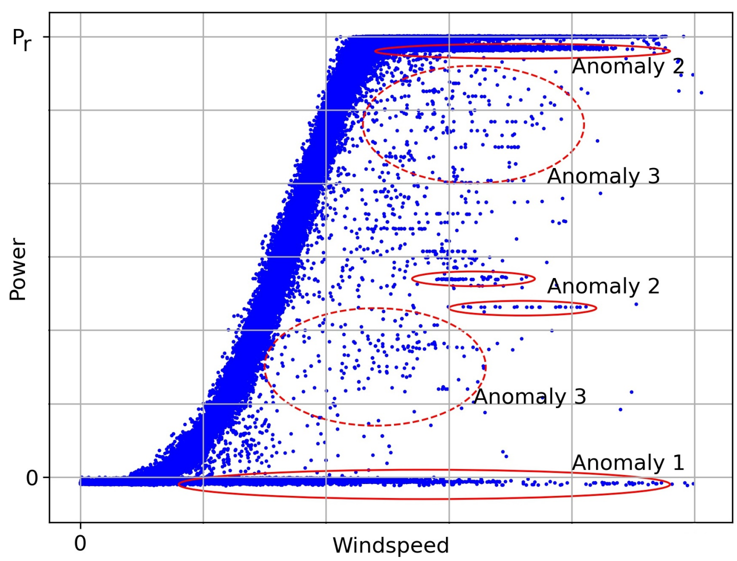

Figure 1: Wind turbine power curve showing examples of anomaly types. P_r indicates rated power of the WT.

To emphasize, contrary to the bulk of AD research, the mechanisms of anomaly types 1 and 2 are well understood. Fault, downtime, and curtailment instances should be explicitly labelled in the SCADA by any competent system. It only remains for the user to remove these instances.

1.2 Literature Review

Taking a broad view, the literature can be split into two groups based on their approaches to cleaning WTPC data: group 1, those that pre-process data prior to running AD and group 2, those that do not. This is, of course, among many divisions that can be made.

In group 2, the non-pre-processing group, a wide range of approaches have been formulated. Notable examples include defining normal behaviour [5], using image-based approaches [6], using a "Change Point Grouping Algorithm and Quartile Algorithm" [7], and using Gaussian Mixture Models (GMM) [8]. Common themes from these works is the difficulty in dealing with "stacked" data, this is where anomalies are so numerous they start to sway the statistical perception of "normal". Another theme is the difficulty to reliably produce good results when anomalies form increasingly large percentages of the data. All of these papers go straight to the anomaly detection method with no pre-processing of data to remove obvious outliers or missing data.

In contrast to the above, group 1 papers, those that implemented pre-processing, appear to have approached WTPC cleaning from a data-mining standpoint. In [9] the authors pre-processed SCADA data into categories of "unnatural", "constant", "exceeding", "missing" or otherwise valid data. This was followed by determination of "irrational" data using a 2-step process. First, each instance of the remaining data was given a weighting depending on its distance from the manufacturer-specified WTPC. Second, the Local Outlier Factor (LOF) technique was applied with these weightings. The entire process resulted in some 4,190 instances of a 18,001 dataset being removed in their case study. Such explicit use of the manufacturer's specifications, i.e. cleaning based on how it performs in theory rather than in practice, is completely at odds with group 2's unsupervised approach. The approach is questionable given that it is not straightforward to compare the specified WTPC of a WT to that being achieved on a given site. To point out the most glaring obstacles, differences in topography, terrain roughness, and wind regime will need to be compensated for, as will any potential wake effects from nearby turbines. There is a risk of introducing bias into the data, a non-problem of the unsupervised group 2. Aside from this, [9] dealt with obvious invalid SCADA data, such as missing data, of which none of group 2 papers even mention. Logically, any AD technique will find defining "normal" easier if valid data makes up a greater percentage of the dataset; [9] pointed out that the LOF, which uses distances between instances, would struggle with stacked data. Unfortunately, the impact on power curve cleaning with and without pre-processing is not compared by the authors.

Another paper that employed pre-processing was [10]. This study took a multi-step approach and implemented simple statistical methods prior to running the DBSCAN (density based spatial clustering of applications with noise) technique. The first step eliminated negative power instances. The second and third steps applied the box plot rule (see **Section 2.4** for an explanation of the box plot rule) to wind speed and power intervals respectively. Finally, the DBSCAN method was applied. The authors stated that the purpose of applying the box plot rule was to eliminate sparse outliers, so making the boundaries of the stacked outliers clearer and improve the efficacy of DBSCAN. The impacts of steps 1-3 were, unfortunately, not evaluated.

Of all the papers referenced in the literature review, only two papers dealt with missing data, references [9] and [11]. The latter study, [11], concerns the detection of blade icing and does so by comparing 3 methods. These are: percentage deviation from the manufacturer's WTPC, standard deviation of power for a wind speed interval, and using quantiles of power for a wind speed interval. Furthermore, this paper is the only one to use explicit fault or curtailment indications in the SCADA data; however, it should be noted that [9] did reference a study that used operator logs. This general lack of acknowledgment is unusual given that the international standard for power curve measurement, IEC 61400-12, prescribes a "data quality check" of removing "unavailable" or "out of range" measurements and data rejection based on power limited instances and faults with referencing to operator logs [12].

On the topic of ice detection, deviation from the WTPC is a common approach for detection of ice accumulation on the blades. The authors note that ice detection studies are rarely referenced by the general WTPC cleaning community, and vice versa. Icing typically manifests itself as a deviation from the WTPC, more specifically a reduction in power compared to manufacturer's specification. Other studies on this topic include [13] and [14]. In [13], the supervised learning Random Forest classifier (not to be confused with iForest) is used on pre-processed and labelled data. The data was pre-processed by the WT operator prior to being handed over to the authors, therefore the precise pre-processing methods are not discussed. In [14], kNN-regression is employed; however, data pre-processing is not mentioned.

The prevailing attitude of group 2 appears to be the less supervision a technique requires the better, provided this is not at the expense of results. Group 1 better embraced the spirit of SCADA data but rarely acknowledge that explicit or implicit indications of faults or curtailment exist, or suggest actively using this knowledge as part of pre-processing. Hesitation around using these indications is understandable given the potential for mislabelling, lags between a fault occurring and the alarm being logged, or the possibility that these logs simply do not exist. It follows that if an AD method can perform just as well without these logs then it would be undesirable to add steps into the process to unnecessarily act upon these logs. However, given the lack of acknowledgement, it is hard to say whether this hesitation is justified. It is the author's experience that these explicit indications of operating status are included in SCADA far more regularly than they are not. After all, SCADA is Supervisory Control and Data Acquisition, if a system does not record these parameters it can hardly be called a SCADA system. As such, the lack of acknowledgement in the literature is surprising. The choice to omit knowledge of indications from an AD method is still a choice.

1.3 Contributions and Paper Organisation

The key contributions of this paper to current knowledge gaps are as follows:

- The impact of filtering SCADA of explicit and obvious anomalies, such as faults and curtailment, based on indications in the data prior to running AD techniques was found to be understudied in the literature. This paper investigates this by comparing the performances of AD techniques with and without filtering applied.
- Going beyond simply filtering out obvious/explicit anomalies, this paper also explores utilizing this data further. Where possible, the filtered data is used to form an anomaly-class. This allows classification-based anomaly detection methods to be used. Such an approach has not been found in the literature.
- Underpinning the previous 2 contributions, simple rules for the filtering are developed based upon explicit fault and curtailment indications in the SCADA, as well as pitch values. These rules avoid the need to overly specify expectations of how the turbine should be performing, i.e. avoids the need to incorporate the

manufacturer-specified WTPC.

- With the large assortment of AD techniques present in the literature, this paper provides comparison between 4 distinct techniques on their ability to clean the WTPC, whilst still maintaining the statistical variability in the wind speed feature.
- Whilst papers in the literature have previously evaluated AD techniques on their ability to maintain data, ability to maintain statistical variability in the wind speed feature appears understudied. Such a metric would highlight good performers. This metric is introduced and applied here.
- Finally, the importance of proper treatment of missing data, that is to say instances with some or all values missing or corrupted, is raised. Treatment of missing data appears to be rarely mentioned in the literature, in this paper the importance of such data in the context of anomaly detection is discussed.

The remainder of this paper is organized as follows. Section 2 describes the methodology used in this paper. This includes descriptions of the 4 AD techniques, their operation, and how they are applied. Additionally, descriptions of 3 different filtering-based approaches to SCADA data are given. Section 3 describes the WTs and wind farms the SCADA data used in this paper originate from. Section 4 details and interprets the impacts of each AD method as well as comparing between them. Finally, Section 5 summarizes the key findings and contributions of the paper.

2 Methodology

The methodology of this study is shown in **Figure 2**. The methodology is composed of 3 main components:

- Missing data treatment.
- Anomaly detection and removal
  - Explicit and obvious anomaly filtering.
  - Anomaly detection proper.

- Evaluation

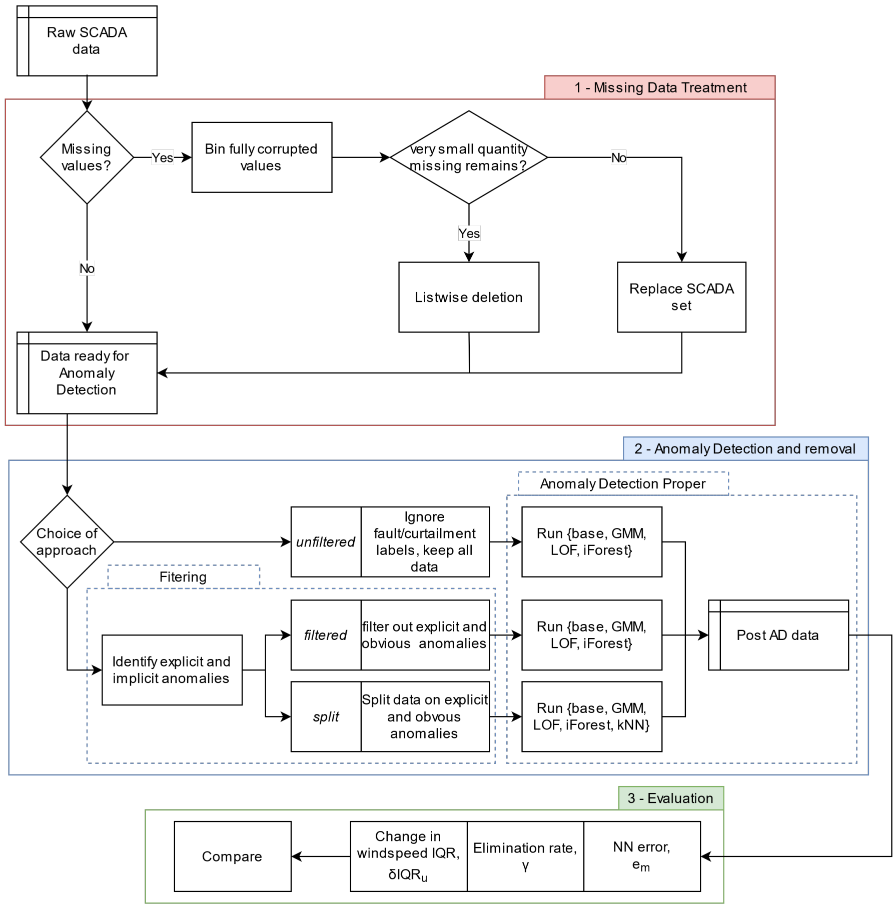

Figure 2: Methodology for treatment of SCADA data for anomaly detection.

Treatment of missing data is discussed in **Sections 2.1**. From the choice to filter or not, 3 approaches of *Unfiltered, Filtered,* and *Split* are proposed in **Section 2.2** along with a description of how filtering is performed. Five anomaly detection techniques, including ”do-nothing”, are described in **Section 2.3** along with how they are applied. The combination of an approach and an AD technique is referred to herein as an ”AD method”. Finally, the evaluation of each AD method is detailed in **Section 2.4**.

2.1 Missing Data Treatment

The treatment of missing data appears to be rarely discussed in papers relying on SCADA data. Looking again at the papers, only [9] and [11] make reference to handling missing data. As discussed in [15], missing data can be categorized into 3 groups:

- Missing Completely at Random (MCAR): There is no correlation between the missingness of data and any variables.
- Missing at Random (MAR): Missingness is correlated with a variable. The cause can be measured and included in missing data methods. Failure to include this would introduce bias into future data models.
- Missing Not at Random (MNAR): Like MAR, missingness is correlated with a variable. However, the cause cannot be measured and so cannot be corrected for in missing data methods.

The author notes the parallels between *missing* data and *anomalous* data. Take, for example, MAR and anomaly type 3. The descriptions are almost interchangeable. Reference [15] notes that common approaches to missing data are listwise deletion and data imputation. Listwise deletion is when an instance is simply deleted entirely. Both methods have been considered outdated for some time now due to their potential to introduce bias into data models. As a precursor to AD, one can see how these methods might be counterproductive by changing the nature of the data models.

Keeping in mind the aim of not introducing bias, listwise deletion is only appropriate if the data to be removed represents a small percentage of the overall data. This not only avoids introducing bias, but also maintains statistical power. Reference [15] suggested that less than 5% would be trivial. If greater percentages exist, then one must determine if the missingness is MCAR, MAR, or MNAR. For the purposes of this document, this is typically tested by a software package. Should MCAR or MAR be the cause, then *Full-Information Maximum Likelihood* (FIML) procedures are recommended. Similarly, the data imputation that would follow is also handled by the software and the theory is not explored here. In the unlikely event that the type of missingness is MNAR, then the methods available to correct the data become few. Reference [15] suggested use of *Pattern-Mixture methods*, however, from a pragmatic approach to SCADA data WTPC cleaning, further use of that dataset should be questioned.

In the context of WT SCADA data, it is common to have many instances that are entirely empty, barring features for WT identification, etc. These can be generated by many means such as clock time changes. These instances do not fall within the 3 missing data categories and should be removed entirely.

2.1.1 Application of Missing Data Treatment

As per **Figure 2**, fully corrupted instances are first removed. The number of instances containing missing data is then calculated. If these represent less than 5% of the data then they are considered trivial (as per [15]) and are removed via listwise deletion. Given the number of WT SCADA sets available, if this 5% threshold is exceeded, the SCADA set will be abandoned and another adopted. This is a pragmatic approach to avoid the need to employ FIML software.

2.2 Explicit and Obvious Anomaly Filtering

As discussed in **Section 1.2**, SCADA data, by its nature, contains explicit indications of the operational state of a WT, such as fault alarms and operational time. Furthermore, some instances can be characterised as "obvious" anomalies, namely high blade pitch values indicate the operator was stopping the turbine. Similarly, power reference values state the maximum power limit the operator imposed on the WT for each period. A value less than P_r indicates curtailment. These anomalies can be filtered from the datasets prior to running AD techniques. Alternatively, all data could be maintained, or the filtered data could be further leveraged. Arising from this, 3 distinct approaches are proposed:

- *Unfiltered*: No filtering occurs. As such, all SCADA data is kept and fed into AD processes.
- *Filtered*: Filtering occurs and the filtered data is removed entirely. The remain-

ing data is fed into AD processes. Issues of stacking and high initial anomaly percentages should be reduced. However, any mislabelled instances are lost from the dataset, along with statistical power.

- *Split*: The dataset is split in two according to the filtering rules. This forms an assumed-anomaly set and an assumed-normal set. AD techniques are run using both sets to identify instances that are mislabelled (i.e. in the wrong set). This approach attempts to make most use of all data.

Comparison of *Unfiltered, Filtered*, and *Split* approaches will identify the impact of pre-processing the SCADA data.

2.2.1 Application of Filtering

SCADA data will be filtered out for *Filtered* and *Split* approaches if any of the following conditions are met:

- SCADA ”Fault” feature indicates a fault;
- SCADA ”Operational time” feature indicates the WT was not in operation for the full duration of recording period. In practice, this is if the WT operated for less than 10-minutes in the 10-minute period.
- SCADA ”Power reference” feature value is less than 99% of P_r. This indicates the WT was curtailed.
- SCADA blade pitch angle feature value is greater than 30°. This indicates the WT was stopped by the operator. As shown in **Figure 3**, stoppage occurs at pitch angles of 80-90°, a threshold of 30° is chosen to capture some operation-to-stop transition instances whilst not incorrectly filtering out normal instances. From **Figure 4**, it is clear that normal operation pitch angles end at approximately 25°.

RPM will not be used in indicating curtailment. Whilst a lack of rotation above cut-in wind speed would likely indicate curtailment, this introduces the need to define a WTPC. Given that the mechanism for stoppage is blade pitching (and braking), using pitch avoids the need to define the WTPC or normal behaviour. This saves a considerable amount of time an effort for the user.

Typical results of the filtering process are shown in **Figure 3**.

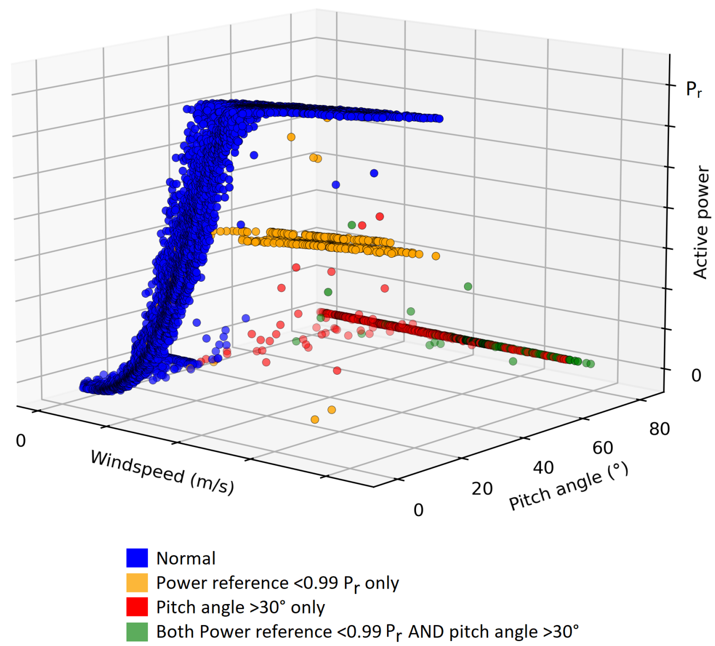

Figure 3: Visualisation of the impacts of explicit and obvious filtering applied to SCADA data from a WT from wind farm B. As shown by the orange, red, and green instances, large amounts of non-normal instances can be filtered out. The blue instances scattered across the figure, as well as the cluster of instances around 25° pitch at 0kW, show that filtering alone is not sufficient and further AD techniques need to be applied.

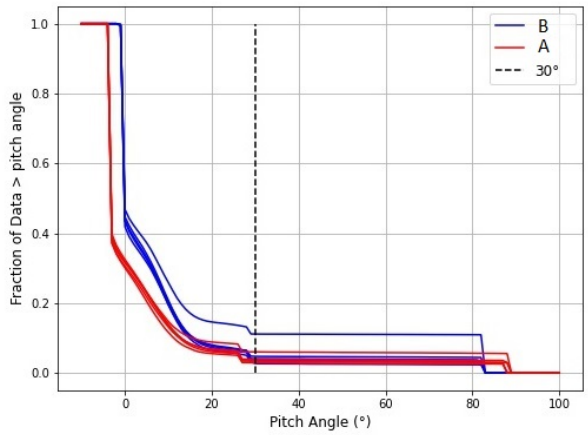

Figure 4: Plot of pitch angle versus fraction of dataset smaller than said pitch angle. Ten WTs from two wind farms are shown. A pitch angle of 30° is indicated, greater than this an instance is labelled as curtailment according to the filtering rules.

2.3 Anomaly Detection Algorithms

Five AD techniques have been chosen, including ”do-nothing”. Each method is named using the format *approach.technique* and all method names are summarized in **Table 1**. Their theory and how they are applied is described below.

Table 1: Names of AD methods used in analysis.

	Approach		
Technique	**Unfiltered**	**Filtered**	**Split**
base (none)	unfiltered.base	filtered.base	split.base
iForest	unfiltered.iForest	filtered.iForest	split.iForest
GMM	unfiltered.GMM	filtered.GMM	split.GMM
LOF	unfiltered.LOF	filtered.LOF	split.LOF
kNN	N.A.	N.A.	split.kNN

Where data is scaled, this will be performed using robust scaling. A robust scaler is used under the assumption that the data contains outliers. Scaling is performed via *Scikit* module *preprocessing.RobustScaler*[16]. This is a standardization method which scales data using the interquartile range (IQR, see **Section 2.3.3** for a description of IQR). As such, it is more robust to outliers than Min-Max scalers. Min-Max scalers scale all values between 0 and 1. A spuriously large values value, say a reading erroneously 10 times larger than the true value, would lead to all the other values being crushed into a small range during Min-Max scaling.

2.3.1 Do Nothing - Base

This "technique" represents no action being taken. Note that this is still after filtering, hence the *unfiltered.base* method uses all data and the *filtered.base* method uses only filtered data. The *split.base* method is identical to the *filtered.base* method as, by the nature of this being the base method, no further action can be taken.

2.3.2 iForest

Isolation Forest (iForest) is a relatively new technique in the field of AD and was developed and introduced by [17]. iForests are ensembles of "iTrees". These iTrees are an evolution of binary search trees in that they partition data, however, in iForests the splits are made at random. For multiple features, a random feature is selected followed by the random split. The shorter an instance's path length the more likely an instance is to be an anomaly. The underlying theory being that normal instances occur in the same region as other instances, hence requiring many splits to be isolated. Conversely, anomalies exist in sparsely populated regions and so require far fewer splits to be isolated. This is shown in **Figure 5**.

iTrees work through the dataset in samples, rather than process the entire dataset in one step. According to the authors, a small sample sizes allows iForest to overcome problems of masking and swamping. A sample size of 2^8 (256) is recommended.

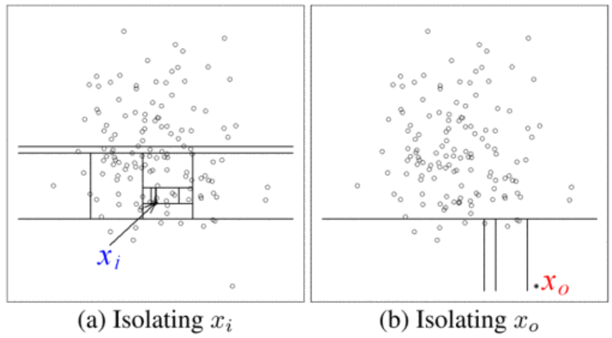

Figure 5: Visualisation of the iForest technique isolating a normal instance (a) and an anomalous instance (b). Figure from [17].

As per the original paper, iForests derive the anomaly score for an instance x from its averaged path length, $h(x)$. The anomaly score for an instance x, given a set of n instances, is given as:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{1}$$

Where $c(n)$ is the average of $h(x)$ given n and $E(h(x))$ is the average path length of x across all the iTrees. If s is close to 1, the instance is likely to be an anomaly, if less than 0.5, then it is likely not an anomaly. According to the authors, if all instances return an s value of approximately 0.5 then the set does not have any anomalies.

The main advantage of iForests are that they are extremely quick to run. iForest does not perform any profiling, distance, density, or co-variance calculations so the computational power required is tiny relative to other AD techniques. iForest processing time can be further reduced by imposing a height limit upon the iTrees, beyond this an instance would be considered normal.

iForest will be implemented in Python using *Sklearn-ensemble-IsolationForest*, an algorithm from *Scikit-Learn* [16]. Default settings of 100 trees and a sample size of 2^8 will be used, along with no assumption for percentage of contamination. Note that no data scaling will be required prior to applying the iForest technique. With reference to the *Split* approach, there appears to be no way to further utilize the initially-anomalous data. As such, *split.iForest* and *filtered.iForest* methods are identical.

2.3.3 Gaussian Mixture Modelling

Gaussian Mixture Model (GMM) is a model-based technique that can be adapted for AD. The underlying assumption is that the model being analysed is composed of k Gaussian distributions. Normal instances are generated from these Gaussian distributions whilst anomalies are not and so occur in low probability spaces [3].

As per [18], for p variables, each distribution has a mean for each variable, $\mu = (\mu_1, \mu_2, ...\mu_p)$. Each distribution will also have a covariance matrix, containing covariance values for each pair of variables. The means and covariance matrix values are estimated using Maximum Likelihood Estimates.

Anomaly scores are simply the distance from the instance to the mean. This is usually Euclidean distance, however some methods use Mahalanobis distance. As such, each instance has as many anomaly scores as there are Gaussian distributions assumed. There appears to be no definitive way to convert these anomaly scores to classifications but many have been proposed. Reference [19] suggests assigning any score greater than 3 standard deviations away from the mean score as an anomaly. Reference [20] suggests using the *box-plot rule*. This is the range between the whiskers of a box plot, equivalent to p_{lower} and p_{upper} given as:

$$[p_{lower}, p_{upper}] = [Q_{25} - 1.5 \times IQR, Q_{75} + 1.5 \times IQR] \tag{2}$$

Where IQR is the interquartile range, equivalent to the difference of Q_{75} and Q_{25}.

The number of mixtures used in the model is typically determined using BIC (Bayesian Information Criterion) curves. The theories behind BIC curves are not explored here. The number of mixtures producing the lowest BIC scores should be used, however, this is not an absolute rule and often the curve "elbow" will be used. The elbow is, essentially, the point of diminishing returns. Adding more mixtures no longer results in a similar drop in BIC score as it did for adding mixtures previously, even if some small drop is witnessed [21].

GMM will be implemented in Python using *Sklearn.mixture.GaussianMixture*, an algorithm from *Scikit-Learn* [16]. The number of mixtures to use has been determined using the BIC curve process on a reduced set of features. These were: all wind speed features, temperature, and mean active power (see **Section 3** for full description of SCADA features). Ten randomly selected turbines were selected (5 from each wind farm, see **Section 3**) and their curves determined. The curve elbow was found to be at 3 mixtures as shown in **Figure 6**.

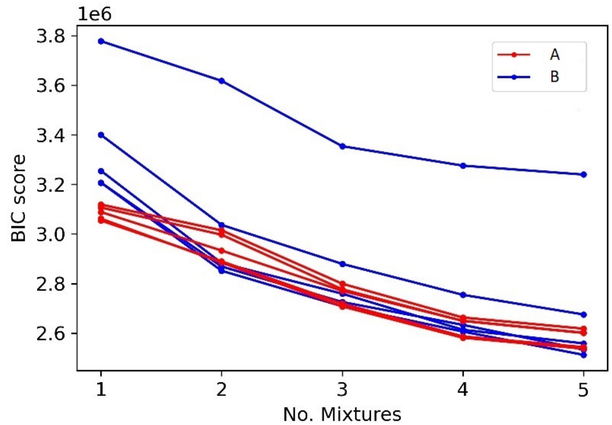

Figure 6: BIC scores versus number of Gaussian mixtures assumed for 10 randomly selected turbines (5 from each wind farm). Raw SCADA data was used.

Unlike the other techniques, *Sklearn-mixture-GaussianMixture* does not have a build-in classification method; however, it does have the *score_samples* method, which calculates probabilities of each instance belonging to each of the 3 mixtures. For each instance, the maximum of the 3 likelihoods will be taken. The box-plot rule will then be applied to determine normal and anomalous data.

In the case of *unfiltered.GMM* all data will be used. For *filtered.GMM* only filtered data will be used. For *split.GMM*, an altogether different approach will be taken. The SCADA sets will be split into 2 groups based on the filtering, an assumed-normal group and an assumed-anomalous group. From each group, the data will be further split into train and test data on an 80-20% split. The training data will be used to train a GMM per group. The testing data will be fed into **both** models to see which model they have greater affinity. The results are recorded. This entire process is repeated 4 more times with different portions of the data forming the test data (i.e. cross-validated) until every instance has been assigned as being more likely normal or more likely anomalous.

2.3.4 LOF

Local Outlier Factor (LOF) is a relative-density based technique proposed by [22]. LOF is concerned with assigning a degree of outlier-ness to an instance, rather than a hard label of 0 or 1. The LOF method uses a local approach, rather than global, hence it is useful in applications where non-homogeneous densities are acceptable.

The LOF method first requires the number of nearest neighbours, $MinPts$, to be chosen. Following this, the LOF of a point x is given as:

$$LOF_{MinPts}(x) = \frac{\sum_{o \in N_{MinPts(x)}} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(x)}}{|N_{MinPts}(x)|} \tag{3}$$

To interpret the above, the LOF of an instance is the ratio of that instance's local reach density, lrd, to that of the average lrd of its $MinPts$ neighbours. The lrd of instance x is given as:

$$lrd_{MinPts}(x) = 1/\left[\frac{\sum_{o \in N_{MinPts(x)}} reach\text{-}dist_{MinPts}(x, o)}{|N_{MinPts}(x)|}\right] \tag{4}$$

The *reach-dist* is essentially the euclidean distance between two points, however, if the distance becomes very small, this reverts to a set value. This minor change has the effect of smoothing LOF scores and making differentiation between inliers and outliers easier.

In **Figure 7** the lrd of an anomalous instance p with $MinPts$ value of 3 is illustrated. The combined reach distances of p to its 3 nearest neighbours is clearly far larger than the same metric for those 3 neighbours. This results in a low lrd compared to the neighbours, i.e. a large volume is required to capture the specified nearest neighbours. Such a large relative difference in lrd then results in a high value for $LOF_{MinPts}(p)$, as per **Equation 3**.

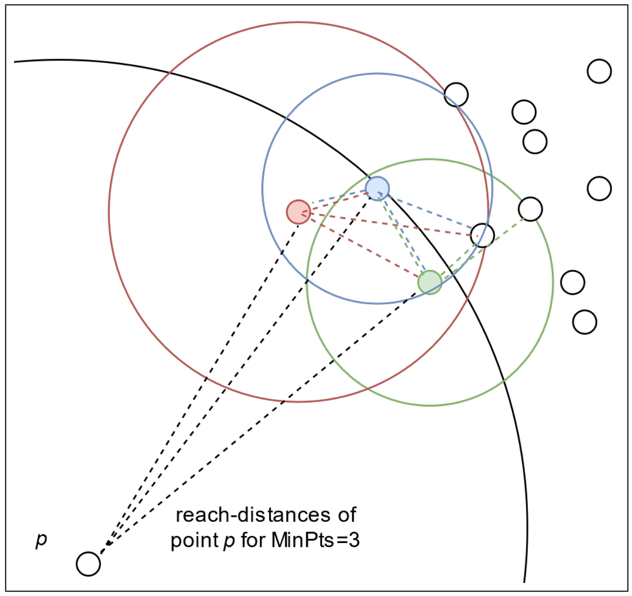

Figure 7: LOF in action, adapted from [22]. The 3 nearest neighbours of p are shown by the green, blue, and red points. Each of their 3 nearest neighbours are shown with the dashed lines of the same colours. Instance p is clearly an outlier with respect to the rest of the set due to its large reach and resulting low local reachability density.

According to [22], most instances should have a LOF value of 1. If the LOF value is greater than 1, it is likely an outlier and sits in a sparsely populated region. A LOF of less than 1 and it is likely an inlier in a densely populated region. There appears to be no definitive threshold LOF value to label an instance an outlier. The value that will be used in this study is 1.5, the default used by the Python module *sklearn-neighbors-LocalOutlierFactor*, as discussed below. Similarly, there is no single value for $MinPts$ recommended by the authors of the LOF method, although no smaller than 10 is recommended to avoid statistical fluctuations. An appropriate value is dependent on the dataset being investigated, if too large a value is used then clusters that are small, but valid, are unfairly treated.

LOF will be implemented in Python using *sklearn-neighbors-LocalOutlierFactor*, an algorithm from *Scikit-Learn* [16]. A $MinPts$ value of 700 will be used. This is based upon a brief analysis of $MinPts$ with removal rates for randomly selected turbines from both wind farms. Removal rates were found to settle (in terms of visual impact on the WTPC) at approximately $MinPts = 400$, with no change between this value and $MinPts = 1000$. It may seem like an unfair advantage has been gotten by choosing the most favourable value of $MinPts$; however, note that all the data is unlabelled and that choice of $MinPts$ here is based on removal rates, not accuracy. The fact that a conservative value of $MinPts$ was "manually" found is a matter of convenience to save coding and computational time, as this process could be automated.

It is important to note that multidimensional data must be scaled prior to use in LOF so as to account for the different dimensions of features in distance calculations.

LOF has no training stage and has only a fit-predict method. Note that the nature of LOF is to make an astronomical number of calculations. As such, for high dimensional data, users of this technique may need to reduce the number of dimensions used to make the computation time feasible. With reference to the *Split* approach, there appears to be no way to further utilize the initially-anomalous data. As such, *split.LOF* and *filtered.LOF* methods are identical.

2.3.5 k Nearest Neighbours

k Nearest Neighbours (kNN) is a nearest-neighbours technique. This technique is a classification method but is adapted here for AD. kNN dates back to 1951 and was introduced in [23]. The underlying assumption is that normal instances occur close to other normal instances, and the same is true for anomalies. As such, this method requires that labelled instances already exist so that new, unlabelled, instances can be classified.

The kNN technique first involves specifying the number of nearest neighbours to consider, k. A distance metric, such as Euclidean, is then specified. An unlabelled instance is then considered against a labelled dataset. Its k nearest neighbours are then determined. The unlabelled instance is then assigned to whichever class comprises the majority of neighbours. This is illustrated in **Figure 8**. Variations upon this technique can be to add weightings to certain instances, for example if these instances have been chosen by experts.

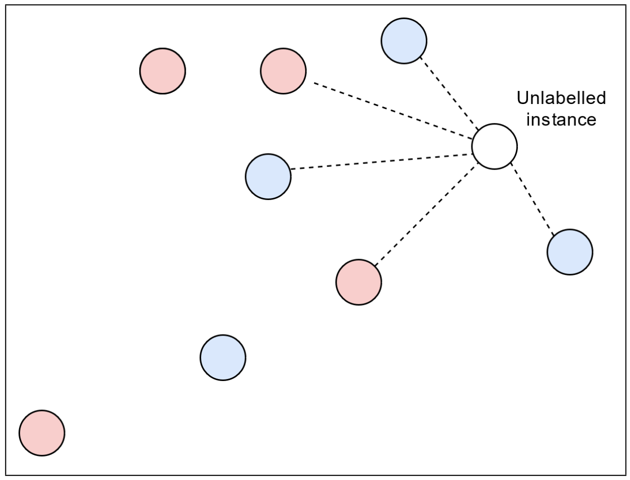

Figure 8: Assigning an unlabelled instance a class via kNN using $k = 5$. The new instance would be assigned to the blue class.

kNN will be implemented in Python using *Sklearn.neighbors.KNeighborsClassifier*, an algorithm from *Scikit-Learn* [16]. This technique is exclusive to the *Split* approach as it requires classes to be assigned. First, the best k value will be chosen from options of 3, 5, 7, and 9. This will be determined by assigning all data classes based upon filtering. Then, the data will be split into train and test data on an 80-20% split. The value of k that produces the highest accuracy for the test set will then be used.

Split.kNN will then be implemented in a similar fashion as *split.GMM*. All data will be given labels based upon filtering. Eighty percent of each group (normal and anomalous) will then be used to train a classifier. The classes of remaining remaining 20% will then be predicted using the k value determined above. This will be repeated 4 further times, with different portions of the data forming training and testing sets, to determine a classification for every instance, i.e. 5-fold cross validation.

Note that there is a need to scale features when using kNN due to the use of distance metrics.

2.4 Evaluation

To paraphrase from [8], the goal of AD in cleaning the WTPC is "to reject potential [anomalies] whilst broadly retaining the statistical characteristics of the WTPC, in particular the mean values of the measurements". As such, the efficacy of each AD method is assessed by three measures:

- Prediction error on the "cleaned" dataset, e_M.
- Elimination rate, i.e. percentage of data removed, γ.
- Change to wind speed interquartile range, δIQR_u.

Prediction error, e_M: A neural network (NN) will be fitted to each AD method's cleaned WTPC using a standard train-test split. The prediction error will then be found and converted to a percentage of the P_r to allow for comparison between turbines of different ratings. This is calculated as follows:

$$e_M = \frac{RMSE(u,h)}{P_r} \times 100\% \tag{5}$$

Where P_r is the rating of the WT, and $RMSE(u,h)$ is the test-set prediction root mean square error of a neural network trained on the training set, given by:

$$RMSE(u,h) = \sqrt{\frac{\sum_{n}^{i=1} \left(h(u_i) - p_i\right)^2}{n}} \tag{6}$$

Where n is the total number of instances in the prediction set, $h(u_i)$ is the predicted value of power for instance i with wind speed u, and p_i is the actual power for that instance.

A simple NN will be constructed using *TensorFlow* and run using *Google Colab's* TPUs (Intel® Xeon® CPU @ 2.30GHz, 64 GB RAM). A shallow configuration is chosen to allow for testing within a reasonable time frame. With 10 unique methods to run across 20 WT SCADA sets, this results in constructing, training, and testing 200 NNs. The NN will consist of 1 input layer, 1 output layer, and 2 dense inner layers each with 64 neurons. Activation functions of the middle layers will be Rectified Linear Units. Given that the NN has only 1 input (wind speed) and 1 output (power), and that the focus of results is in the comparison between methods, such a shallow NN is deemed appropriate and optimization of each NN is unnecessary. Obtaining results for a large number of turbines, therefore determining more reliable averages, is preferable to highly accurate results for a small numbers of turbines.

Elimination rate, γ: As per [10], elimination rate is defined as the percentage of data removed:

$$\gamma = \frac{N_b - N_a}{N_b} \times 100\% \tag{7}$$

Where N_b is the original number of instances in the SCADA set, and N_a is the number of instances after the AD method has removed the anomalies it has identified. If methods have similar prediction errors, the method that retains more data would clearly be superior. Conversely, any methods that rig the system by removing high percentages of data, so as to only make predictions over a very small range, can be identified as performing poorly.

Change to wind speed interquartile range, δIQR_u: The spread of the wind speed feature will be recorded both before and after anomaly removal via IQR. As per **Section 2.3.3**, IQR is simply the difference of Q_{75} and Q_{25}. Note: this is different from the box plot rule which includes the whiskers of the box plot. Change to IQR_u is calculated as:

$$\delta IQR_u = \frac{IQR_{u,b} - IQR_{u,a}}{IQR_{u,b}} \times 100\% \tag{8}$$

Where $IQR_{u,b}$ is the IQR of wind speed before the AD process, and $IQR_{u,a}$ is the IQR of wind speed after the AD process. IQR is chosen as it is robust to outliers than simply taking the absolute range as an anomalous measurement might read an order of magnitude higher, thus the change to absolute range would be drastic if this instance was (correctly) removed. By incorporating this IQR change and percentage removal a better representation of the method's quality can be achieved. Given that the AD methods will be applied in an unsupervised way, with error being calculated using a dataset the AD method has been applied to, in theory, a method could achieve a low error by reducing the SCADA set down to a narrow, predictable band. Incorporating data and wind speed variability retention allows for better understanding of a method's appropriateness for AD in WTPC cleaning.

3 Data Description

3.1 Wind Farms

SCADA data from 2 offshore wind farms has been provided. These are referred to as wind farms *A* and *B*. The characteristics of both wind farms are summarised in **Table 2**.

Table 2: Characteristics of the 2 wind farms used in the analysis.

Wind farm	A	B
Location	North Sea, UK	Northern Europe
Distance from shore	< 25km	< 25km
No. turbines	< 50	> 50
Turbines used in analysis	10, randomly selected	10, randomly selected
SCADA duration	24 months	18 months
SCADA frequency	ten-minute	ten-minute

3.2 Feature Engineering

The features provided in the SCADA sets is shown in **Table 3**. Note that wind speeds are from nacelle anemometers for all WTs. Timestamps were not included in the AD process, each instance is considered in isolation. All features of "Yaw" and "RPM" were also dropped. These features are ineffective for determining curtailment (see **Section 2.2.1**) and are not useful in generalizing the turbine power output for a site's given wind regime. Air density was not calculated due to a lack of pressure readings; regardless, variations in air density have little impact upon the WTPC [24].

All other features are included in the AD process (including the treatment of missing data). No further features were present in the SCADA set. The author notes that SCADA features will vary between SCADA systems and some may contain more explicit indications of operating mode.

Table 3: SCADA features with indication of use in Anomaly Detection. Blanks indicate that the feature was not present in the SCADA data.

Feature	Mean	Min	Max	Std. Dev.
Timestamp	N			
wind speed	Y	Y	Y	Y
Yaw	N	N	N	N
Pitch	Y	N	N	N
RPM	N	N	N	N
Power ref.	Y			
Power	Y	N	N	N
Temperature	Y			
Operation time	Y			
Fault	Y			

4 Results and Discussion

All SCADA datasets were cleaned of missing data instances via listwise deletion prior to performing filtering and AD. The percentage of the SCADA that was erroneous was less than 5% in every case and, therefore, less than the threshold value for being considered negligible. For wind farms A and B, erroneous instances made up an average of 0.23% and 0.06% respectively. This excludes instances in which all features were missing. As such, no substitution of SCADA sets was necessary.

Each anomaly detection method was applied to each of the 20 turbine SCADA sets. The impacts of each AD method upon the WTPC are visualized in **Figure 9** with colours indicating instances as being determined normal, anomalous, or filtered out by the AD method. Note that kNN technique only applies to the *split* approach and that iForest and LOF techniques are identical between *filtered* and *split* approaches, hence these results are not shown.

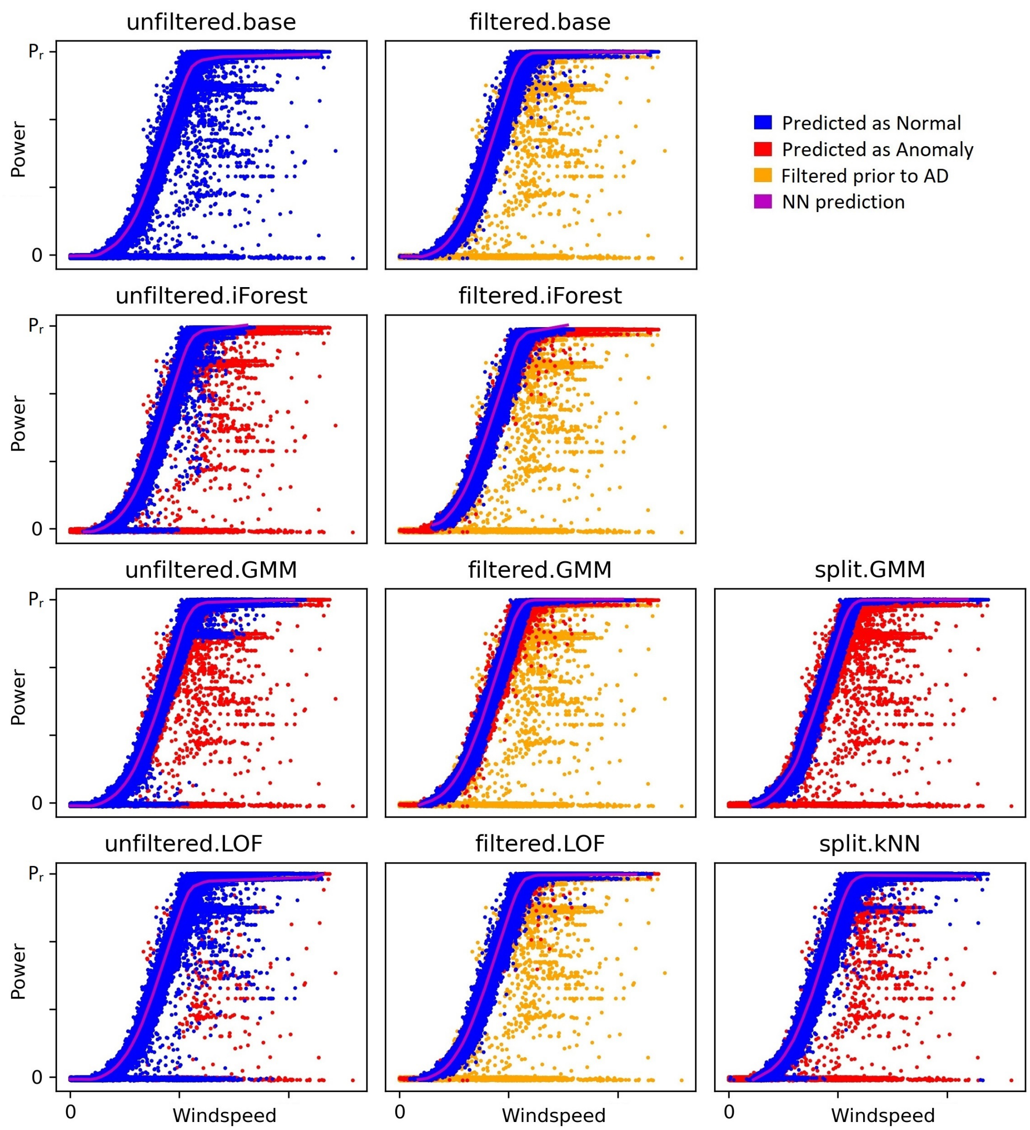

Figure 9: Visualisation of the impacts of different AD techniques and approaches when applied to a WT from wind farm A. As per the legend, different colours indicate whether instances were filtered out prior to AD or later labelled as "normal" or "anomalous" by the AD technique. In each subplot, the test-set predictions of the NN used for evaluation is shown by the purple line.

From an inspection of **Figure 9**, the differences in performances between techniques and approaches can be seen. Comparing *unfiltered.base* and *filtered.base*, it is clear that applying simple filtering rules greatly cleaned the WTPC. However, many Type 3 (random) anomalies still exist, as does a group of Type 2 (curtailment) instances immediately under the flat part of the WTPC. This group of instances is most visible in the LOF methods. Inspecting the *unfiltered* methods, it appears none managed to deal with the Type 1 (stoppage) anomalies well, especially not at lower wind speeds. Additionally, none managed to remove the Type 2 group discussed previously.

Looking at the iForest results, this technique did not handle the flat, P_r part of the WTPC well. This technique does appear to have dealt with Type 3 anomalies reasonably well. Moving on to the GMM technique, all three variations appear to have removed the majority of Type 3 anomalies. *Unfiltered.GMM* appears to have struggled at the knee of the WTPC and not removed a group of Type 2 anomalies at approximately 80% of P_r. The *filtered* and *split* variations of GMM are the only methods which removed the Type 2 group at approximately immediately under the flat part of the WTPC. Inspecting the LOF technique, it appears to have been very conservative and only removed the most isolated of instances. Stacking was clearly an issue for *unfiltered.LOF*. *Filtered.LOF* only eliminated a handful of instances after filtering occurred. Finally, kNN appears to have performed worse than *filtered.base*. It appears initially anomalous points have been reclassified to normal, rather than the other way around.

As per the methodology, prediction error as a percentage of P_r (e_M), elimination rate (γ), and interquartile range of the wind speed feature (IQR_u) were determined for each cleaned dataset, along with a percentage change (δ) from the base case, where appropriate. The results have been averaged across the 20 turbines and are shown in **Table 4** and **Table 5**.

Table 4: Results by method. Note that *split* and *filtered* approaches are identical for the techniques of base, iForest, and LOF. Elimination rate includes both filtered data and that detected as anomalous.

	e_M				IQR_u		
Method	pre	post	δe_M (%)	γ	pre	post	δIQR_u (%)
unfiltered.base	11.35	NA	NA	NA	5.86	NA	NA
unfiltered.iForest	11.35	4.89	56.95	16.02	5.86	5.16	11.94
unfiltered.GMM	11.35	5.53	51.23	6.20	5.86	5.77	1.60
unfiltered.LOF	11.35	8.06	28.94	1.67	5.86	5.84	0.47
filtered.base	11.35	3.78	66.71	10.97	5.86	5.54	5.50
filtered.iForest	11.35	3.73	67.17	27.47	5.86	4.78	18.51
filtered.GMM	11.35	3.38	70.25	15.13	5.86	5.50	6.29
filtered.LOF	11.35	3.72	67.19	11.74	5.86	5.52	5.91
split.GMM	11.35	3.39	70.11	14.11	5.86	5.48	6.62
split.kNN	11.35	4.05	64.35	7.99	5.86	5.62	4.12

Table 5: Subsequent rates of evaluation metrics, calculated by method. Note that *split* and *filtered* approaches are identical for the techniques of base, iForest, and LOF.

Method	δe_M $/\gamma$	δe_M $/\delta IQR_u$	δIQR_u $/\gamma$
unfiltered.base	NA	NA	NA
unfiltered.iForest	3.55	4.77	0.75
unfiltered.GMM	8.26	31.97	0.26
unfiltered.LOF	17.33	61.73	0.28
filtered.base	6.08	12.12	0.50
filtered.iForest	2.45	3.63	0.67
filtered.GMM	4.64	11.16	0.42
filtered.LOF	5.72	11.36	0.50
split.GMM	4.97	10.58	0.47
split.kNN	8.06	15.61	0.52

4.1 Impact of Filtering

The impact of filtering alone can be isolated by comparing *unfiltered.base* and *filtered.base*. From **Figure 9**, it is clear that a large amount of data was removed, this averaged 10.97% for these wind farms. This resulted in a substantial improvement to prediction error, e_M, with an average decrease of 66.7% from *unfiltered.base* to *fil-*

tered.base. This occurred at a minimal change to IQR_u with only a 5% reduction. The average e_M of *unfiltered* and *filtered* methods was 7.46% and 3.65% respectively. Clearly, filtering was extremely beneficial to prediction error due to the reduced influence of stacked anomalies.

4.2 Error and Elimination Rate

The relationship between e_M and γ is explored in **Figure 10**. Elimination rate includes both data filtered out prior to AD as well as data labelled as anomalous by the AD method. A polynomial line of best fit has been added.

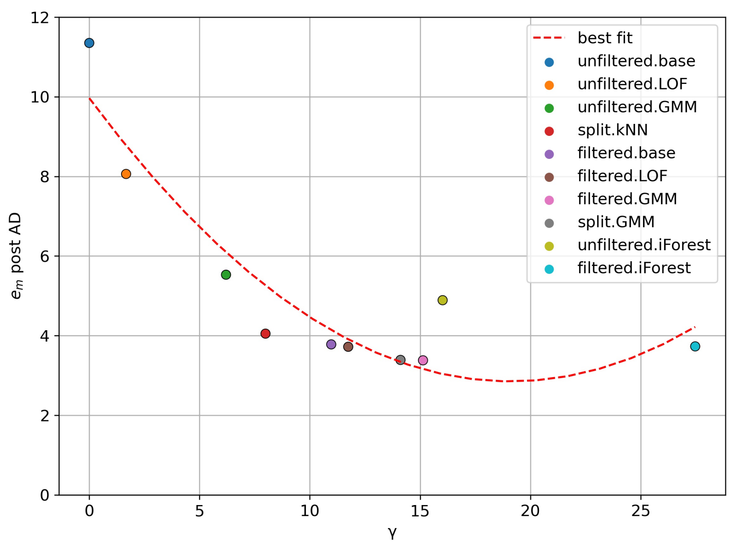

Figure 10: Results: Comparison of AD method averaged performance, γ versus e_m. On the x-axis are the percentages of SCADA data removed, γ, this includes both filtered and that eliminated by the AD technique. On the y-axis are prediction errors of the "cleaned" SCADA data, e_m, presented as a percentage of P_r.

Looking at these results, it is initially difficult to separate which is the greatest driver of reducing error, the approach or the method. From **Figure 10**, there appears to be a correlation between prediction error and amount of data removed, as shown by the line of best fit. This is intuitive, as removing anomalies should increase accuracy and will reduce the amount of data remaining. However, the gradient of the line of best fit shows that not all anomalies have equal influence on error. Using the line of best fit, removing 10% of instances reduces error by 55.2% compared to retaining all instances. Removing a further 10% of data only reduces error by a further 16.2% (total reduction of 71.2%). Going beyond a γ of 17% appears to have no substantial improvement on prediction error with the final method of *filtered.iForest* coming in at a γ of 27.5% and an e_M of 3.73%. The high γ rates of the iForest methods appears to be due to classifying large portions of the rated (flat) part of the WTPC as anomalous. Note that this is a simple part of the curve to make power predictions for, hence the generally low e_m is more impressive.

In terms of improvement to prediction error per percentage of data removed ($\delta e_M/\gamma$), the *unfiltered.LOF* method's performance is, by some margin, the best at 17.3 (see **Table 4**). This is slightly more than double that of the next best method, which is *unfiltered.GMM* at 8.26. However, assessment of *unfiltered.LOF*'s performance must be tempered by its poor absolute prediction error. Except for the base-case itself (*unfiltered.base*), *unfiltered.LOF* ranks as the worst method for e_M. At a high level, this trade-off suggests that different requirements of the SCADA data can determine the choice in AD method. For example, if outright smallest prediction error was preferred, no matter the elimination rate, *filtered.GMM* may be suitable. If the preference was to retain data as far as possible then *unfiltered.LOF* may be selected.

4.3 Error and wind speed Variability

A similar trend can be seen for e_M and change to wind speed variability, δIQR_u, as shown in **Figure 11**. This equates to an approximately linear relationship between γ and δIQR_u.

From the results for *filtered.base* it appears that some reduction in IQR_u is acceptable. This method reduces IQR_u by 5.5% and, looking at the WTPC in **Figure 9**, it appears that this method has high precision but low recall. *Precision* is the measure of how many instances labelled as "anomaly" truly were anomalies. *Recall* is the total amount of anomalies identified as a portion of all the anomalies i.e. true positives versus the set of true positives and false negatives.

Comparing *filtered* and *unfiltered* methods, it is clear that the latter have the lowest δIQR_u. The *unfiltered* methods average a IQR_u of 5.66 (change of 3.5%), whilst *filtered* methods have an average of 5.33 (change of 9.1%). The *split* methods fair marginally better than *filtered* methods at an average of 5.39 (change of 8.14%).

Significantly further along the x-axis we have the two iForest methods. These methods have comparable prediction error results as others. Looking at their WTPCs in **Figure 9**, it seems the high δIQR for these SCADA sets is a result of labelling high and very low wind speeds as anomalous.

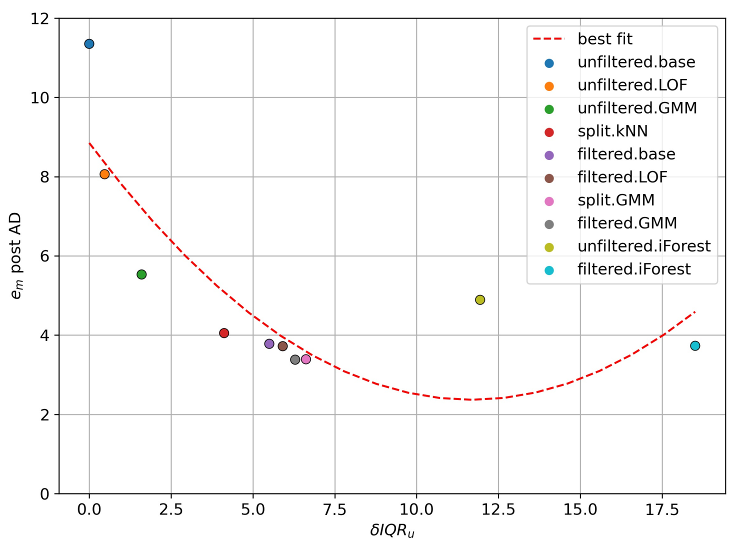

Figure 11: Results: Comparison of AD method averaged performance, δIQR_u versus e_m. On the x-axis are the percentage changes to windspeed IQR of each method, before and after it was applied, δIQR_u. On the y-axis are prediction errors of the "cleaned" SCADA data, e_m, presented as a percentage of P_r.

4.4 Advantages to Splitting?

As discussed previously, it is clear that from an absolute error perspective *filtered* produces far better results than *unfiltered*, but we must also consider *split*. As discussed in the methodology, the methods of *base, iForest*, and *LOF* are the same as for *split* and *filtered* so cannot be used to isolate the impact of *split*. *GMM* differs between *split* and *filtered*, and *kNN* is unique to *split*.

For GMM, *split* and *filtered* had broadly similar results. This is especially clear when looking at **Figure 10**. Whilst the 2 approaches had similar results in terms of error and IQR_u, *split* provided a very slight advantage with γ. *Filtered.GMM* removed 15.13% of data, marginally more than *split.GMM* with 14.11%. As such, *split.GMM* may be appropriate in applications where high data retention is preferred.

For *split.kNN*, this method appeared to have good performance in terms of e_M/γ and e_M/IQR_u but ultimately ranked poorly in terms of absolute e_M. A reason for this can be found by comparing *split.KNN*'s initial class assignments (i.e. the same as *filtered.base*) against final assignments, as shown in **Figure 12**. It appears the method has incorrectly flipped many instances back from anomalous to normal so increasing γ but reducing e_M. For the example WT shown in the figure, some 21,000 instances were initially labelled as anomalous. Of these, approximately 9,000 (42%) flipped class to ”normal”. Approximately 4,000 initially ”normal” instances were assigned as ”anomalous”. This is likely due to unbalanced class numbers whilst new instances were being assigned. In the example, there were 3.5 times more initial ”normal” instances than ”anomalies”. This challenge should be accounted for in future research, potentially through use of weightings.

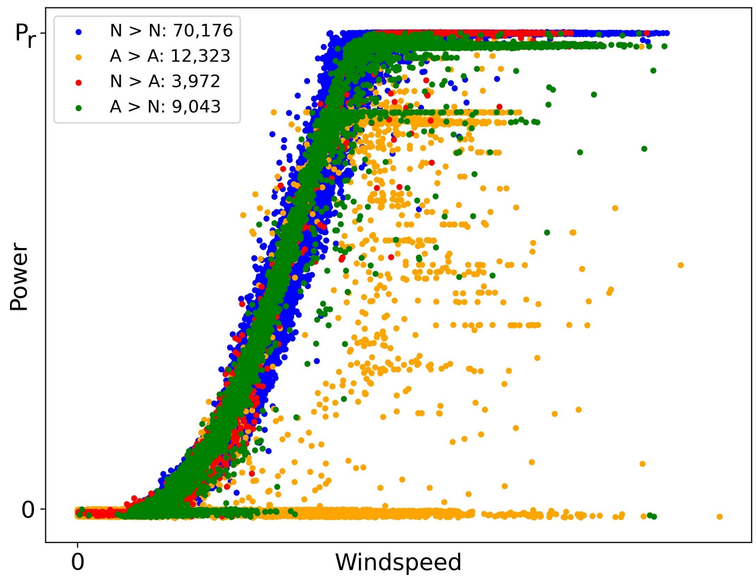

Figure 12: Visualisation of the change of classes from initial to final classification for the *split.kNN* method when applied to a WT from wind farm *A*. For the legend, "N" refers to "normal", "A" for "anomalous", and ">" the change from initial to final classification. The number of instances belonging to each category is shown in the legend.

4.5 Choice of AD Method

As discussed previously, the AD methods have been shown to have different performance characteristics in terms of prediction error, elimination rate, and change to the wind speed feature characteristics. The choice of AD method would therefore depend on the what the user of the SCADA data wishes to achieve. Two distinct scenarios are: (a) the lowest prediction error possible without unnecessary removal of SCADA data and change to wind speed variability; and (b) improving prediction error whilst maintaining as much SCADA data as possible.

For scenario (a), we start by examining *filtered.base*. As discussed previously, it appears likely that this method has high precision but low recall. As such, these results are used as a starting point from which to compare the other methods. This is shown in **Table 6**. Using this table, we can remove any methods with lower elimination rates on the ground of being less effective. These are: *unfiltered.base, unfiltered.GMM, unfil-*

Table 6: Results by method and relative to the *filtered.base* method. Note that *split* and *filtered* approaches are identical for the techniques of base, iForest, and LOF. Elimination rate includes both filtered data and that detected as anomalous.

Method	e_m	e_m relative to filtered.base	γ	γ relative to filtered.base	$IQR_{u,a}$	$IQR_{u,a}$ relative to filtered.base
unfiltered.base	11.35	-7.57	0.00	-10.97	5.86	0.32
unfiltered.iForest	4.89	-1.11	16.02	5.06	5.16	-0.38
unfiltered.GMM	5.53	-1.76	6.20	-4.77	5.77	0.23
unfiltered.LOF	8.06	-4.29	1.67	-9.30	5.84	0.30
filtered.base	3.78	-	10.97	-	5.54	-
filtered.iForest	3.73	0.05	27.47	16.50	4.78	-0.76
filtered.GMM	3.38	0.40	15.13	4.16	5.50	-0.05
filtered.LOF	3.72	0.05	11.74	0.77	5.52	-0.02
split.GMM	3.39	0.38	14.11	3.14	5.48	-0.07
split.kNN	4.05	-0.27	7.99	-2.98	5.62	0.08

tered.LOF, and *split.kNN*. Whilst this is not a guarantee that the remaining methods eliminated the same instances as *filtered.base*, looking at **Figure 9**, this does appear to be the case. Of the remaining 6 methods, the 2 iForest methods, (*filtered.iForest* and *unfiltered.iForest*), can be removed as they have low rates of $\delta e_m/\delta IQR_u$ compared to the other 4 methods, as per **Table 5**.

This leaves 4 unique methods in consideration. In order of e_M, these are: *filtered.base* itself (3.78%), *filtered.LOF* (3.72%), *split.GMM* (3.39%), and *filtered.GMM* (3.38%). The 2 GMM methods were clearly the more accurate and both are recommended for scenario (a). As discussed in **Section 4.4**, it is slightly more advantageous to use *split.GMM* over *filtered.GMM*; however, it should be noted that *split.GMM* was considerably more challenging to implement than *filtered.GMM* due to the cross validations required (see **Section 2.3**). As such, some users may prefer *filtered.GMM*.

For scenario (b), we are concerned with methods with high rates of $\delta e_M/\delta IQR_u$ i.e. improvement to prediction error at minimal cost to the wind speed feature. The 3 methods with the highest rates in this category are: *unfiltered.LOF* (61.73), *unfiltered.GMM* (31.97), and *split.kNN* (15.61), according to **Table 5**. From **Figure 9**, *unfiltered.LOF* was clearly overly conservative and can be eliminated. The remaining 2 methods showed similar results for their WTPCs; however, *unfiltered.GMM* is the better choice with a δIQR_u approximately half that of *split.kNN* whilst still maintaining a marked decrease in error from the base case (δe_M of 51.23%).

5 Conclusion

Three pre-processing approaches have been compared, along with 5 anomaly detection techniques for a total of 10 unique AD methods. These methods have been applied to SCADA data from 2 different wind farms for a total of 20 turbines. The efficacy of the AD methods have been studied in terms of improvements to power prediction error, amounts of data removed, and ability to retain the underlying statistical characteristics of the wind speed feature. From this, and with respect to the SCADA sets used in the study, the following conclusions are drawn:

- It is beneficial to pre-process the SCADA data by filtering out obvious anomalies and explicit instances of faults/curtailments prior to applying anomaly detection techniques.
- A pitch angle of >30° is proven to be a reasonable threshold for the above pre-processing.
- All anomalies do not have equal impact upon error. The rate of prediction error reduction reduces as more data is removed and anomalies become harder to detect.
- The AD method of choice is dependent on the application, with some methods achieving lower error at the cost of increasing percentages of data removal and reduction in wind speed variability.
- The GMM technique is shown as an effective method to significantly reduce error whilst maintaining statistical characteristics of wind speed data. This is especially so when combined with pre-processing anomalies, in which error reduces by more than 70% compared to no pre-processing and no anomaly detection technique.

- The *split.GMM* method appear to maintain marginally more data than its *filtered* counterpart, however, the increased complexity in implementing this method may make it undesirable.

Additionally, the importance of proper treatment of SCADA data regarding missing data has been raised. Given that SCADA data is the basis of so many findings, conclusions, and concepts it is paramount that this treatment is discussed so that all that follows can be relied upon or can be replicated by others.

6 Acknowledgments

The authors thank the anonymous data provider for providing the SCADA data for both wind farms. This research was funded by the EPSRC Doctoral Training Partnership (EP/R513222/1 ENG).

References

[1] European Commission. *An EU Strategy to harness the potential of offshore renewable energy for a climate neutral future*. Tech. rep. European Commission, 2020, p. 27. arXiv: `arXiv:1011.1669v3`. URL: `https://ec.europa.eu/commission/presscorner/detail/en/IP_20_2096`.

[2] UK Government. *Press Release: New plans to make UK world leader in green energy*. Oct. 2019. URL: `https://www.gov.uk/government/news/new-plans-to-make-uk-world-leader-in-green-energy`.

[3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. *Anomaly detection: A survey*. 2009. DOI: `10.1145/1541880.1541882`.

[4] Zi Lin, Xiaolei Liu, and Maurizio Collu. "Wind power prediction based on high-frequency SCADA data along with isolation forest and deep learning neural networks". In: *International Journal of Electrical Power & Energy Systems* 118 (2020), p. 105835.

[5] Ran Bi, Chengke Zhou, and Donald M Hepburn. "Detection and classification of faults in pitch-regulated wind turbine generators using normal behaviour models based on performance curves". In: *Renewable Energy* 105 (2017), pp. 674–688. DOI: https://doi.org/10.1016/j.renene.2016.12.075. URL: https://www.sciencedirect.com/science/article/pii/S0960148116311351.

[6] Huan Long et al. "Image-based abnormal data detection and cleaning algorithm via wind power curve". In: *IEEE Transactions on Sustainable Energy* 11.2 (2019), pp. 938–946.

[7] Xiaojun Shen, Xuejiao Fu, and Chongcheng Zhou. "A combined algorithm for cleaning abnormal data of wind turbine power curve based on change point grouping algorithm and quartile algorithm". In: *IEEE Transactions on Sustainable Energy* 10.1 (2018), pp. 46–54.

[8] Yue Wang et al. "Copula-based model for wind turbine power curve outlier rejection". In: *Wind Energy* 17.11 (2014), pp. 1677–1688.

[9] Le Zheng, Wei Hu, and Yong Min. "Raw wind data preprocessing: a data-mining approach". In: *IEEE Transactions on Sustainable Energy* 6.1 (2014), pp. 11–19.

[10] Yongning Zhao et al. "Data-driven correction approach to refine power curve of wind farm under wind curtailment". In: *IEEE Transactions on Sustainable Energy* 9.1 (Jan. 2018), pp. 95–105. ISSN: 19493029. DOI: 10.1109/TSTE.2017.2717021.

[11] Neil Davis et al. "Ice detection on wind turbines using observed power curve". In: *Wind Energy* 19.6 (2016), pp. 999–1010. ISSN: 1095-4244. DOI: 10.1002/we.1878.

[12] IEC: TC 88 - Wind energy generation. *IEC TR 61400-12-4:2020*. Tech. rep. IEC, 2020.

[13] Lijun Zhang et al. “Ice Detection Model of Wind Turbine Blades Based on Random Forest Classifier”. In: *Energies* 11.10 (2018). ISSN: 1996-1073. DOI: 10.3390/en11102548.

[14] Georgios Alexandros Skrimpas et al. “Detection of icing on wind turbine blades by means of vibration and power curve analysis”. In: *Wind Energy* 19.10 (2016), pp. 1819–1832. DOI: https://doi.org/10.1002/we.1952.

[15] John W Graham, Patricio E Cumsille, and Allison E Shevock. “Methods for handling missing data”. In: *Handbook of Psychology, Second Edition* 2 (2012).

[16] Fabian Pedregosa et al. “Scikit-learn: Machine learning in Python”. In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830.

[17] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. “Isolation forest”. In: *2008 eighth ieee international conference on data mining*. IEEE. 2008, pp. 413–422.

[18] Brad Boehmke and Brandon M. Greenwell. *Hands-On Machine Learning with R*. Boca Raton, FL, USA: CRC Press, 2019. ISBN: 978-1-138-49568-5.

[19] W. A. Shewhart. “Economic Quality Control of Manufactured Product”. In: *Bell System Technical Journal* 9.2 (1930), pp. 364–389. ISSN: 15387305. DOI: 10.1002/j.1538-7305.1930.tb00373.x.

[20] Jorma Laurikkala et al. “Informal identification of outliers in medical data”. In: *Fifth international workshop on intelligent data analysis in medicine and pharmacology*. 2000, pp. 20–24. URL: https://scholar.googleusercontent.com/scholar.bib?q=info:mZbjUlJAXQoJ:scholar.google.com/&output=citation&scisdr=CgXbp3N2ENqZ0IA7nlw:AAGBfm0AAAAAYH0-

hlyr8srXhooBul_m-gqWthofgOu3&scisig=AAGBfm0AAAAAYH0-htKw6xhcqO5qUslOF0AjRn1OnEBc&scisf=4&ct=citation&cd=-1.

[21] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media, 2019.

[22] Markus M Breunig et al. "LOF: identifying density-based local outliers". In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 2000, pp. 93–104.

[23] E Fix and JL Hodges Jr. *Discriminatory Analysis. Nonparametric Discrimination: Small Sample Performance.* Tech. rep. 1952. URL: https://apps.dtic.mil/sti/citations/ADA800391.

[24] Zi Lin and Xiaolei Liu. "Wind power forecasting of an offshore wind turbine based on high-frequency SCADA data and deep learning neural network". In: *Energy* 201 (2020), p. 117693. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2020.117693. URL: https://www.sciencedirect.com/science/article/pii/S0360544220308008.