Università della Svizzera italiana

Stochastic Methods 2021

Final Project Report

Gearbox Fault Analysis with Common Machine Learning Methods

Abstract

The gearbox is an essential machinery used in industry to change the speed and load conditions according to the needs. Failure of some of the gearbox components will result in unplanned downtime and increased maintenance costs. To prevent an unintended malfunction, the broken component must be detected early. Vibration signal analysis is a common tool for predicting gearbox faults. Therefore, the use of vibration signals for automatic gearbox fault diagnosis is discussed in this study. This paper analyzes vibration measurement signals for automated fault diagnosis of gearbox. The fast Fourier transform (FFT) is being used to analyze the current signals and the statistical features are extracted from the acquired spectrum data. The extracted features are given as an input to three different machine learning techniques, namely Support Vector Machine, Random Forest and K-means for fault identification. Looking at our analysis of the dataset, we try to find a threshold clearly separating the broken and healthy data in the frequency domain. Lastly, the performance of the gearbox fault classification models are discussed and compared.

Keywords: gearbox, fault diagnosis, statistical features, support vector machine, random forest, k-means

Student 1

Constantin Jehn

Student 2

Glejdis Shkëmbi

Student 3

Franciskus Xaverius Erick

1 Introduction

The gearbox is an essential rotating tool used in industries to adjust speed and load conditions based on the needs. Failure in any of the gearbox components will disrupt production in a factory, resulting in unplanned downtime, production losses and increased maintenance costs. Moreover, gearbox tooth failure is the most common form of gearbox fault [19]. Therefore, in order to avoid unexpected breakdown and improve the mechanical device's reliability, it is important to detect earlier component failure. Fault detection and analysis is considered to be the most important technology in condition-based maintenance (CBM) systems for rotating devices [13]. Vibration signal analysis is a standard method to monitor the machine's condition for predictive maintenance and to efficiently predict the gearbox faults [14].

In this study, the vibration signal, acquired from the online Gearbox Fault Diagnosis dataset, are classified using three different machine learning techniques, namely Support Vector Machine, Random Forest and K-means. In the first step, Fast Fourier Transform is being used to extract the features from our dataset. In the second step, statistical features, extracted from each frequency bin, are being used to have a better accuracy in classification. The extracted features are given as an input to three different machine learning techniques, namely Support Vector Machine, Random Forest and K-means for fault identification. In this study, there are only 2 classes to be classified: healthy and broken conditions of the gearbox. Each gear is tested with 10 different loading conditions. Lastly, the performance of the gearbox fault classification models are discussed and compared with each other and with other research conducted in the field.

Our research question in this study was: "Can these classic machine learning methods help us achieve a good performance in classifying the Gearbox Fault Diagnosis?"

1.1 Goals

The goals of this study are:

- to review current research conducted on gearbox fault diagnosis,
- to study the FFT (Fast Fourier Transform) as the only pre-processing method for gearbox fault analysis,
- to analyze and evaluate effectiveness of Support Vector Machines, Random Forest and K-means in gearbox fault classification.

2 Materials and Methodology

2.1 Dataset Used

In this study, the Gearbox Fault Diagnosis dataset, which is free and publicly available on OpenEI, is used [12]. This dataset contains vibration signals recorded using SpectraQuest's Gearbox Fault Diagnostics Simulator. Such data is collected with the help of four different vibration sensors placed in four directions, recorded under variations of 0 to 90 percent loadings [12]. Moreover, the data set contains two different labeled data: healthy condition and broken tooth condition. Overall, there are 20 loadings, 10 loadings for the healthy gears and 10 loadings for the broken ones. Each loading contains information from four different sensors, and each sensor contains 80.000 to 110.000 records. The readings are taken at a 30 Hz sampling rate. The total sampling time for each combination of loading and state is 1 hour.

2.2 Fast Fourier Transform

The Fourier Transform is a mathematical concept that decomposes a given input signal into its constituent frequencies [6]. The most striking distinction between Fourier Transform and Fast Fourier Transform is that the latter considers as input a discrete signal, while the former takes a continuous signal [6]. The Fast Fourier Transform (FFT) enables signal analysis by allowing us to compute the Discrete Fourier Transform (DFT) of a data vector with fewer operations and fewer round-off errors [7]. Moreover, it is considered to be the most popular and universal algorithm for digital and discrete data analysis and manipulation [16]. The FFT reduces the number of computations required for a problem of size N from $\mathcal{O}(N^2)$ to $\mathcal{O}(n \log N)$[10]. DFT converts a discrete signal into frequency components. In other words, it converts a time-domain discrete signal into a frequency-domain signal [6]. The DFT , defined as

$$X[k] = \sum_{j=0}^{N-1} x[j] \exp\left(\frac{-i2\pi jk}{N}\right), \tag{1}$$

transforms N discrete-time samples to the same number of discrete frequency samples.

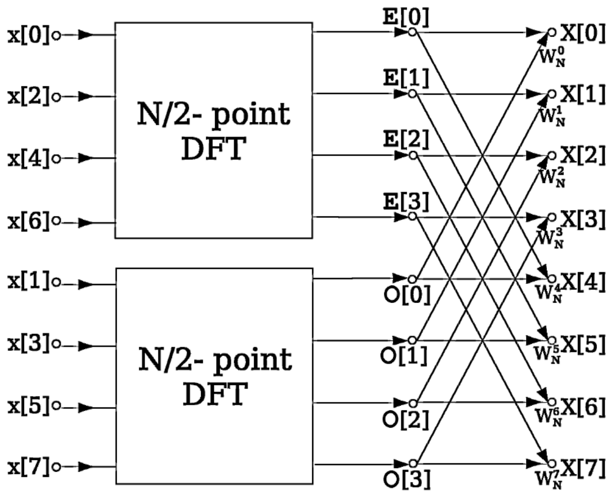


Figure 1. The FFT algorithm breaks the initial problem of length *N* recursively down into smaller sub problems. The reuse of previous factors leads to computational savings[15].

Figure 1 graphically illustrates how all DFT frequency outputs X(k) can be computed as the sum of the outputs of two DFTs of length N/2. The even-indexed time samples are denoted as E[k] and the odd-indexed time samples as O[k] [15]. Due to the periodicity with frequency samples of length N/2, E[k] and O[k] are used to compute two DFT frequencies of length N, that is X[k] and X[k + N/2]. This reuse of these short-length DFT outputs explains the computational efficiency of FFT [15]. Therefore, it has a wide range of applications such as medical imaging, audio signal processing, image processing, error-correcting codes and so on etc., by providing a fast algorithm for convolution, enabling fast large polynomial multiplications, as well as efficient matrix-vector multiplication [16]. In 1994, Gilbert Strang, a great mathematician, known for its contribution to wavelet analysis and linear algebra, described the FFT as "the most important numerical algorithm of our lifetime" [18].

2.3 Data Segmentation

Objective evaluation and high accuracy of a classification model in machine learning research is crucial. Hence, using all the vibration data measurements of the Gearbox Fault Diagnosis dataset in the classification process, seems, at first glance, the right path to follow. However, this is time consuming and computationally inefficient. The goal of segmentation is to cut a signal into meaningful components. In this study, we consider the vibration measurement of only two sensors, sensor 1 and sensor 2, to train our classifiers. We conclude this decision after plotting and carefully observing the results of the FFTs of each sensor data of each of the 10 loadings. Moreover, based on such results, we split the frequency bands in bins. In sensor 1, our bins consist of the frequency bands [0,5],[5,9],[9,15] Hz, while in sensor 2, our bins consist of the frequency bands [0,9],[9,15] Hz. More detailed information of the procedure followed and the FFT graphs can be found under the section Overall Procedure.

2.4 Feature Extraction

In order to have a highly diagnostic accuracy in gearbox analysis, dimensionality reduction of the feature extraction is needed. In the study of Praveenkumar, T., et al. [14], time domain statistical features such as mean, median, mode, standard deviation, variance, skewness, kurtosis, root mean square are used. Zhang Xin, et al. [20] uses time domain parameters including energy, root mean square and peak value. Moreover, Sahoo, Sudarsan, et al [17] performs statistical analysis, such as root mean square, mean, peak value, skewness, standard deviation, variance, kurtosis for the healthy, Type-I and Type-II defect gear. In the article from Bajric, Rusmir, et al. [3] Discrete Wavelet Transforms are being used in revealing fault related information from nonstationary signals acquired on rotating machineries. The output of FFT provides us with a good picture of the distribution of the vibration signal in the frequency domain. However, to reduce the dimensionality of our feature space, in this study, statistical indices are being used as follows:

1. The mean of the values in each frequency bin.
2. The standard deviation in each frequency bin.
3. The kurtosis in each frequency bin.
4. The root means square (RMS) in each frequency bin.

2.5 Machine Learning Classifiers

2.5.1 Random Forest Classifier

A Random Forest (RF), first introduced by Leo Breiman [4], is an ensemble method based on decision trees and bagging. Combining the principles of random feature selection and bagging, it, first, generated an ensemble of trees, which then vote for the most popular class [9] (see Figure 2). Bagging is a procedure that does random sampling with replacement, resulting in improved stability and accuracy of classification and regression models [13]. In other words, bagging creates many different “copies” of the training data, applies the weak classifier to each of these copies, and combines their result. By using multiple samples of the original dataset, it results in reduced variance of the final model, which results in low overfitting [5]. In order to build these ensembles, for the k-th decision tree, this approach generates a random vector k with identical distribution for all the trees in the forest, but independent of all the previous random vectors i where i = 1,...,k-1, and grows a tree using the training set and the generated random vector [4]. As a result, we have a classifier h(x,k) where x denotes the input vector and k denotes the generated random vector [4]. So as to achieve low bias, the trees are allowed to grow to the largest extent possible without pruning [2]. After a large number of trees is acquired, they select the winner class based on the number of votes it obtained [4]. RF classifiers are well known for the following advantages making them one of the most widely used ensemble methods [9]:

- Can handle both categorical and continuous features;
- Can work with missing and noisy data;
- Can be used on data with a very large number of features because it selects only a subset of them;
- There is little risk of overfitting.

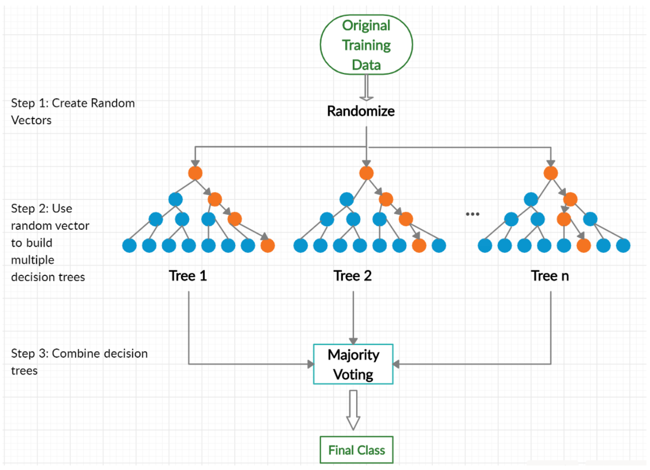

Original Training Data
Step 1: Create Random Vectors
Randomize
Step 2: Use random vector to build multiple decision trees
...
Tree 1
Tree 2
Tree n
Step 3: Combine decision trees
Majority Voting
Final Class

Figure 2. Random Forests are ensembles of randomized decision trees. The final classification is done by majority voting.

2.5.2 Support Vector Machine Classifier

Support Vector Machines (SVM) were first introduced by Vapnik in 1995 [11]. These supervised learning algorithms are widely used for classification and regression [21]. SVM is a binary linear classifier which maximizes the margin between examples from different classes in the data [21]. In other words, one can imagine an SVM model for two-dimensional feature vectors as an algorithm that tries to find an optimal separating hyperplane $\omega^T x + b = 0$ that linearly separates two classes while maximizing the distance to the closest point from either class [11]. An example will be classified in the positive class if $\omega^T x + b \geq 0$, and will be classified in the negative class if $\omega^T x + b < 0$ [8]:

$$f(x) = \begin{cases} 1 & \text{if} \quad \omega^{\mathrm{T}} x + b \geq 0 \\ -1 & \text{if} \quad \omega^{\mathrm{T}} x + b < 0 \end{cases} \tag{2}$$

Please refer to Figure 3 for a better visualization. Solid and empty circles belong to data of class 1 and -1 respectively, while the two data samples on the margin are the support vectors.

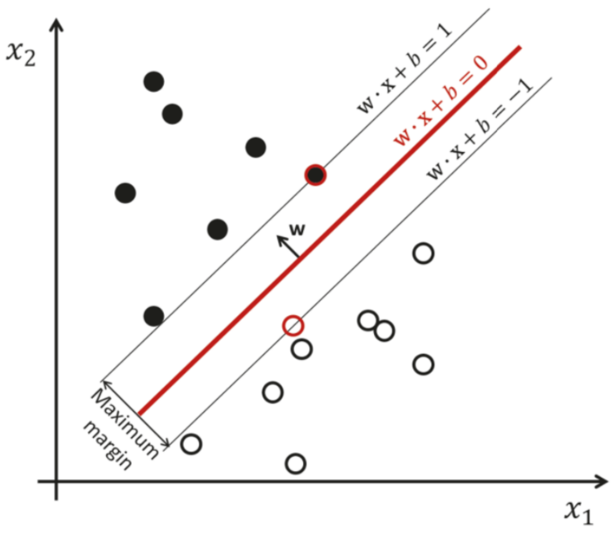


Figure 3. SVM models maximize the margin between the clusters to be classified. The decision boundary is solely determined by the Support Vectors which are encircled in red.[21].

In real life, data is not always linearly separable. In addition to linear classification tasks, SVM can classify non-linearly separable classes by applying different kernel transformations for nonlinear mapping to a higher dimensional feature space (labeled by K(.) in 4) [1]. The kernel exploits the fact that we can use the dot product to write the following [8]:

$$\omega^T x + b = b + \sum_{i=1}^{m} y_i \alpha_i x^T x^{(i)}. \tag{3}$$

Now, by replacing the dot product with a kernel function, we can make predictions with the following function [8]:

$$f(x) = b + \sum_{i} y_i \alpha_i k\left(x, x^{(i)}\right). \tag{4}$$

In other words, SVM task does not change except from instead of a dot product, we now have a nonlinear kernel function, such that the input data is mapped to a higher dimensional space in which the separating hyperplane maximizes the margin.

Some popular kernel functions used for SVM are polynomial functions, Gaussian radial basis functions or hyperbolic tangent functions [21]. SVM classifiers are well known for the following advantages [21]:

- Can efficiently deal with large and sparse datasets, because they rely on support vectors to find hyperplanes;
- Can deal with large feature space, because the complexity of the algorithm does not depend on the
- dimensionality of the feature space;
- Guaranteed to converge to a single global solution;
- Using soft margin, overfitting can be controlled.

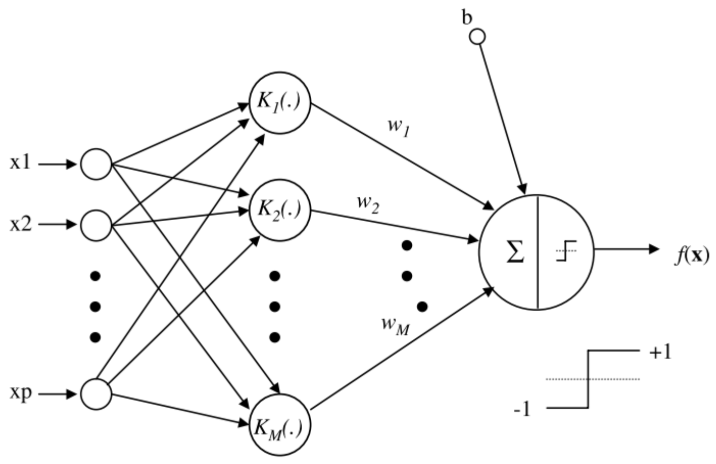


Figure 4. Visual representation of the architecture of SVM [1].

2.5.3 K-means clustering

Clustering is an unsupervised learning algorithm that tries to divide the data into clusters, such that data points of the same cluster are similar to each other and are different to data points of the other clusters [21]. It is important to note that the algorithm does not know beforehand what comprises a cluster, therefore, it is mostly used for knowledge discovery rather than prediction [9]. The K-means algorithm is a vector quantization method and maybe the most used clustering techniques [21]. It assigns each of the n examples into one of the k mutually excessive groups, called clusters, aiming to minimize the difference of data points within each cluster and maximize the differences between clusters [9]. First, a number k for the number of clusters has to be chosen (see 5). Then, k feature vectors, called centroids (represented by squares in Figure 5), are put randomly to the feature space. The distance between each example x and each centroid c is computed using some metric (eg. Euclidean distance), and, then, the example x is assigned to the closest centroid according to this distance metric [5]. For each centroid, the average feature vector of examples labeled with the centroid id is calculated, and such vectors then become the new location of the centroids [5].

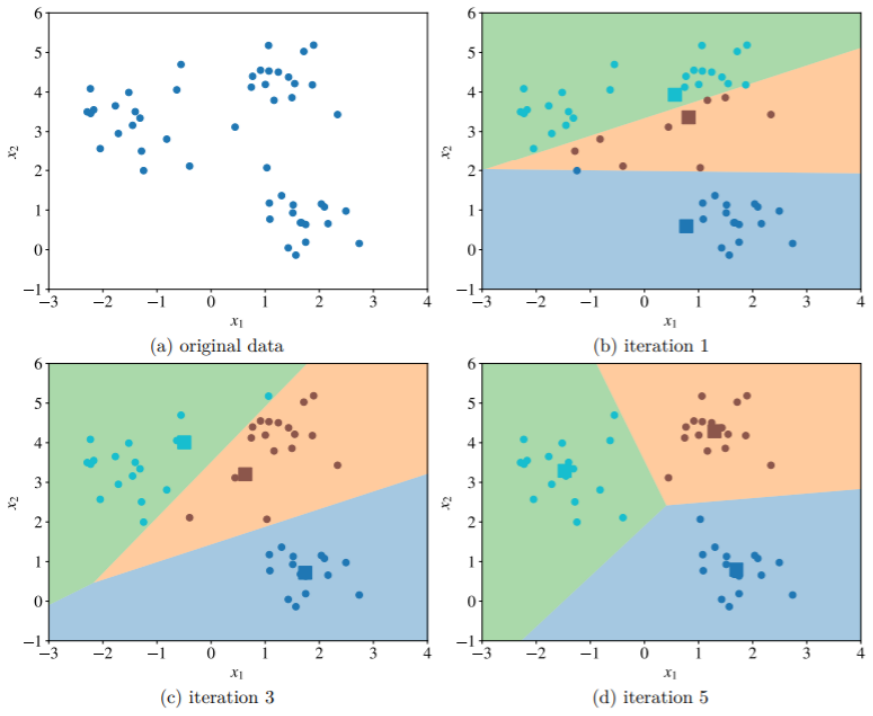


Figure 5. The K-means algorithm iteratively searches for k centroids of clusters (here $k = 3$). The feature vectors are assigned to one on the centroids according to shortest distance according to a predefined metric. [5].

The following shows some of the main advantages of K-means [9]:

- Is simple and can be understood without using statistical terms;
- Is highly flexible;
- Is fairly efficient in dividing the data into meaningful clusters.

However, it is also important to mention a few disadvantages [21]:

- Is used only for data objects in a continuous space;
- It requires the number of clusters k to be specified in advance;
- Is not suitable to discover non-convex shape clusters;
- Is sensitive to outliers.

2.5.4 Model Performance Assessment

There exist different evaluation metrics to assess the performance of a classifier. In this paper, we are considering four different statistical approaches: confusion matrix, ROC curve, F-measure and overall accuracy. The confusion matrix shows how good the classifier is at predicting examples belonging to different classes, by categorizing predictions on whether the predicted label is the same as the actual label [5]. It is a table containing the predicted examples in one axis and the true values in the other axis. The ROC (receiver operating characteristic) curve is a graph obtained by plotting true positive rate (sensitivity) in the x-axis and false positive rate (1 - specificity) in the y-axis indicating the performance of a given classifier at all classification thresholds. Sensitivity indicates how often a test correctly gives a positive result for gears that are broken, while sensitivity indicates how often a test correctly gives a negative result for gears which are healthy. True Positive Rate (TPR), also known as recall, is defined as follows:

$$\text{Sensitivity } = TPR = \frac{TP}{TP+FN}, \tag{5}$$

while False Positive Rate (FPR) is defined as:

$$1- \text{ Specificity } = FPR = 1 - \frac{TN}{FP+TN} = \frac{FP}{FP+TN}. \tag{6}$$

The closer the ROC curve is to the left upper corner of the graph, the better the model is at identifying positive values [9]. Moreover, one can calculate the AUC (area under the curve) value which measures the entire two-dimensional area under the ROC curve, which can range from 0.5 for a random classifier to 1.0 for a perfect classifier [9].

Precision shows the ratio of positive predictions that were actual positives, while recall shows the ratio of actual positives that were predicted correctly.

$$\text{Precision } = \frac{TP}{TP+FP} \quad \text{and} \quad \text{Recall } = \frac{TP}{TP+FN}. \tag{7}$$

Finally, the overall accuracy is the most used method in model assessment, defined as:

$$\text{Accuracy } = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

3 Experiments

3.1 Overall Procedure

In this study, the Gearbox Fault Diagnosis dataset is being used. This dataset contains vibration signals, collected with the help of four different vibration sensors placed in four directions, recorded under variations of 0 to 90 percent loadings [12]. Overall, there are 20 loadings, 10 loadings for the healthy gears and 10 loadings for the broken ones. Each loading contains information from four different sensors, and each sensor contains 80.000 to 110.000 records [12]. Figure 6 gives a visual representation of the proposed procedure of the analysis on vibration data of wind turbines. The experiments were implemented in Matlab.

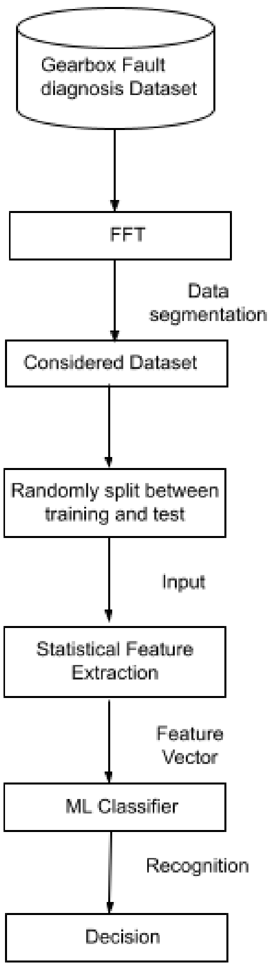

Gearbox Fault diagnosis Dataset
FFT
Data segmentation
Considered Dataset
Randomly split between training and test
Input
Statistical Feature Extraction
Feature Vector
ML Classifier
Recognition
Decision

Figure 6. Procedure of analysis on vibration data of wind turbines.

First, we apply FFT on all four sensors and each of the 10 loadings, both for the healthy data and for the broken data. In Figure 7, we may see the FFT for the healthy data is plotted in green and for the broken data is plotted in red. From carefully observing all the FFT plots of all the sensors for each loading, we conclude that using only the vibration measurements of the two sensors, sensor 1 and sensor 2, provides enough distinctive information to help us better train our classifiers.

Next, we compute and plot the spectrogram of the vibration measurements, returning the discrete short-time Fourier Transform of the input signal for the healthy and broken data. We specify the sampling rate at 30 Hz, as specified in the dataset. We set the segment length of 128 samples, meaning that we divide the signal into sections of length 128. Lastly, we specify 64 samples of overlap between adjoining sections and evaluate the spectrum at floor(128/2+1) = 65 frequencies. The 128 DFT points and the default Hamming window are being used. Such analysis was implemented using Matlab built-in functions fft and spectrogram. The results of the loading 5 in sensor 2 are shown in Figure 8. The plot tells us that our data is most likely stationary, which means that the statistical properties of our time series or rather the process generating it do not change over time. Therefore, we decide we stay only with FFT and do not apply any other wavelet analysis. Moreover, from the spectrogram we see that the healthy and broken data have very distinctive frequency distribution in the time domain. Considering this information, during the experimental procedure we will aim to find a threshold of the data by plotting them in different time bins.

The frequency bands are split into 15 bins, because we have a resolution of maximum 15 Hz, each bin corresponding to 1 Hz. Next, FFT for the vibration data of only the first two sensors is being computed, both on the healthy dataset and the broken dataset. In order to extract the features for the classification, sensor 1 is split into frequency bands [0,5], [5,9], [9,15] Hz and sensor 2 is split into frequency bands [0,9], [9,15] Hz, such that we have a better distinction between healthy data and broken data, as shown in Figure 7. The output of FFT provides us with a good picture of the distribution of the vibration signal in the frequency domain. However, to reduce the dimensionality of

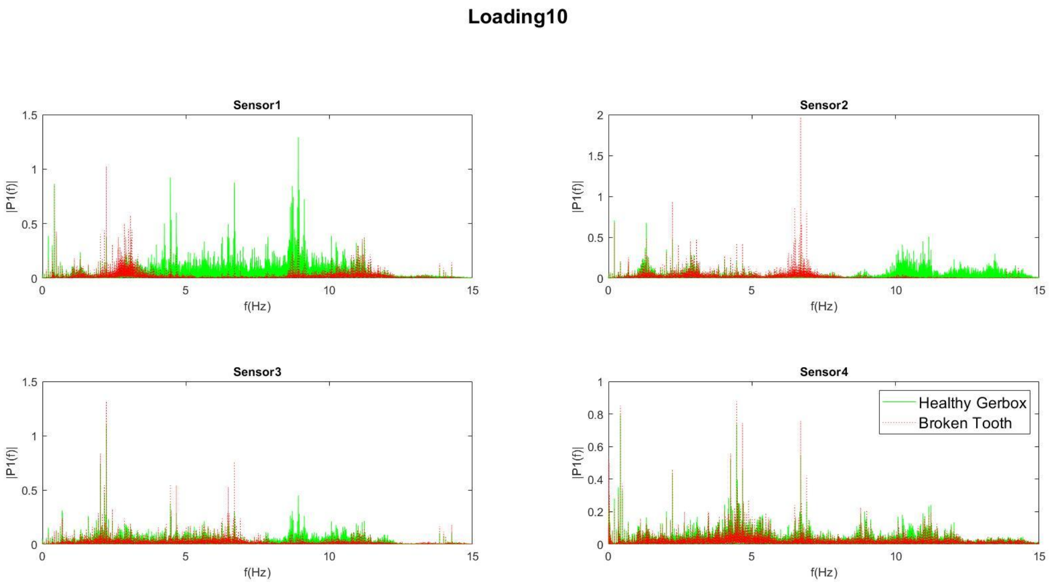


Figure 7. The exemplary FFT of the 10 % loading shows a significant difference in the frequency domain between a healthy and a broken gearbox. Especially sensors 1 and 2 show very discriminative behaviour in the frequency domain

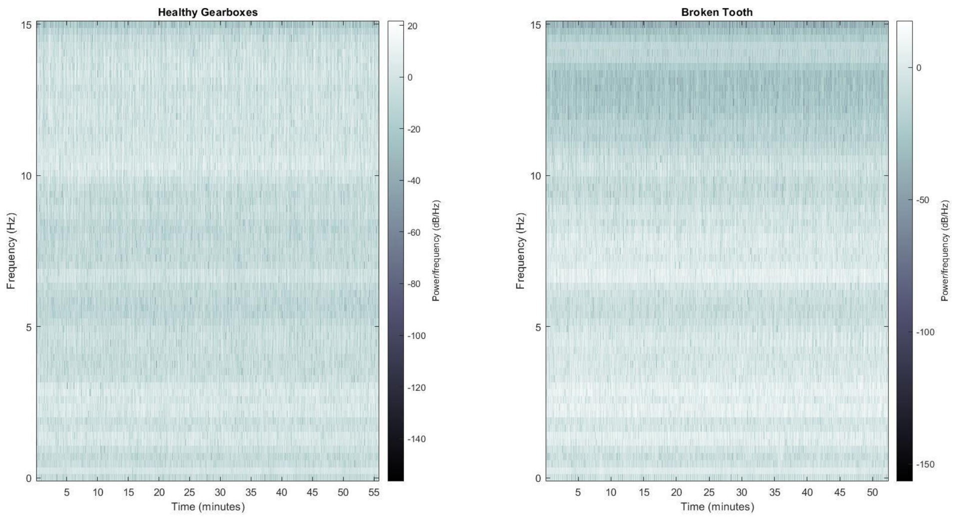


Figure 8. The difference in particular frequency domains between the two classes are also apparent in the spectrogram. The spectrogram of sensor 2 for 50% loading shows the amplitudes between 10 and 15 Hz are significantly increased for the broken gearbox.

our feature space, in this study, statistical indices, such as the mean, the standard deviation, the kurtosis and the RMS in each of these frequency bins, are being used. As a result, we have five values for each of the statistical indices, and this is applied in each of the 10 loadings. In order to use the most important features with the highest weight in prediction we perform a visual representation of the variables from the feature set. As it can be seen from figures 9 and 10, we are optimistic to classify the time series successfully using the “mean sensor 1 bin 2” and the “mean sensor 2 bin 2” as the strongest features for training our machine learning models. Moreover, we compare this result to the FFT to see if such features make sense, and we observe that the tendency of the mean matches the absolute value in the frequency domain.

The vibration measurements of the healthy and broken data are randomly split into bins, each bin containing random part of the time measurements. We provide two different splitting methods by setting the argument mix to either true or false. If mix is set to true, data is split into bins by randomly taking selected slices from each loading from both healthy and broken dataset, while if it is set to false, then we always take the same slice. The seed is set to provide reproducibility of our experiments. Furthermore, those randomly selected vibration measurement data are

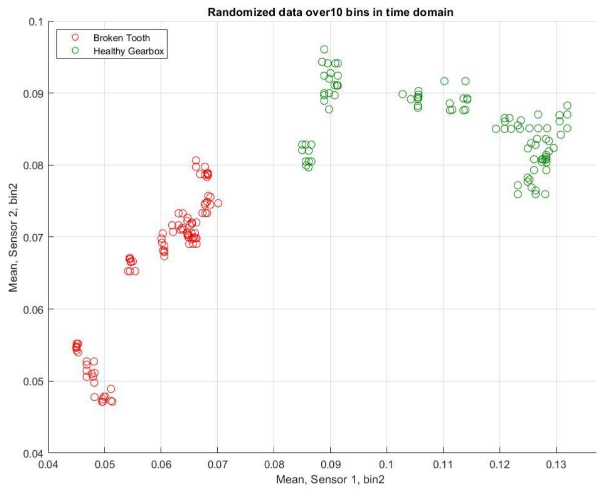


Figure 9. Six minutes of recordings are drawn randomly ten times. The resulting time frames are represented in a highly discriminative two-dimensional feature space.

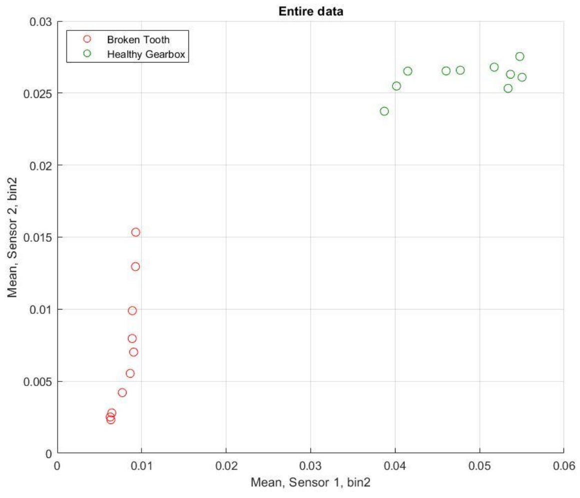


Figure 10. The entire data recording of 60 minutes in each loading and state (healthy or broken) are represented in a two-dimensional feature space. The data is linearly separable.

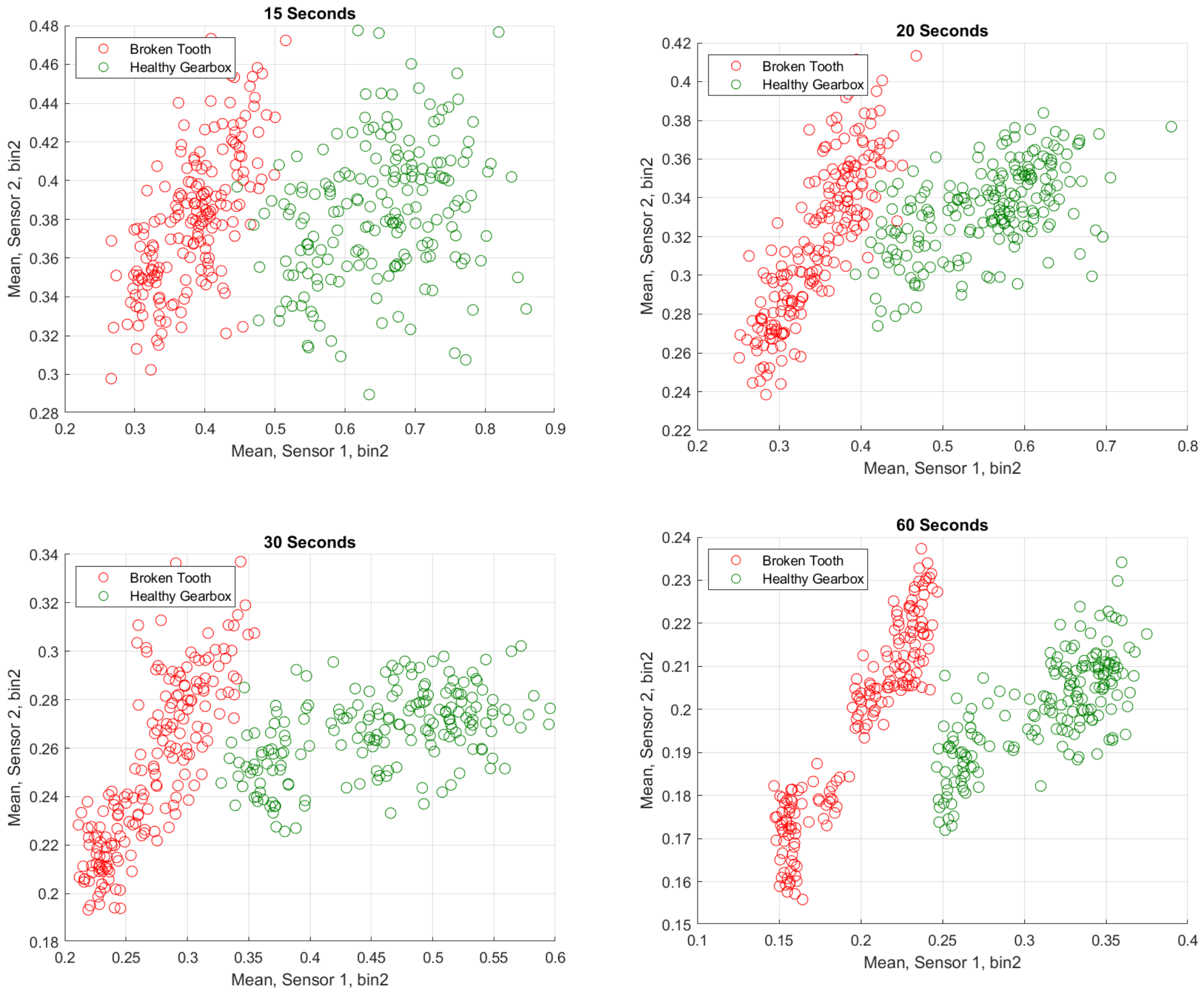


Figure 11. Twenty time frames are drawn randomly from the overall recording (including all loading) and represented in a two dimensional feature space. We repeat the procedure for time frames of length 15, 20, 30 and 60 seconds. The feature "Mean Sensor 1, bin 2" is highly discriminative. In particular for 60 seconds it provides a good threshold.

randomly split into 70% for the training samples and 30% for the testing samples. The classification is performed using three machine learning techniques: SVM, random forest and k-means. Moreover, in this study, 30% holdout cross validation is used to estimate model accuracy. The Random Forest is implemented using the MATLAB built-in function TreeBagger, which grows an ensemble of decision trees using bootstrap samples of data to avoid overfitting, by selecting only a subset of prediction features at each decision split. We set the number of trees to 500 and set the OOBPrediction in order to store info on what observations are out of the bag for each tree. Finally, k-means clustering, to partition the observations into k clusters, is implemented using the kmeans function of MatLab. We set $k = 2$ and we set replicates to 5, which indicates the number of times to repeat the clustering using initial cluster centroid positions.

Lastly, after observing the spectrogram, we performed a small analysis on different time frames, in order to find the correct one to use for the thresholding of our dataset. We plot our dataset for different time bins, namely 15, 20, 30 and 60 seconds. Note also that the sampling rate of the sensors is 30 Hz. The plots are shown in the Figure 11 respectively. In the scatter plots, all loadings are included and the time bins are drawn 20 times with different random seeds. As one can expect, the separation gets clearer with longer time we consider. As our aim is to find a threshold in only one feature dimension, we decide to take the 60 seconds to define the clearest boundary in the x-axis. Taking into account the practical implication of the problem of the wind turbines, this is also a reasonable time to intervene, as the process is not too agile and it will take at least several minutes to slow down the rather inert system. Furthermore, for the maintenance action to be taken, it will take a few more days until the employees arrive in the wind turbine plant. This is the reason we decide to use a longer time frame and expect better classification performance.

4 Experimental Results

In this section, we will be reporting and analysing the results obtained from all our experiments. Each prediction model was tested and evaluated using the evaluation matrices described in section 2.5.4. Figure 12 plots the support vector machines of the whole dataset trained using two features, the decision boundary, and the support vectors displayed in black circles. The green dots correspond to the healthy data of the training set, while the red dots correspond to the broken data. The green circles correspond to the data points from the test set classified as healthy from SVM, while the red circles correspond to the data points from the test set classified as broken from the SVM. The pluses represent the golden standard. From this representation we can clearly see that we have 100% accuracy.

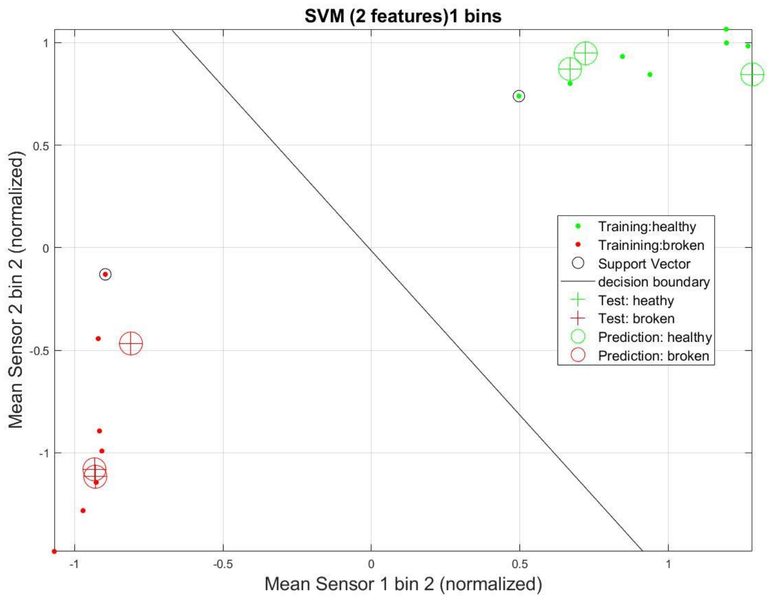


Figure 12. The SVM model for 2 features is represented graphically. It provides a reliable decision boundary applied on the entire data of 60 minutes.

The performance of the SVM model in terms of precision, recall and specificity for different numbers of bins in the time domain is given in figure 13 for mixed splitting and figure 14 for not mixed splitting in the training dataset. In the case of mixed splitting, after training the SVM in different bin numbers we notice that all the SVM classifiers perform very well, giving 100% precision and specificity on the test set for all bin numbers except for the SVM classifier trained on 30 bins, where we have a 75% precision and 67% specificity. Moreover, the SVM has a perfect recall on the test set for all bin splits except when we split data into 2, 7 and 40 bins. In the case of not mixed splitting in the testing dataset, we observe perfect precision and specificity on the test set, while recall drops when we split data into 2, 30 and 40 bins.

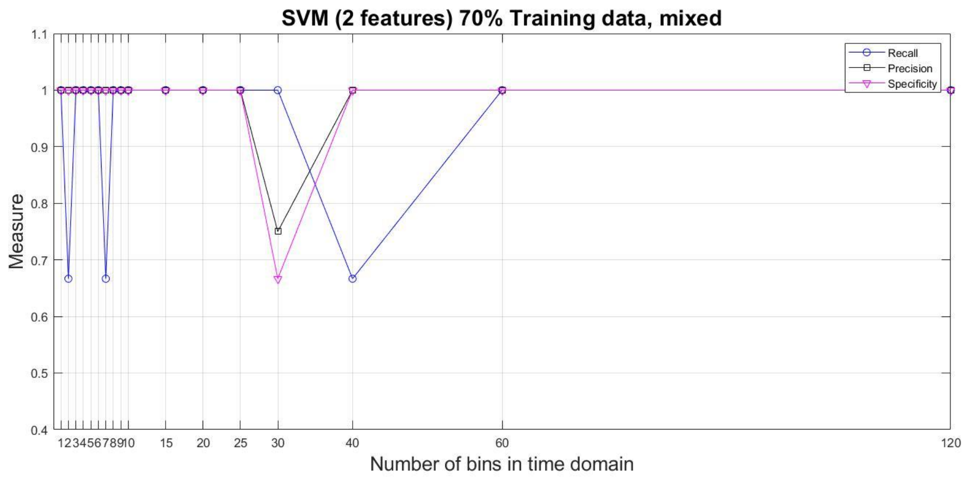


Figure 13. Plot of precision, recall and specificity of the test dataset prediction made by our SVM model trained on mixed dataset. Note that some bins result to dips in the metrics. However, in overall the performance is still very satisfactory.

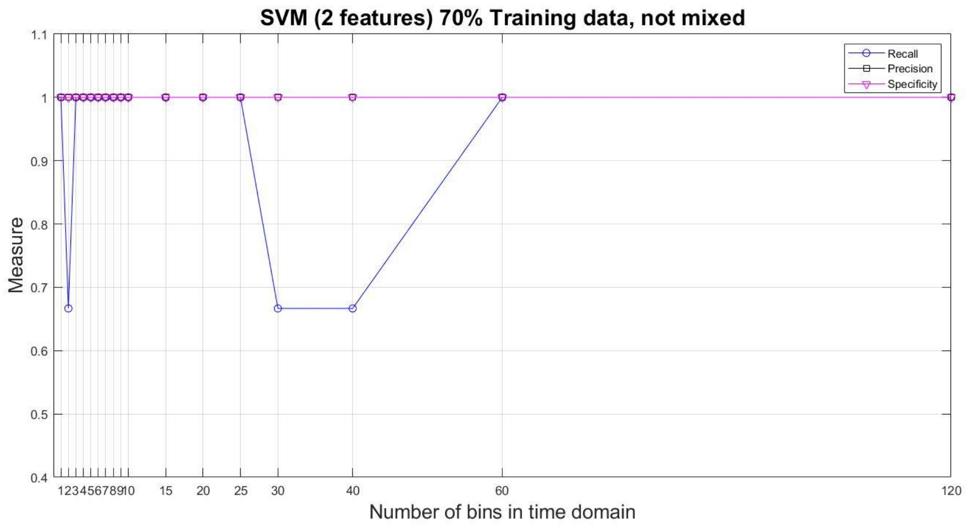


Figure 14. Plot of precision, recall and specificity of the test dataset prediction made by our SVM model trained on non-mixed dataset. Note that some bins result to dips in the metrics. However, in overall the performance is still similary satisfactory as when using mixed data.

Figure 15 and 16 provide the confusion matrix of the two classes for the SVM classifier, when we split data into 40 bins for the mixed version of SVM training and into 2 bins for the not mixed version. We chose to display only these two that show the worst performance in terms of recall, precision and accuracy, as shown in the figure 13 and 14. The rows represent the golden standard and the columns represent the classifier outputs. The confusion matrix shows that the 3 healthy data points are classified correctly from the SVM model. For the broken data group, one of the data points is misclassified as healthy. This result we observe when we split the data into 40 bins to train the mixed version of SVM model, and when we split the data into 2 bins for the not mixed version.

Figure 17 shows the 10 fold, 30% holdout cross validation accuracy and the accuracy on the test set for the mixed SVM for different number of bin splits of the data in time domain. We observe 100% accuracy in most bin splits. However, there is a decrease of accuracy when we split the data in 2, 7, 8, 30 and 40 bins. The accuracy on the test set drops to 84%, while the cross validation accuracy drops even further to 50%, which characterizes a random classifier. In figure 18, we see a slightly better cross validation accuracy, which drops to 50% only when we split the

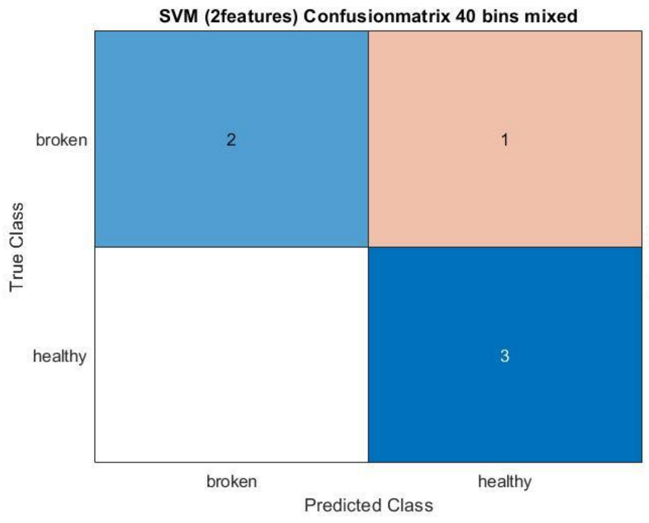


Figure 15. Confusion Matrix of the SVM model with mixed training data. There is one false negative prediction out of all 6 predictions.

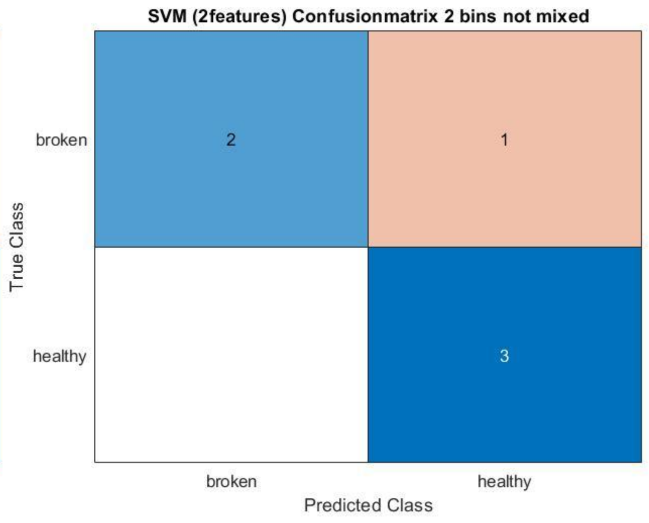


Figure 16. Confusion Matrix of the SVM model with non-mixed training data.The false negative prediction is similar to the case of mixed data.

data into 7 bins.

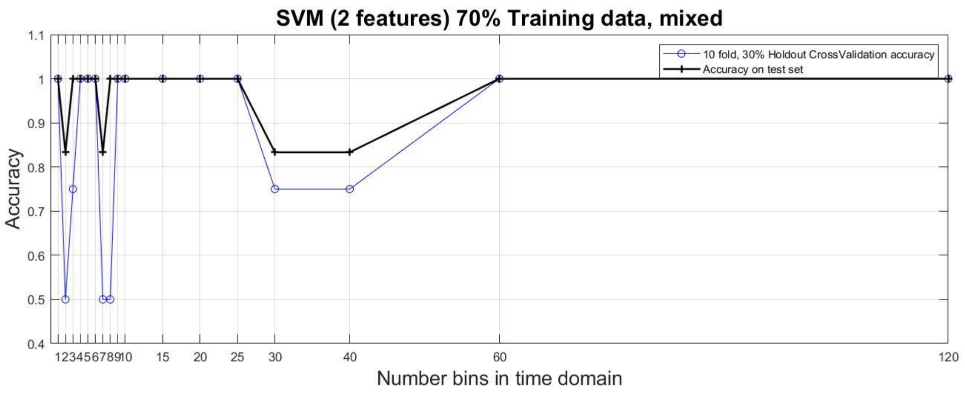


Figure 17. Cross Validation accuracy and accuracy on the test set for our SVM model with mixed dataset. While for most cases the predictions are accurate, for some bin values the accuracy dip to 50%.

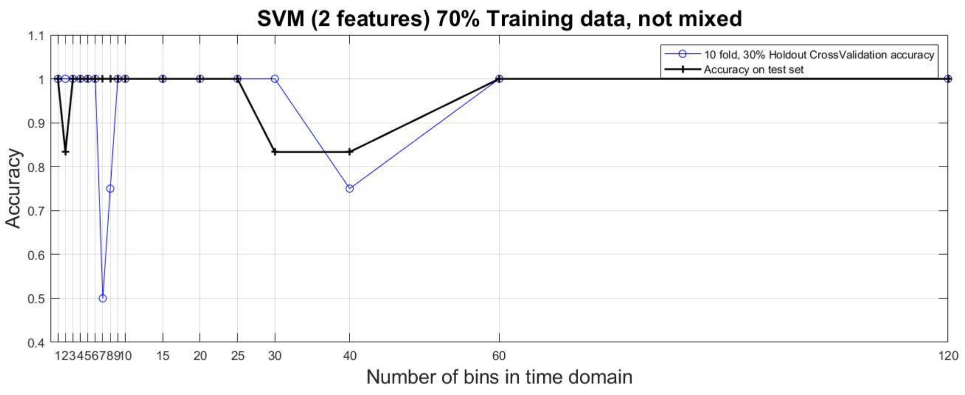


Figure 18. Cross Validation accuracy and accuracy on the test set for our SVM model with non-mixed dataset. The plot is again similar to the mixed dataset case, however the dip of accuracy is not as drastic.

The area under the ROC curve has been widely used in computer-based decision making to help define the accuracy of the model. One of the advantages of using AUC is its visual accessibility from a ROC plot, as shown in Figure 19. Each plot represents one ML method used in gearbox fault classification, and each color represents a different number of bin splitting of the data. As mentioned previously, the closer the curve is to the left upper corner of the graph, the better the classifier performs for this specific bin split. As it can be seen in the plot of the Figure

19, SVM classifier gives perfect result for most of the splits. The pink triangle curve representing 7 bins split shows a slightly worse result compared to the rest. This matches the result we showed above in terms of accuracy, precision and recall. Figure 20 shows the area under the curve for the SVM classifier for different bin sizes. We observe 1 AUC for almost all the bin splits, except for 2 and 7 bins. This result agrees with the performance we showed so far with the other evaluation metrics. Moreover, the cross validation accuracy and the off training accuracy of the SVM classifier for the whole data set is 100%.

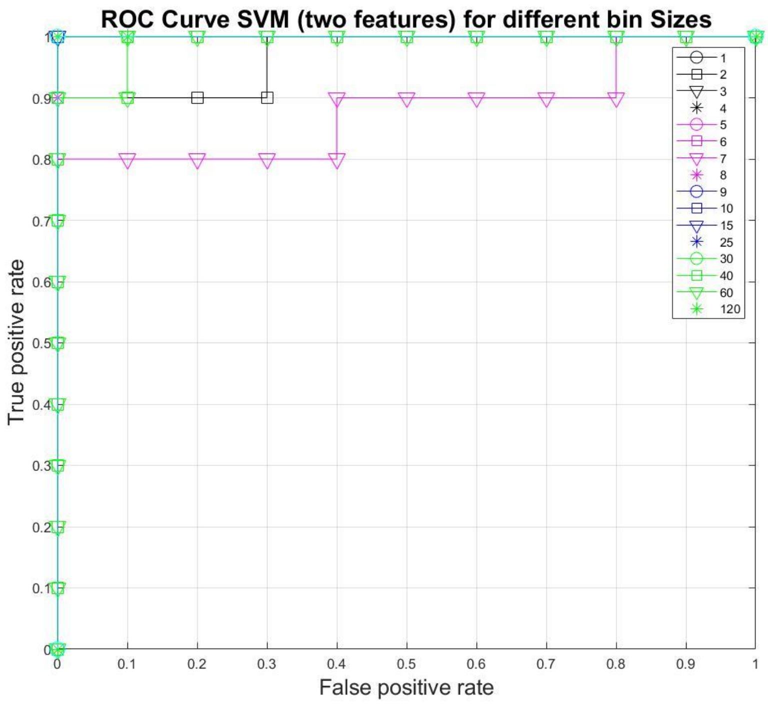


Figure 19. Receiver operating characteristics (ROC) for each bin split for SVM model. This shows that the SVM model prediction is very accurate for most bin values as the curves tend towards the upper left corner of the diagram.

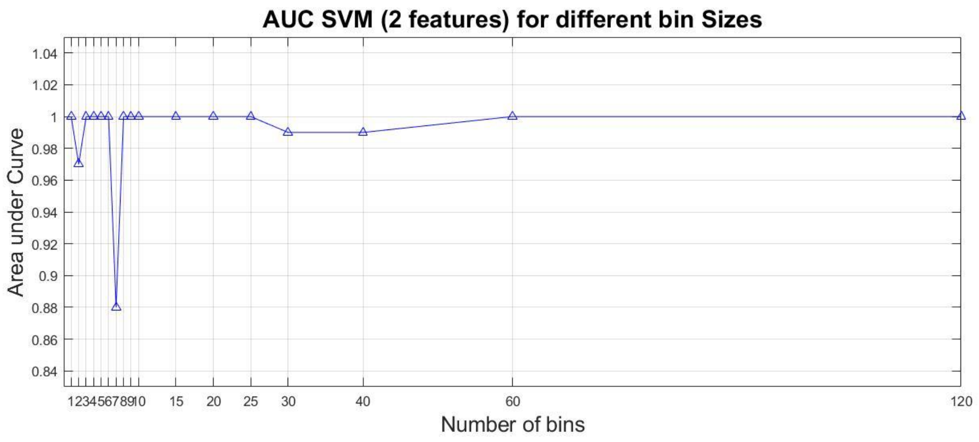


Figure 20. Area Under the Curve for different bin sizes SVM classifier. The plot confirms that in overall, the SVM model provides good predictions for our dataset with the exception of a few bin values. This provides an interesting insight into the behaviour of our time-series data and how splitting the data affects the data and performance.

Figure 21 shows the output of the random forest classifier and what the model believes to be a good boundary to classify the points into. If the x-axis is more than 0.0258, then the test points will be classified as broken. This result can also be better visualised in the scatter plot of the 22, where the green and the blue empty circles are the 6 test points. The blue empty circles are the test points classified as healthy data, while the green empty circles are the test points classified as broken from the random forest classifier. The red dots represent the broken data from the training set, while the green dots are the healthy data points from the training set. Therefore, we can clearly observe that the RF model performs perfect classification.

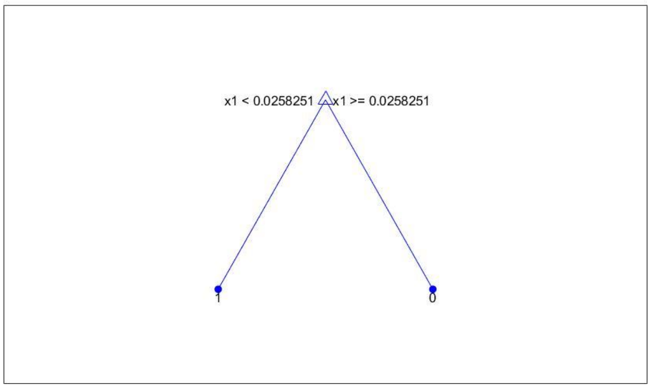


Figure 21. The overall random forest classifier as given by MATLAB. The averaged decision tree would be a single node decision tree, whereby the splitting decision is taken to be the value of x_1, which is the mean of sensor 1. The input would then be classified as either 'broken' (1) or 'healthy' (0).

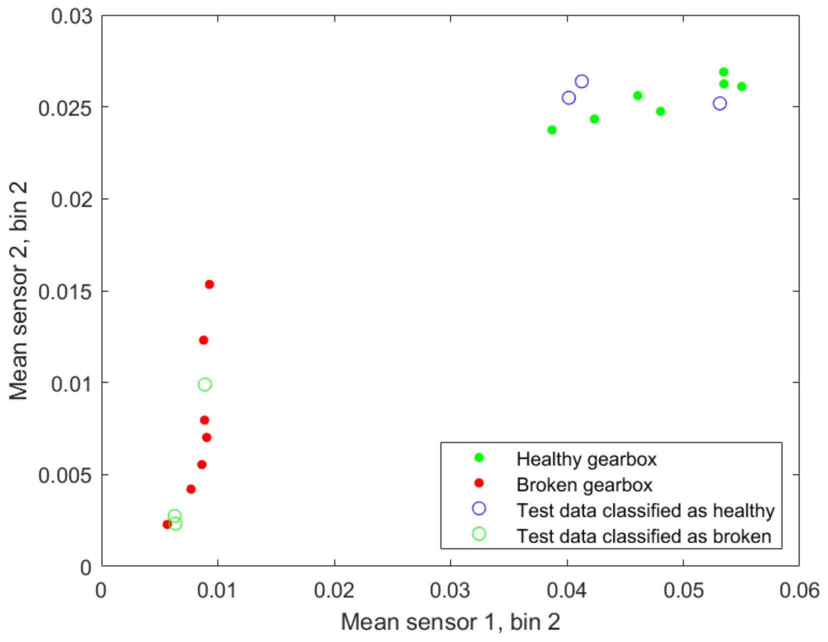


Figure 22. Scatter plot of Random Forest classifier. This visualization further confirms the splitting decision taken by the overall random forest classifier for our dataset. By visual inspection one can observed that the dataset can be classified from the mean of sensor 1.

The performance of the Random Forest model in terms of precision, recall and specificity for different numbers of bins in the time domain is given in Figure 23 for mixed splitting and Figure 24 for not mixed splitting in the training dataset. In the case of mixed splitting, after training the RF in different bin numbers we notice that all the RF classifiers perform very well, giving 100% precision and specificity on the test set for all bin numbers except for the RF classifier trained on7 and 30 bins, where we have a lower precision and specificity. Moreover, the RF has a perfect recall on the test set for all bin splits except when we split data into 2 and 40 bins. In the case of not mixed splitting in the testing dataset, we observe the same pattern, with some lower performance in bin splits of 2, 7, 30 40. If we compare this to the SVM model, we also saw a decrease in performance in the same bin splits, namely 2, 7, 30 and 40.

Figure 25 and 26 provide the confusion matrix of the two classes for the Random Forest classifier, when we split data into 2 bins for the mixed version of SVM training and into 2 bins for the not mixed version. We chose to display only these two that show the worst performance in terms of recall, precision and accuracy, as shown in the figure 23, 24, 6.5 and 6.6. The rows represent the golden standard and the columns represent the classifier outputs. The confusion matrix shows that the 3 healthy data points are classified correctly from the RF model. For the broken data group, one of the data points is misclassified as healthy. The same result was observed for the mixed and not mixed model.

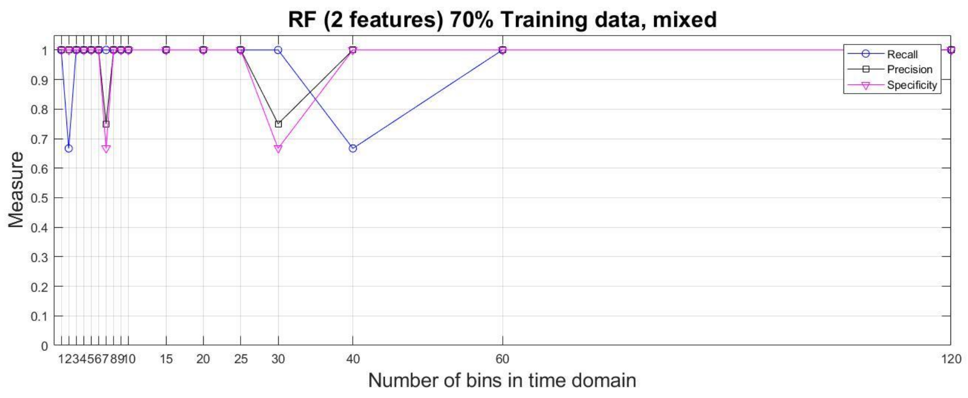


Figure 23. Plot of precision, recall and specificity of the test dataset prediction of our Random Forest trained with mixed dataset. This also shows that the random forest model performs relatively well when trained with binned and mixed dataset, with the exceptions for a few bin values where the metric values dip.

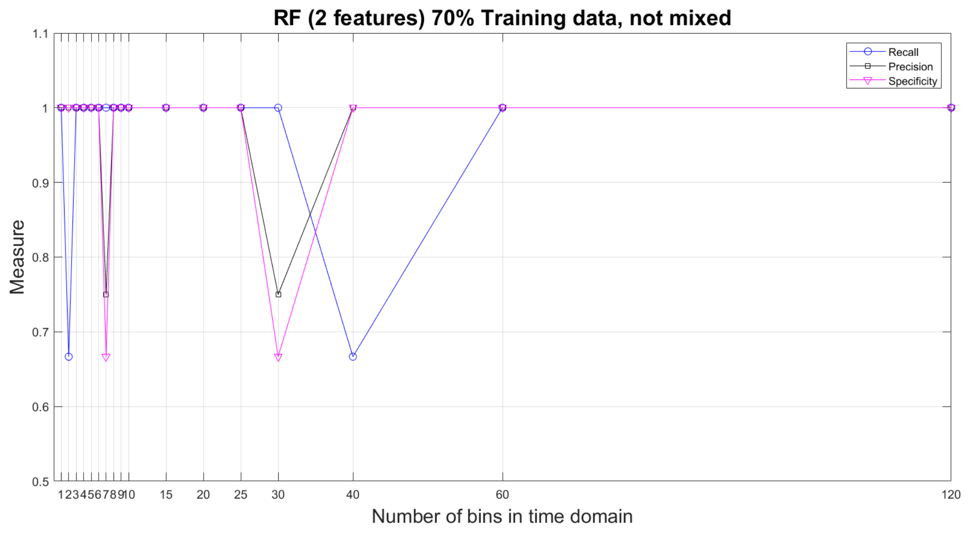
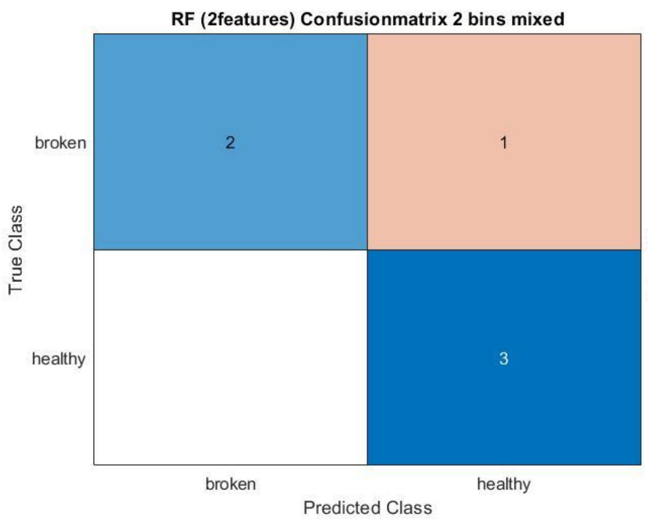


Figure 24. Plot of precision, recall and specificity of the test dataset prediction of our Random Forest trained with non-mixed dataset. The metrics look similar to the mixed dataset case.



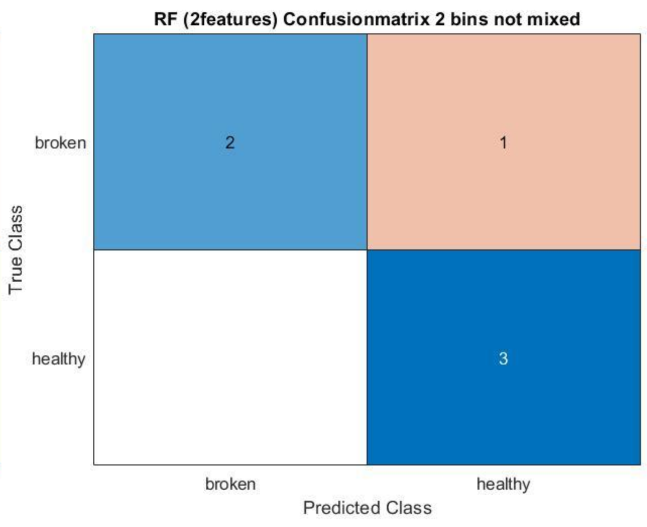


Figure 25. Confusion Matrix of the random forest model with mixed training data split into 40 bins. There is one false negative prediction out of all 6 predictions, which is comparable to the SVM model performance.

Figure 26. Confusion Matrix of the random forest model with non-mixed training data split into 40 bins. There is one false negative prediction out of all 6 predictions, similar to the case of training with mixed training data.

The Figures 27 and 28 display the cross validation accuracy of the Random Forest model for the mixed and non mixed model. This graph projects the results we already observed above in terms of precision, recall and specificity regarding the bin splits 2, 7, 30, 40.

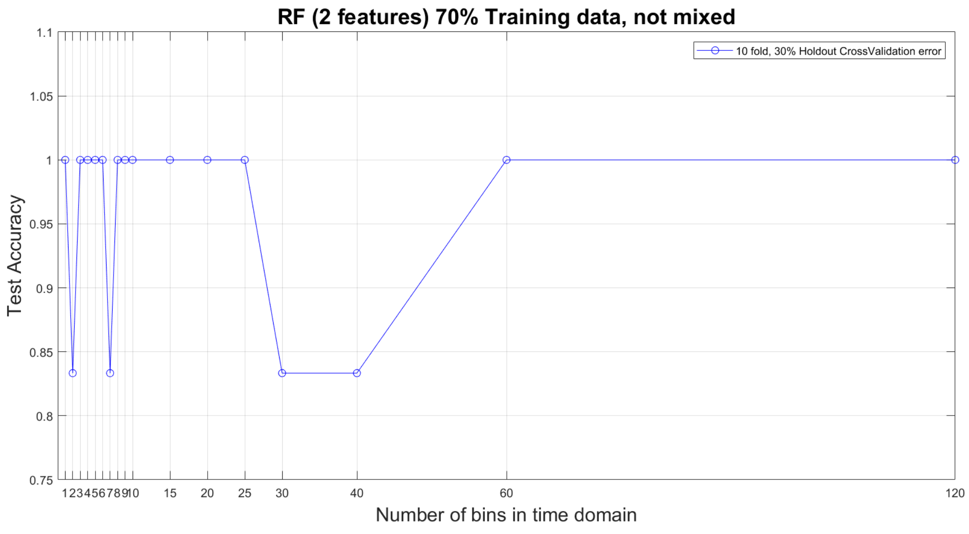


Figure 27. Cross Validation accuracy of the random forest model training on non-mixed dataset. The accuracy values dip slightly for some bin values but overall the model shows very high accuracy.

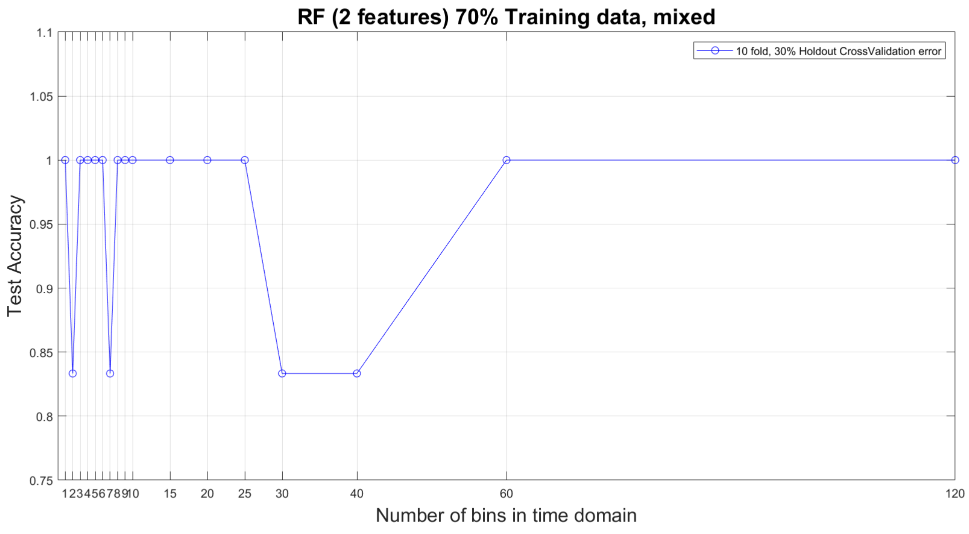


Figure 28. Cross Validation accuracy of the random forest model training on mixed dataset. The performance again is very similar to the case of the model trained on non-mixed dataset.

K-means clustering is visualized in figure 29. The two class centroids are represented in black cross. After repeating the clustering 5 times using initial cluster centroid positions, we have a best total sum of distances of 0.000374446. The broken data samples are displayed in red dots, while the healthy ones are displayed in green dots. The empty blue circles represent the test data put into the broken cluster, while the empty green circles represent the test data put into the healthy cluster. From this visual representation, we conclude that the test data samples are correctly put into the healthy and broken clusters.

The performance of the k-means model in terms of precision, recall and specificity for different numbers of bins in the time domain is given in figure 30 for mixed splitting and figure 31 for not mixed splitting in the training dataset. In the case of mixed splitting, after training the k-means in different bin numbers we notice that the performance is very poor, especially in the case of 10 and 25 bin splits, where the specificity drops to 0. This means that there are no true negatives, and all gears without the broken condition are false positives. Furthermore, the precision of the model drops to 0 even when we split our data into 3, 10, 25, 30 and 40 bins. We observe the same result in terms of recall. In the case of not mixed splitting in the testing dataset, we observe perfect precision and specificity on the

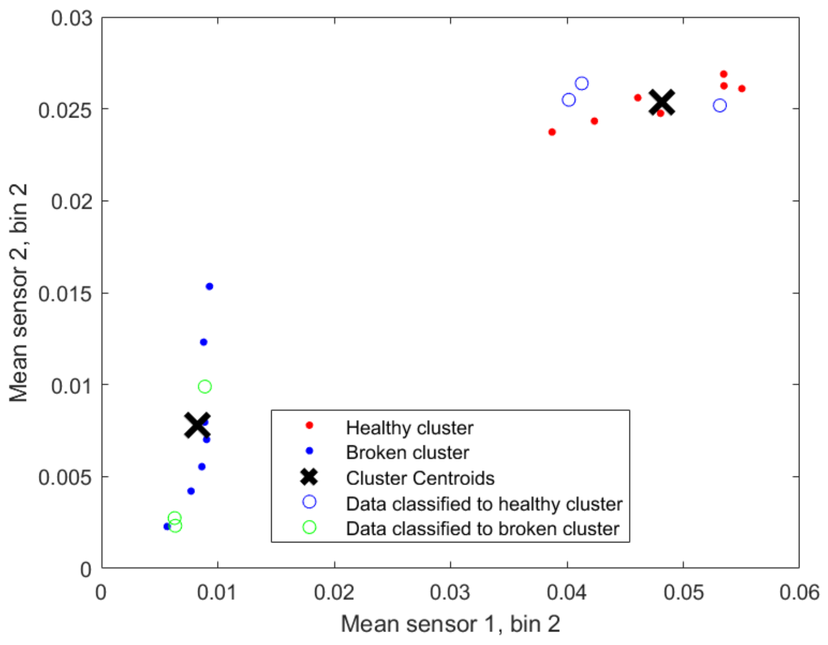


Figure 29. A visualization of k-means clustering applied to our dataset. For the whole dataset, one can see that the k-means clustering cluster the dataset easily as the distance between healthy and broken datasets are very far apart.

test set, the results are similar.

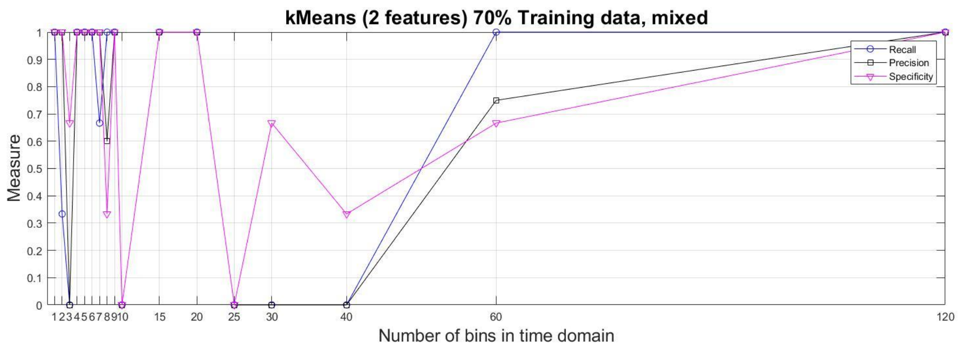


Figure 30. Plot of precision, recall and specificity of the test dataset classification made by k-Means clustering on mixed dataset. We can observe that the k-Means performs very poorly in general when the dataset are split into various number of bins.

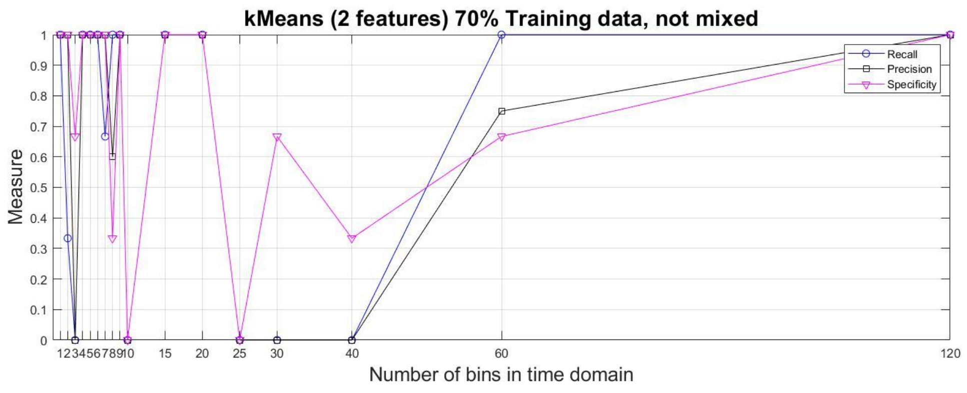


Figure 31. Plot of precision, recall and specificity of the test dataset classification made by k-Means clustering on non-mixed dataset. Similary, k-Means performs very poorly as in the case with mixed dataset.

As it can be seen in figure 32 and 33 provide the confusion matrix of the two classes for the k-means model, when we split data into 2 bins for the mixed version of the k-means model training and into 2 bins for the not mixed version. The confusion matrix shows that the 3 healthy data points are classified correctly. For the broken data group, 2 of the data points are misclassified as healthy. This result we observe when we split the data into 2 bins to train the mixed version of the model, as well as when we split the data into 2 bins for the not mixed version.

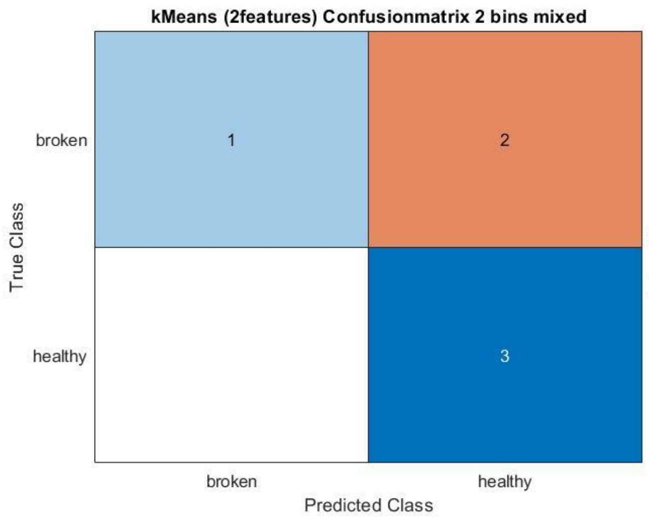


Figure 32. Confusion Matrix of the k-Means classifier on mixed dataset split into 40 bins. We can observe that k-Means clustering performs worse than SVM and random forest, whereby 2 out of the 6 classifications are false negatives.

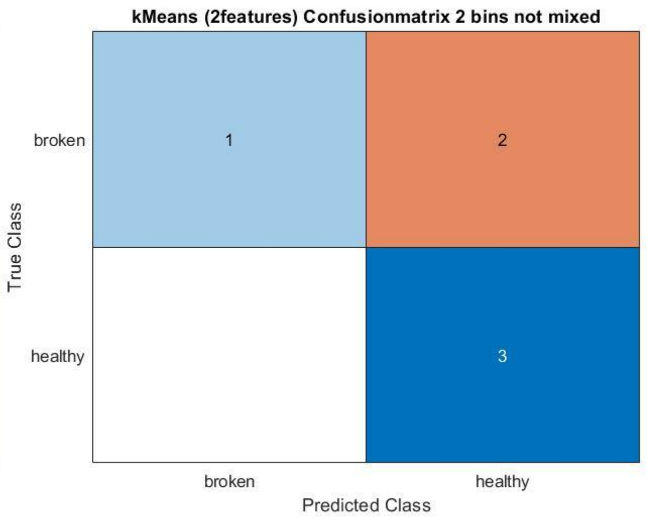


Figure 33. Confusion Matrix of the k-Means classifier on non-mixed dataset split into 40 bins. We can observe that k-Means clustering performs worse than SVM and random forest, similar again to the mixed dataset case.

Figure 34 shows the accuracy on the test set for the mixed k-means for different bin splits of the data in the time domain. We observe 100% accuracy in most bin splits. However, there is a decrease of accuracy to 0% when we split the data in 10 and 25 bins. In figure 35, we observe the same results. Therefore, we may conclude that splitting the data into bins by randomly taking selected slices from each loading from both healthy and broken dataset or always taking the same slice does not affect at all the results of our K-means model.

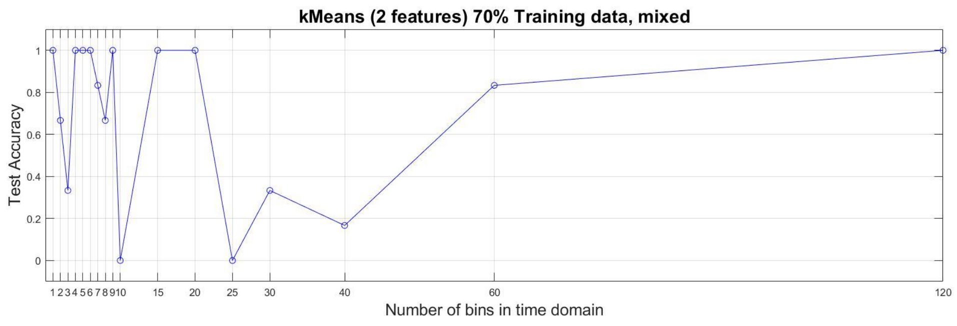


Figure 34. Classification accuracy of K-means clustering on mixed test dataset. In overall, the accuracy plot also shows how poor the k-means clustering is for binned datasets.

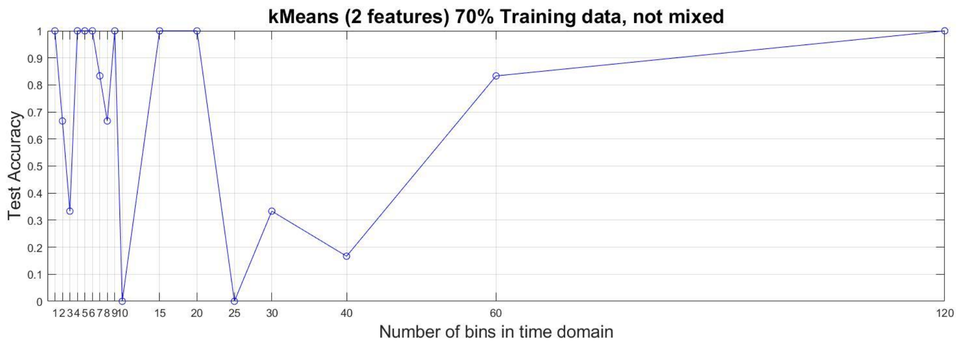


Figure 35. Classification accuracy of K-means clustering on non-mixed test dataset. The same poor accuracy and performance can also be observed when the clustering is done on non-mixed test dataset.

Lastly, we would like to discuss the results of classifying the data by a threshold in only one feature. As already explained in Section 3.1 we consider 60 seconds of data for classification, providing a reasonable period of time for the underlying use-case to our understanding. We visually identified a threshold using twenty randomly drawn time frames, leading to 400 samples. The threshold is tested on 8 distinct time frames.

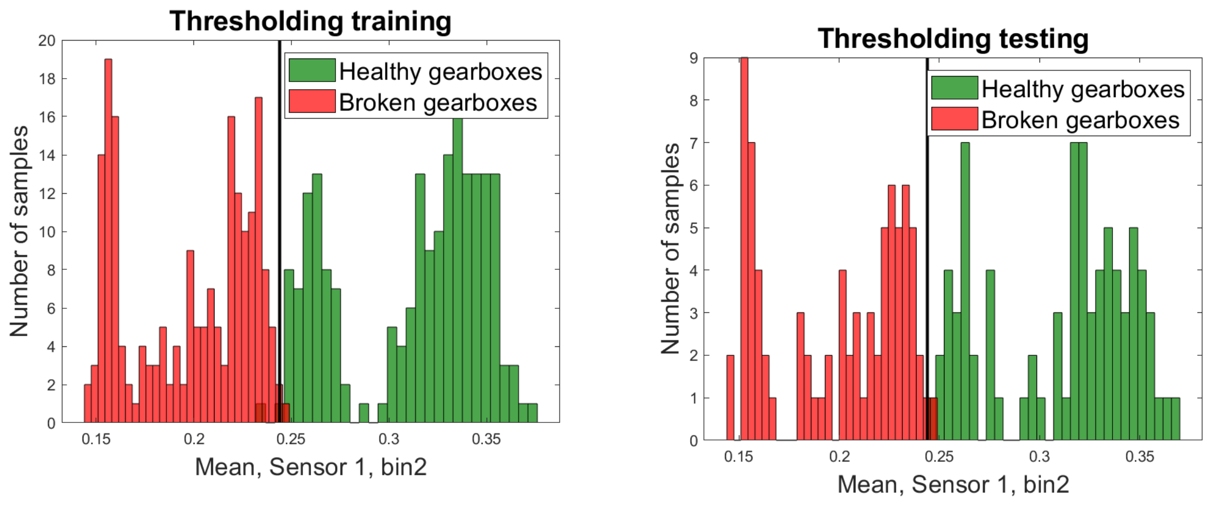


Figure 36. Twenty randomly drawn time samples of 60 seconds have been drawn from all loadings and states (400 snippets). They were used to visually identify a the threshold of 0.244 (Training). Eight distinct samples of 60 seconds are used to test the threshold on unknown data.

As we can already see from Figure 36 the identified threshold separates the test data very well. This is also reflected in the performance measures in Figure 37. As in the other classifier, however, there are false-negatives leading to a non-perfect recall.

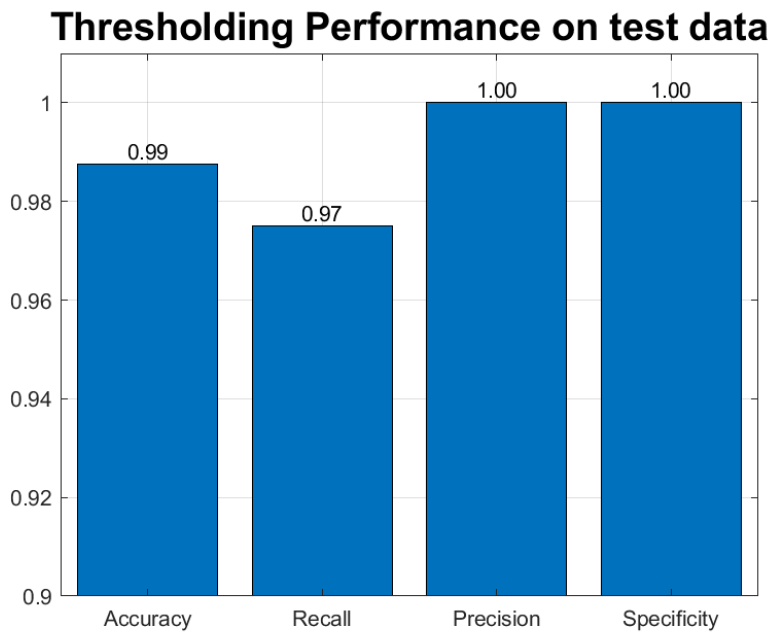


Figure 37. The results of the threshold are satisfactory. In the balanced test set of 160 samples there are 2 false-negatives leading to a slight decrease in recall and accuracy. Precision and specificity are maximal.

5 Discussion

Praveenkumar, T. et al., (2014) collect their own vibration signal using healthy gears and face wear gears, and test each gear with two different speeds and loading conditions. The statistical features are being extracted from the vibration signal, and are being used to train their SVM model. They obtain a classification accuracy of 96.62% for the Gear 1, 92.4% for the Gear 2, 98.75% for the Gear 3 and 100% for the Gear 4. Patel, Raj Kumar, and V.K. Giri. (2016) explores the performance of Random Forest and artificial neural networks for the four class mechanical fault diagnosis of an induction motor using statistical feature extraction. They obtain a very promising performance for the Random forest with a 100% accuracy for the normal and ORF (Outer Raceway Fault) gears and 99.65% and 99.85% accuracy for the IRF (Inner Raceway Fault) and BF (Bearing Ball Fault) gears respectively. In the paper from Zhang Xin, et al. (2016), a modified K-means cluster analysis is performed for bearing fault diagnostics. In contrast to the two above papers, in their study the data of Case Western Reserve University are used for performance evaluation. They obtain an accuracy rate of 58.75% using the traditional K-means and 90.63% using the modified K-means for the data of 14 miles diameters fault. However, when they increase to 21 miles diameters fault, the accuracy of the traditional K-means increases significantly to 85.6%, while the accuracy of the modified K-means goes to 93.1%. All the above papers use statistical feature parameter extraction.

6 Conclusion

This paper is focused on reviewing current research conducted on the automatic gearbox fault diagnosis, analyzing vibration measurement signals for automated fault diagnosis of gearbox and on analyzing the Fast Fourier Transform as the only preprocessing method for the gearbox fault classification on the Gearbox Fault Diagnosis database. The dataset contains vibration signals, collected with the help of four different vibration sensors with 10 loadings for the healthy gears and 10 loadings for the broken ones. Using FFT in analyzing the current signals and the statistical indices for feature extraction, the gearbox fault identification is then performed with three different machine learning techniques, namely Support Vector Machine, Random Forest and K-means, where the highest accuracy is achieved on the SVM model. We observe a tendency for false negatives in all classifiers. Moreover, the results obtained suggest that the supervised classifiers perform better than the unsupervised model. Lastly, after observing the spectrogram, the data is classified by a threshold in only one feature dimension, considering only 60 seconds of the data for the classification. A threshold is visually identified, and the test data are very well separated, with an accuracy of 99 %, and a perfect precision and specificity. However, as in the other classifiers, the tendency of false negatives leads to a non-perfect recall. The positive results obtained in this study encourage future further design and evaluation of the proposed models in the gearbox fault diagnoses. With this work, we succeed in an efficient, simple and practical approach in classifying the data by a threshold using only one feature.

7 Future Work

Based on our analysis we have found that all models have a tendency to produce false negatives, meaning gearbox failure is not detected by the classifier. As this error has the worst consequences in the application of fault diagnosis for wind turbines we should seek to lower the false negative rate even further. For instance this could be done for the threshold by moving the separating value towards the negative class. Additionally, we can also perform in-depth robustness study to our models. Furthermore, our study was limited by the size of the dataset. Hence, it would be interesting to try synthetic data generation to have more input to train the different classifiers. To generate actual benefit of our study we would have to implement the threshold model as deployable application.

References

[1] N. Acır. A support vector machine classifier algorithm based on a perturbation method and its application to ecg beat recognition systems. *Expert Systems with Applications*, 31:150–158, 07 2006.

[2] E. Alickovic and A. Subasi. Medical decision support system for diagnosis of heart arrhythmia using dwt and random forests classifier. *Journal of Medical Systems*, 40:1–12, 2016.

[3] R. Bajric, N. Zuber, G. A. Skrimpas, and N. Mijatovic. Feature extraction using discrete wavelet transform for gear fault diagnosis of wind turbine gearbox. *Shock and Vibration*, 2016:6748469, Dec 2015.

[4] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[5] A. Burkov. *The Hundred-Page Machine Learning Book*. Andriy Burkov, 2019.

[6] K. Chaudhary. Understanding audio data, fourier transform, fft and spectrogram features for a speech recognition system. `https://towardsdatascience.com/understanding-audio-data-fourier-transform-fft-spectrogram-and-speech-recognition-a4072d228520/`, 2020.

[7] W. T. Cochran, J. W. Cooley, D. L. Favin, H. D. Helms, R. A. Kaenel, W. W. Lang, G. C. Maling, D. E. Nelson, C. M. Rader, and P. D. Welch. What is the fast fourier transform? *Proceedings of the IEEE*, 55(10):1664–1674, 1967.

[8] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[9] B. Lantz. *Machine Learning with R: Learn how to Use R to Apply Powerful Machine Learning Methods and Gain an Insight Into Real-world Applications*. Community experience distilled. Packt Publishing, 2013.

[10] C. Maklin. Fast fourier transform. `https://towardsdatascience.com/fast-fourier-transform-937926e591cb/`, January 2019.

[11] V. V. Naoumovitch. *The nature of statistical learning theory / Vladimir N. Vapnik*. Statistics for engineering and information science. Springer, New York, 2nd edition edition, cop. 2000.

[12] Y. Pandya. Gearbox fault diagnosis data - openei datasets. `http://openei.org/datasets/dataset/gearbox-fault-diagnosis-data/`, June 2018.

[13] R. K. Patel and V. Giri. Feature selection and classification of mechanical fault of an induction motor using random forest classifier. *Perspectives in Science*, 8:334–337, 2016. Recent Trends in Engineering and Material Sciences.

[14] T. Praveenkumar, S. Muthusamy, P. Krishnakumar, and R. K I. Fault diagnosis of automobile gearbox based on machine learning techniques. *Procedia Engineering*, 97, 12 2014.

[15] B. R. *DSPA*. Calic J, 2010.

[16] D. Rockmore. The fft: An algorithm the whole family can use. *Computing in Science and Engineering*, 2:60 – 64, 02 2000.

[17] S. Sahoo, R. Laskar, J. Das, and S. Laskar. Gear fault diagnosis and classification using machine learning classifier. pages 69–72, 03 2019.

[18] G. Strang. Wavelets. *American Scientist*, 82(3):250–255, 1994.

[19] X. Tian, J. Lin, K. Fyfe, and M. Zuo. Gearbox fault diagnosis using independent component analysis in the frequency domain and wavelet filtering. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, volume 2, pages II–245, 2003.

[20] e. a. Xin, Zhang. Rolling bearing fault diagnosis using modified k-means cluster analysis. *Vibroengineering PROCEDIA*, 10:155–160.

[21] G. N. Yannakakis and J. Togelius. *Artificial Intelligence and Games*. Springer, 2018. `http://gameaibook.org`.