Strojniški vestnik - Journal of Mechanical Engineering 66(2020)4, 215-226
DOI:10.5545/sv-jme.2020.6563
Original Scientific Paper
Received for review: 2020-01-20
Received revised form: 2020-03-24
Accepted for publication: 2020-03-25

Gaussian Mixture Model Based Classification Revisited: Application to the Bearing Fault Classification

Branislav Panić* – Jernej Klemenc – Marko Nagode
University of Ljubljana, Faculty of Mechanical Engineering, Slovenia

Condition monitoring and fault detection are nowadays popular topic. Different loads, enviroments etc. affect the components and systems differently and can induce the fault and faulty behaviour. Most of the approaches for the fault detection rely on the use of the good classification method. Gaussian mixture model based classification are stable and versatile methods which can be applied to a wide range of classification tasks. The main task is the estimation of the parameters in the Gaussian mixture model. Those can be estimated with various techniques. Therefore, the Gaussian mixture model based classification have different variants which can vary in performance. To test the performance of the Gaussian mixture model based classification variants and general usefulness of the Gaussian mixture model based classification for the fault detection, we have opted to use the bearing fault classification problem. Additionally, comparisons with other widely used non-parametric classification methods are made, such as support vector machines and neural networks. The performance of each classification method is evaluated by multiple repeated k-fold cross validation. From the results obtained, Gaussian mixture model based classification methods are shown to be competitive and efficient methods and usable in the field of fault detection and condition monitoring.
Keywords: Gaussian mixture models, classification, bearing fault estimation, parameter estimation, performance of classification methods

Highlights
- *Gaussian-mixture-model-based classification was applied to the bearing-fault classification.*
- *To discriminate the faulty from non-faulty bearings only simple statistics from vibrational data was used.*
- *Two different datasets, the Case Western Rice University dataset and Bearing vibration data collected under time-varying rotational speed conditions dataset are used.*
- *The Gaussian-mixture-model-based classification method showed to be a competitive and efficient method.*

0 INTRODUCTION

Structural health monitoring, condition monitoring, damage and fault detection are popular topics in engineering [1] and [2]. The early detection of a failure or a fault can be taken as a synonym for the improved maintenance, safety and reliability of a mechanical system or a structure. Constantly evolving fields such as machine learning, data mining and data analysis have greatly facilitated the above-mentioned fields for a great deal of mechanical engineering and engineering generally. Methods from the machine-learning group such as classification methods are widely utilized for different tasks from the diagnostics of aircraft engine blades [3] to the health monitoring of steel plates [4] and the classification of failure modes and the prediction of the shear strength for reinforced concrete beam-column joints [5].

Another great example of the utilization of the classification methods is the bearing-fault classification [6] and [7]. Bearing-fault detection is a very popular problem in mechanical engineering since bearings are one of the most utilized rotational mechanical elements [8] and [9]. This is due to the many phenomena affecting the working conditions of bearings [10] and [11]. Additionally, bearings are mechanical elements that are easily replaceable, yet the untreated fault of a bearing can cause the failure of other elements in a mechanical system, shafts and other [12]. Failures of other elements can cause a high security risk in some applications or larger economic losses due to longer maintenance times in other applications.

Studies on bearing-fault classification differ in two ways. The first type of studies covers different signal-processing techniques for the classification of bearing faults [7] and [13] or for feature extraction and selection from vibrational data, which are then used to enhance the results of an applied classification method [14]. Other studies mostly utilize the different classification methods to obtain better classification results [6] and [15]. This paper is of the latter type. We have applied four types of classification methods based on the Gaussian mixture model (GMM) to the problem of bearing-fault classification. To compare the performance of the GMM-based classification method, three different non-parametric classification methods are used. All the results are obtained on two real-world datasets, the famous *Case Western*

*Corr. Author's Address: University of Ljubljana, Faculty of Mechanical Engineering, Aškerčeva 6, 1000, Ljubljana, Slovenia, branislav.panic@fs.uni-lj.si

University dataset [16] and the *Variable rotational speed bearing fault* dataset [17].

The paper is structured as follows. Section 1 gives the background on GMM-based classification along with a thorough explanation of the different methods and parameter-estimation algorithms. Section 2 gives a brief overview of other non-parametric classification methods. Section 3 tackles the evaluation of the performance of each classification method on the particular dataset. Section 4 describes the datasets and the feature-extraction process. The results and discussion are given in Section 5 and the paper ends with the concluding remarks in Section 6.

1 GAUSSIAN-MIXTURE-MODEL-BASED CLASSIFICATION

Data with a known class affiliation, used for determining a classification model, is often perceived as a realization of random variables. This fact is used in the framework of Bayes decision theory [18]. The classification of new observations to one of K classes is conducted by estimating posterior probabilities $\mathrm{P}(C_i|\mathbf{y})$ for every class and choosing the class j with the maximum posterior probability Eq. (1).

$$j = \arg\max\left(\mathrm{P}\left(C_i|\mathbf{y}\right)\right), \quad i = 1,...,K. \tag{1}$$

The estimation of posterior probabilities for each class $\mathrm{P}(C_i|\mathbf{y})$ is calculated using the Bayes allocation rule (Eq. 2), which depends on probability density function (PDF) $\mathrm{P}(\mathbf{y}|C_i)$ and the apriori probability of each class $\mathrm{P}(C_i)$. The estimation of latter becomes the main problem of classification.

$$\mathrm{P}\left(C_i|\mathbf{y}\right) = \frac{\mathrm{P}\left(C_i\right)\mathrm{P}\left(\mathbf{y}|C_i\right)}{\sum_{j=1}^{N}\mathrm{P}\left(C_j\right)\mathrm{P}\left(\mathbf{y}|C_j\right)}. \tag{2}$$

Estimating the class PDF is not a simple task as clear evidence does not exist as to which probability distribution family to use. The choice of probability distribution affects the discriminating functions between classes. For example, in [19], a Gaussian distribution with the same covariance matrices for each class (homoescadity assumption) is used. This results in linear discriminating functions between classes, illustrated on the first column of plots in Fig. 1, hence the method was named linear discriminant analysis (LDA) [20]. However, if the assumption of homoescadity is removed (the covariance matrices are different for each class), quadratic discriminating functions are achieved, represented in the second column of the plots in Fig 1. This is known as quadratic discriminant analysis (QDA) [20].

The discriminating functions between classes can be more complex. The class distribution can be multi modal or skewed. Hence, an extension of classical linear and quadratic discriminant analysis was made in [21]. This extension, mixture discriminant analysis (MDA) utilizes mixture models (MM), precisely Gaussian mixture models (GMM), for the class PDF. Essentially, GMMs are used for cluster analysis and a semi-parametric probability density estimation. It is shown that GMMs can be used to estimate any continuous density with arbitrary accuracy [22] and [23]. They have a lower footprint on memory usage in comparison with non-parametric density estimators (kernel density estimators) as they do not require all the data to be stored once the parameters have been estimated. Additionally, the utilization of GMMs for estimating the class PDFs results in general non-linear discriminating functions between classes (third column on Fig 1).

1.2 Estimation of Parameters of Gaussian Mixture Models

A GMM is defined as the sum of c differently weighted Gaussian probability density functions where the sum of all weights w_l is equal to 1, Eq. (3), [24]. For example, the GMM used for modeling the class PDF on Fig. 1 contained five components with its mean value given with a yellow star and the covariance matrix represented as a red or blue ellipse.

$$\mathrm{f}\left(\mathbf{y}|\boldsymbol{\Theta}\right) = \sum_{l=1}^{c} w_l \mathrm{f}\left(\mathbf{y}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\right). \tag{3}$$

The difficulty in estimating the parameters $\boldsymbol{\Theta}$ of GMMs lies in the estimation of the number of components c, their weights w_l, mean vectors $\boldsymbol{\mu}_l$ and covariance matrices $\boldsymbol{\Sigma}_l$.

1.2.1 EM Approach

The most commonly used approach for the estimation of weights of components and component parameters (means and covariance matrices) is via the expectation-maximization (EM) algorithm [25]. The EM algorithm iteratively estimates the parameters of GMMs by maximizing the likelihood function. As the EM algorithm requires the number of components and some initial guess of the component weights and component parameters, an additional procedure is involved. The estimation of the GMM parameters is usually carried out via multiple runs of the EM

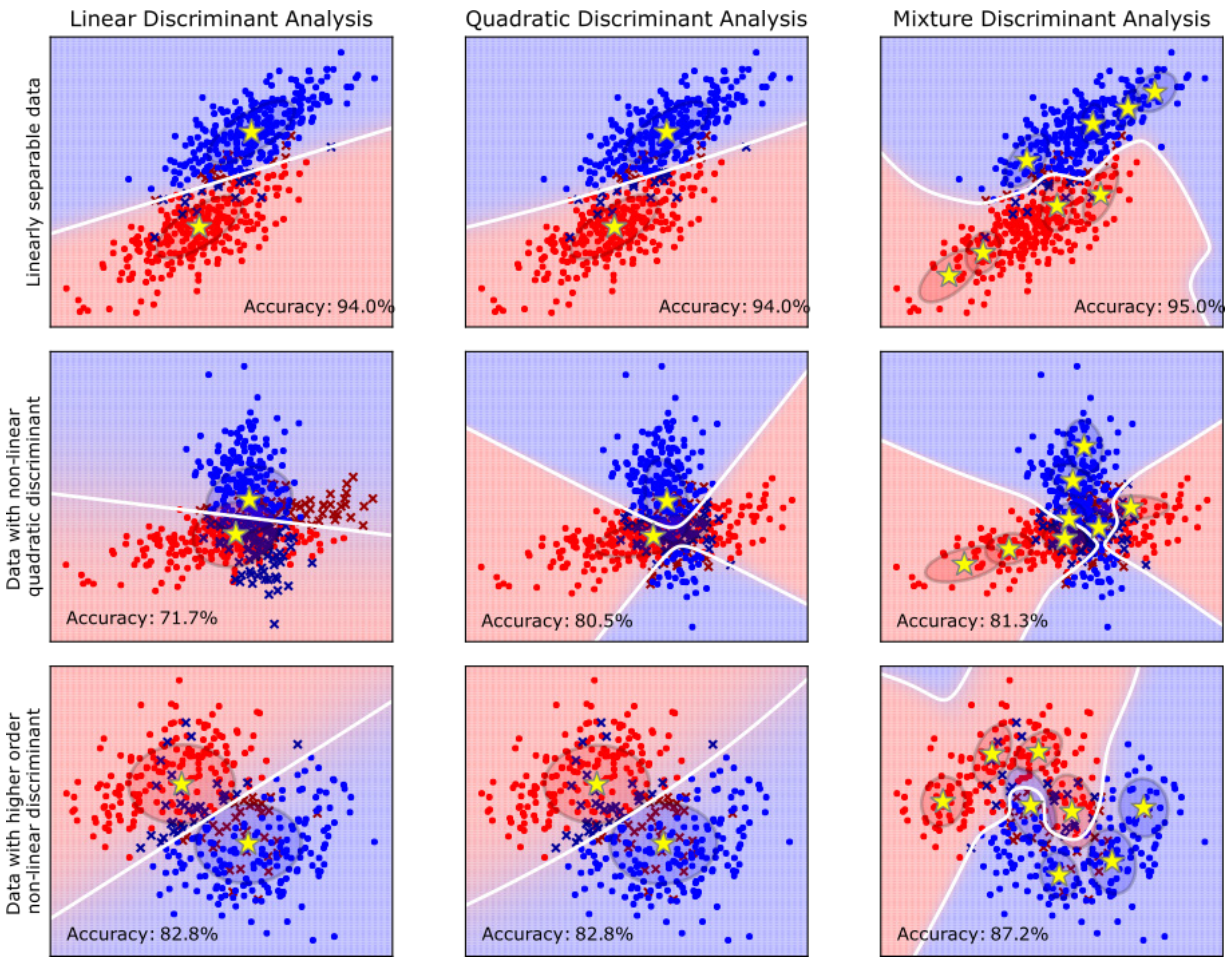

Fig. 1. *Discriminant functions for LDA, QDA and MDA classification methods for the classification problems with linear discriminant, quadratic discriminant and non-linear discriminant function; first column represent LDA method; second column gives plots of QDA method; third column represents MDA method; first row: the dataset has linear separation between classes; second row: the dataset used has quadratic discriminant; third row: the dataset has non-linear discrimant*

algorithm with different numbers of components. The initial guesses of weights of components and component parameters is achieved, for example, either by the random selection of points from the dataset, the *k*-means clustering algorithm or hierarchical clustering [26]. Furthermore, the EM algorithm does not guarantee convergence of the likelihood function for each initial guess of parameters, nor does it guarantee convergence to global optima. Therefore, multiple selections of initial guesses of parameters for each number of components is desirable. This makes the procedure of estimating the parameters of GMMs computationally burdening, especially for large datasets and datasets with a large number of dimensions. For a more in-depth explanation and mathematical derivation of the EM algorithm, readers are referred to [18] and [24].

1.2.2 REBMIX Approach

The rough-enhanced-Bayes mixture estimation (REBMIX) algorithm [27] and [28] can be used to estimate the parameters of a GMM. The algorithm is a numerical procedure that combines an empirical density estimation, mode-finding, clustering and maximum-likelihood estimation procedures for the estimation of such parameters. Instead of specifying the number of components and the initial guesses of component weights and parameters, the input parameters for the REBMIX algorithm are the smoothing parameters for the empirical density estimation procedures, for example, the number of bins in a histogram density estimation or the number of nearest neighbors for a *k*-nearest-neighbors (KNN) density estimation. Another parameter needing to be

specified is the maximum number of components in the GMM. For a given set of input parameters there are multiple estimations of the parameters for a GMM, which differ in terms of the number of components, and the component parameters and weights in the GMM.

1.3 Gaussian Mixture Model Selection for the Class Probability Density Function

In general, the results of both the procedures involving the EM algorithm and REBMIX algorithm are multiple parameters of the GMM, which differs in the number of components, the component parameters and weights. The selection of the appropriate parameters and the number of components is based on calculating the information criterion (IC) and selecting those parameters that yield a minimum value for the IC [24]. The IC is used to penalize complex models and hence avoids over-fitting problems. There is a lot of IC presented in the literature, see Chapter 6 of [24]. Out of them mostly used ones are definitely the Akaike information criteria(AIC) and the Bayesian information criterion (BIC). AIC is defined in Eq. (4), where $M = c - 1 + c \cdot d + c \cdot d(d+1)/2$ is the number of parameters in the d-dimensional GMM for the number of components c and L is the likelihood value.

$$\mathrm{AIC} = -2\log(L) + 2M. \tag{4}$$

The second one, BIC, is defined in Eq. (5). Although AIC is a good criterion it penalizes less than BIC and for a large amount of data it can result in an over-fitted GMM. That being said, for the purpose of density estimation the BIC is usually best suited [29].

$$\mathrm{BIC} = -2\log(L) + M\log(L). \tag{5}$$

1.4 Software Implementations of GMM-Based Classification

Software implementations of the GMM-based classification procedures are applied using the R programming language [30]. The R programming language is mainly used for statistical computing, machine learning and data mining and therefore provides one of best environments for classification problems. For the software implementations the following convention is used: package names are written in italic font; function names are written in bold font.

1.4.1 Mixture Disciminant Analysis

The *mda* package [31] offers GMM-based classification described in [21]. The estimated GMMs for each class PDF have a known number of components in advance. The covariance matrix of each component in the GMM is diagonal and equal for all the components in the estimated GMM. Equal covariance matrices are also kept throughout all the GMMs estimated for different classes of PDF. The estimation of the GMM is achieved using the EM algorithm and *k*-means clustering is used for the initialization technique of the EM algorithm. The R package *mda* offers function **mda** for classification purposes. The user-specified input parameters for the **mda** function are the number of components for each class in the classification model. A simple validation procedure was employed for the selection of the number of components in the GMM. Additionally, each class in the classification problem was assumed to have the same number of components in the GMM and the best number of components was selected based on a minimal training error for each dataset. The number of components was selected to range from 1 to 9.

1.4.2 Model-Based Classification

Another widely used R package implementation for GMM-based classification is the *mclust* package [32]. Model-based classification improves upon the original mda method. The PDF of every class is assumed to follow a parsimonious GMM. Improvements are made in the sense of allowing the GMM to have different covariance structures. These covariance structures are thoroughly described with implementations in [33]. GMMs can have a different number of components for each class. The estimation of the parameters of a GMM is calculated using the EM algorithm coupled with hierarchical clustering initialization (hclust) [34]. The appropriate parameters of GMM are selected via the BIC. The R package *mclust* offers function **MclustDA** which is used for GMM-based classification. User-specified input parameters for the **MclustDA** function is the maximum number of components in the GMM. For the maximum number of components, the default value was 9.

1.4.3 REBMIX-Based Classification

The *rebmix* R package [35] offers GMM-based classification based on an estimation of the GMM for class PDF with the REBMIX algorithm [27], [36] and

[37]. For the estimation of the empirical probability density, the following procedures are implemented: histogram, kernel density and KNN. Different ICs can be used for the assessment of the number of components, component weights and component parameters [27]. The R package *rebmix* offers the function **REBMIX** for an estimation of the GMM for each class. User input parameters are a type of empirical density estimation and can be chosen from the histogram, kernel density estimation or KNN density estimation. Additionally, the user must supply the number of bins in the histogram and kernel density estimation or the number of nearest neighbors for the KNN density estimation. Additionally, the maximum number of components in the GMM is required. For the empirical density estimation we have used a histogram because it offers the fastest estimation of the GMM; and for the smoothing parameter, the number of bins selected was the default value. Additionally, due to the fact that REBMIX algorithm can be used as a standalone procedure or combined with EM algorithm [37] we have used two variants of this implementation, namely *rebmix* and *rebmix&EM*. The *rebmix&EM* used here corresponds to the Exhaustive REBMIX&EM strategy described in [37]. The maximum number of components was kept the same as for the *mda* and *mclust* case, which was 9.

Table 1. *Properties of different GMM bsed classification methods*

	mda	mclust	rebmix	rebmix&EM
Uses EM?	yes	yes	no	yes
EM-init*	k-means	hclust	/	rebmix
Shrink**?	yes	yes	no	no
pros	mild***	diverse	rapid	mild***
cons	limiting	slow	faulty	over-fitting

* How is the initialization of EM algorithm performed?
** Does the method shrink the number of parameters in GMM?
*** Mild refers to the computational intensity of both methods.

The main differences and the advantages and disadvantages of each GMM-based classification method are listed in Table 1. The choice of algorithm for the estimation of GMM parameters may affect classification performance. Three methods use EM algorithm for estimation and only *rebmix* does not. Since the *rebmix* is merely an heuristic, the final estimated parameters of GMM can be degenerated, which is the main disadvantage. On the other hand, it provides rapid estimation compared to the EM algorithm [28]. Additionally, the EM algorithm used for the other three methods may be trapped in a local optima and requires careful initialization [37]. The choice of initial parameters directly affects the final estimated GMM parameters, so we assume that different initialization can have advantages for the classification results. Finally, the GMM has a lot of parameters that need to be estimated. Most parameters belong to the covariance matrices of the different GMM components. Therefore, the general GMM with an unrestricted covariance matrix can produce over-fitting, and this is the main disadvantage of *rebmix&em* method. On the other hand, the *mda* method assumes a hard parsimony, which can probably be limiting. The *mclust* method offers 14 different types of covariance structures [32], which can be fruitful for classification problems. However, since this is very computationally intensive, this method can be quite slow.

2 NON-PARAMETRIC CLASSIFICATION METHODS

Non-parametric methods are also very useful tools for classification purposes. We have selected methods which are, in our opinion, most commonly used for engineering purposes [6] and [15]. In the following paragraphs brief explanations of different classification methods are given.

2.1 Support vector machines

Support vector machines (SVM) create a separating hyperplane between classes in N-dimensional space [38]. The optimal separating hyperplane is determined via a maximal margin between a small amount of selected observations, referred to as support vectors. Estimation of the SVM based classification was carried out using the *e1071* R package [39] which is an interface to the LIBSVM C++ library [40]. The function used for SVM based classification was **svm** with all parameters kept to a default value for both simplicity and a reduction in computational time.

2.2 Artificial Neural Networks

An artificial neural network (ANN) is a classification method which mimics brain structure and information processing in the brain [18]. The structure of a neural network is represented as layers of connected neurons. The structure can be divided into three layers, input layer, hidden layer and output layer. Hidden layer can additionally have more sub layers for more complex information processing, commonly referred to as deep networks. Used R package in this study is *nnet* package [41] and [42], which offers modeling of single hidden layer neural networks. Used function in the *nnet* package was **nnet** with all parameters kept to default value.

2.3 *k*-nearest neighbor

KNN method uses votes of nearest observations with a known class affiliation to decide the class membership of a new observation with an unknown class affiliation [43]. The class with the most votes amongst the *k*-nearest observations is chosen as the class membership of new observations. For the KNN classification method the R package used is *FNN* [44]. The function used in the *fnn* package was **fnn**. The user specified input parameter needed for this classification was the number of nearest neighbors used in the voting stage. The number of nearest neighbors was selected based on the minimal training error. The number of considered nearest neighbors was 2, 5, 10, 15, and 20.

Table 2 summarizes the main advantages and disadvantages of selected non-parametric classification methods.

Table 2. *Properties of different non-parametric classification methods*

Method		Properties
svm	pros	less parameters, less memory intensive, intuitive, rapid
	cons	black-box method, less flexible
nnet	pros	more flexible, can have multiple hidden layers (deep neural networks)
	cons	black-box method, more parameters, can produce overfit, generally slower, more memory intensive
knn	pros	simple, intuitive
	cons	least flexible, most memory intensive (dataset needs to be stored), can be time consuming

3 PERFORMANCE EVALUATION AND FEATURE EXTRACTION

3.1 Performance Evaluation of Classification Method

For a reliable estimation of the performance measures for a classification method on a single dataset, multiple repetitions of the classification with different perturbations of the dataset are needed. One of the techniques mentioned earlier which can be used for this purpose is *k*-fold cross validation [45]. The dataset is split into *k* equally sized subsets (as opposed to random splitting where the data may, for example, be split 70 % and 30 %). All *k* subsets are then used for testing and training purposes. If the dataset is additionally randomly perturbed, different subsets can be obtained and we can perform multiple *k*-fold cross validations.

Most of the measures of fit used in evaluating the performance of classification with a particular method can be found in [46] and [47]. Different measures of fit certainly reveal different aspects of the performance of classification methods. Furthermore, by obtaining multiple values through multiple repeated *k*-fold cross validation of that measure of fit, some useful statistic such as the mean or median can be extracted and used for comparison, as can be seen in Meyers comparison of support vector machines [48].

For the evaluation of performance in a single turn of cross validation, two measures are used. The first is a classification error. The classification error is widely accepted and commonly used measure of fit that is appropriate as a general purpose measure of fit for classification tasks. It is defined as the percentage of wrongly classified observations from a certain dataset in the classification problem. A smaller classification error generally yields a better performance. The other performance measure used here was the computation time of the training and testing phases.

Multiple repeated *k*-fold cross validation yields multiple values of the classification errors and computation times. From the results of multiple repeated *k*-fold cross validation useful statistics can be derived such as, mean or standard deviation (std) of classification errors or computational times. This statistics can give more appropriate representation of the performance versus the single value which is usually obtained with random split of the dataset into train/test datasets.

3.2 Feature Extraction and Construction of Classification Datasets

For this study two different datasets for the bearing-fault classification were used. The first dataset is the widely used and known *Case Western Reserve University* (CWRU) dataset [16]. The second one is the bearing-vibration data under the time-varying rotational speed (VRSB) dataset [17]. All the datasets represent time-series vibration data collected from normal healthy bearings and a faulty bearing with different fault conditions, such as inner/outer race defects or ball defects. The CWRU dataset contains vibrational data for normal/healthy bearings along with vibrational data for bearings with an inner race, outer race and ball defects. The testing is made on 6205-2RS JEM SKF, deep groove ball bearing and 6203-2RS JEM SKF, deep groove ball bearing. Testing load ranged from 0 Nm to 2205 Nm and the testing speed ranged from 1730 r/min to 1797 r/min. Fault sizes were following: 0.1778 mm, 0.3556 mm,

Table 3. *Used statistics for the feature extraction process*

Statistic	Equation	Statistic	Equation	Statistic	Equation
Root mean square	$\mathrm{RMS}=\sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}$	Crest factor	$\mathrm{CF}=\frac{\max(x_i)}{\mathrm{RMS}}$	Kurtosis factor	$\mathrm{KF}=\frac{\mathrm{KV}}{\mathrm{RMS}^4}$
Square root of the amplitude	$\mathrm{SRA}=\left(\frac{1}{N}\sum_{i=1}^{N}\sqrt{\lvert x_i\rvert}\right)^2$	Impulse factor	$\mathrm{IF}=\frac{N\max(x_i)}{\sum_{i=1}^{N}\lvert x_i\rvert}$	Frequency center	$\mathrm{FC}=\frac{1}{N}\sum_{i=1}^{N}f_i$
Kurtosis value	$\mathrm{KV}=\frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i-\mu_x}{\sigma_x}\right)^4$	Margin factor	$\mathrm{MF}=\frac{\max(x_i)}{\mathrm{SRA}}$	Root-mean-square frequency	$\mathrm{RMSF}=\sqrt{\frac{1}{N}\sum_{i=1}^{N}f_i^2}$
Skewness value	$\mathrm{SV}=\frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i-\mu_x}{\sigma_x}\right)^3$	Shape factor	$\mathrm{SF}=\frac{\max(x_i)}{\mathrm{SV}}$	Root variance frequency	$\mathrm{RVF}=\sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_i-\mathrm{FC})^2}$
Peak-to-peak value	$\mathrm{PPV}=\max(x_i)-\min(x_i)$				

0.5334 mm and 0.7112 mm. The other dataset contains only vibrational data for normal/healthy bearings and vibrational data of bearings with with inner-race and outer-race defects. The bearing used for testing was ER16K ball bearing. In previous studies [6], [15] and [49] a plethora of features that could be extracted from the vibrational data were studied, specifically from the time domain, frequency domain or the time-frequency domain using various signal-processing tools such as the Fourier transform, Hilbert transform, Wavelet transform, etc. The feature-extraction part can greatly enhance the results of the classification

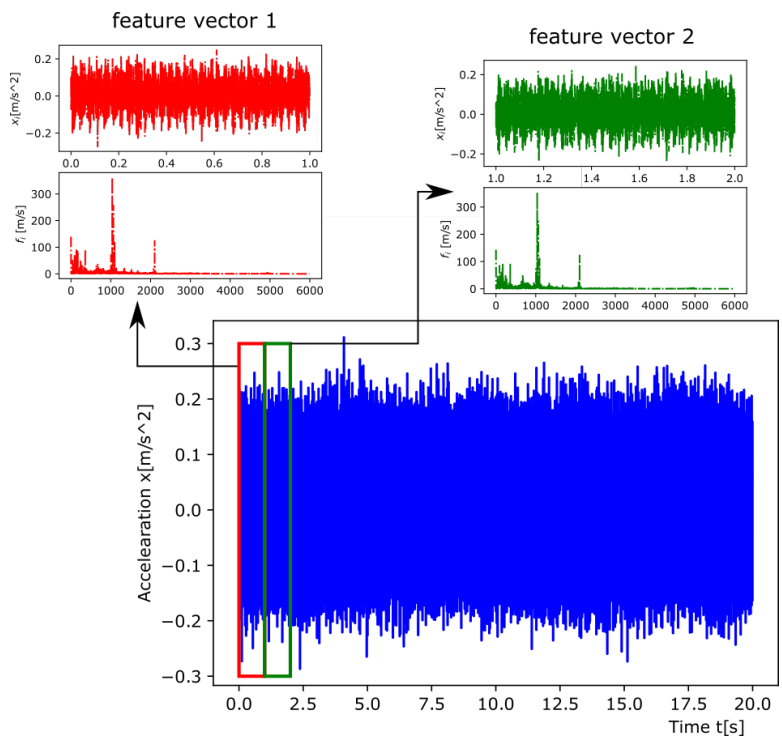

Fig. 2. *Feature-extraction process from the vibrational data of a healthy bearing of the Case Western Reserve University dataset*

and there is a lot of studies emerging on this topic [50]. However, since this paper represents the application of a classification method and its variants to one of the most dominant problems in the field of bearing and rotating-machinery fault detection we will simplify the feature-extraction process to only the statistical features of the vibrational signals in the time and frequency domains.

This resulted in thirteen different most popular statistical features, judging by the literature [6], [15] and [49]. Features are given in Table 3, where x_i is the i^{th} amplitude of the acceleration signal, N is the number of samples in the signal, μ_x is the mean value of the signal, σ_x is the standard deviation of the signal, f_i is the corresponding i^{th} frequency amplitude.

To construct the classification datasets with the presented statistical features an interval of 1s is used, Fig. 2. For example, the CWRU dataset contains signals with a length of 20 s sampled at 12,000 samples per second (sps) or sampled at 48,000 sps. Those signals resulted in 20 instances for the CWRU classification dataset. Table 4 summarizes the characteristics of the constructed classification datasets.

Table 4. *Used data sets*

	Number of instances	Number of features	Number of classes
CWRU	1906	14	4
VSBD	360	14	3

Fig. 3 gives the pseudo code of the algorithm flow for the evaluation of classification methods.

Input: vibrational data, classification method,
number of folds k, number of repeats r;
Output: classification errors $\boldsymbol{E}$ and evaluations times $\boldsymbol{T}$;
1: Extract the features from vibrational data;
2: do:
3: set random partitioning scheme i;
4: split the input dataset into k subsets;
5: do:
6: merge $(k-1)$ subsets so that j subset is left out;
7: estimation time t_t = estimate classification model;
8: evaluation time t_e, classification error e =
evaluate model on j subset;
9: total time $t = t_t + t_e$;
10: merge solution e and t into result arrays $\boldsymbol{E}$ and $\boldsymbol{T}$;
11: while $j < k$;
12: while i < r;

Fig. 3. *Evaluation of classification method using the multiple repeated k-fold cross validation*

4 RESULTS AND DISCUSSION

First, let us address the parameters used for the multiple repeated k-fold cross validation. The number k of folds was set to 5. The number of random perturbations of the datasets was set to 10, meaning that for each dataset the methods were applied 50 times and 50 different values of the classification errors and computational times were acquired. Those results were illustrated using box-plots. A box-plot represents the distribution of the data, where the boundaries of the box represent the 25 % and 75 %. The line inside the box represents the median value of that distribution. Additionally, for each dataset and classification method the mean value and standard deviation are given in tables.

4.1 CWRU Classification Dataset

The results are given in Fig. 4 and Table 5. The results of the classification errors (Fig. 4a plot) yielded three clusters of performers. Standalone *rebmix* gave the worst performance with respect to the accuracy of the classification. The best performers were the methods *nnet* and *rebmix&em*. The methods *mda*, *mclust*, *svm* and *nnet* were the average performers. On other hand, judging by the computational times (Fig. 4b) the *rebmix* method was the fastest method, performing 2 to 10 times faster than other methods. The *nnet* and *svm* methods yielded equal performance with respect to the computational time and were the second-fastest performing methods. A comparison of only the GMM-based classification methods on the CWRU dataset yielded *rebmix&em* as the best-performing method. This method yielded the smallest values of the classification error, while preserving the shortest computational times, judging by the mean values and standard deviations in Table 5.

Table 5. *Mean and standard deviation of the results on the CWRU dataset*

Method	Error [%] mean (std)	Time [s] mean (std)
mda	23.22 (2.67)	0.42 (0.41)
mclust	24.34 (2.80)	0.48 (0.45)
rebmix	34.85 (2.78)	0.05 (0.05)
rebmix&em	9.89 (3.15)	0.27 (0.35)
knn	23.97 (1.88)	0.36 (0.33)
svm	21.77 (2.19)	0.09 (0.08)
nnet	10.32 (2.61)	0.10 (0.10)

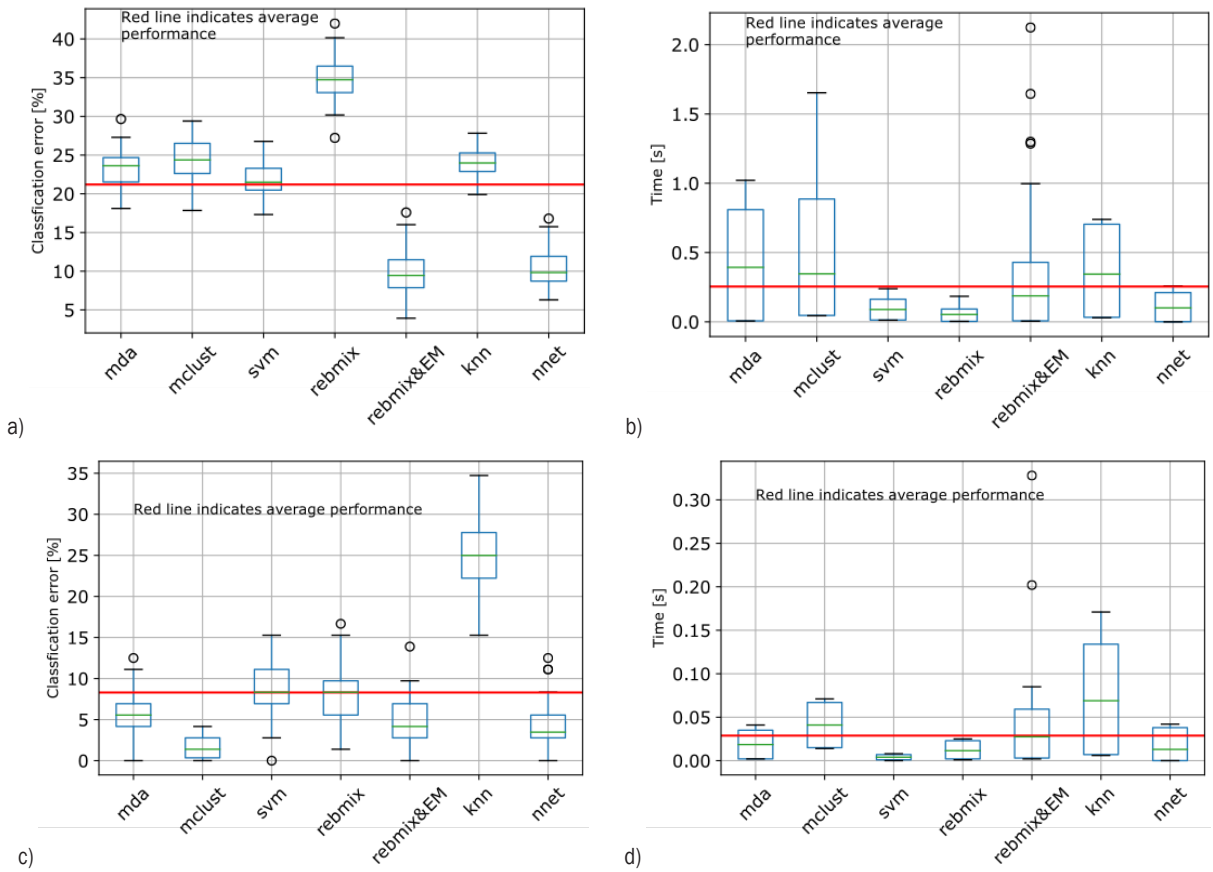

Fig. 4. *Box-plots of a) classification errors, and b) computational times on CWRU dataset, c) classification errors, and d) computational times on VSBD dataset*

Although, the standalone *rebmix* method had convincingly the shortest computational time, the results of the classification error were much larger than the other GMM classification methods, which deteriorates the overall performance of the *rebmix* method. The *nnet* method gave the best overall performance on the CWRU dataset. The method gave fast results while preserving almost the smallest classification error.

4.2 VSBD Classification Dataset

The results for the VSBD dataset are given in Fig. 4 and Table 6. Judging by the box-plots of the classification errors (Fig. 4c) the best performer was the *mclust* method. The worst performer was the *knn* method.

Additionally, the average performers can be placed into two clusters: the first one consisting of the methods *svm* and *rebmix*, which gave a slightly worse performance, and the methods *mda*, *nnet* and *rebmix&em*, which gave a slightly better performance.

Judging by the box-plots of the computational times (Fig. 4d), the method *svm* had the best performance, while *knn* had the worst performance. The methods *rebmix* and *nnet* had equal performance, which was a little better than the average performance and the methods *mda*, *mclust* and *rebmix&em* gave an average performance.

Table 6. *Mean and standard deviation of results on VSBD dataset*

Method	Error [%] mean (std)	Time [s] mean (std)
mda	5.72 (2.67)	0.02 (0.02)
mclust	1.58 (1.21)	0.04 (0.03)
rebmix	7.83 (3.47)	0.01 (0.01)
rebmix&em	4.69 (2.79)	0.04 (0.04)
knn	25.25 (4.67)	0.07 (0.06)
svm	8.94 (3.45)	0.003 (0.003)
nnet	4.11 (2.74)	0.01 (0.01)

On the VSBD dataset all the methods were extremely fast. This can be linked to the small number

of instances in the VSBD dataset. Therefore, the computation time has a far smaller impact than in the case of the CWRU dataset. The best-performing method of the GMM classification methods was the *mclust* method, although the *mda* and *rebmix* methods had smaller values of the mean and the standard deviation of the computation time (see Table 6). In the overall comparison, the *nnet* method had a slight deterioration in performance with respect to the classification errors, while it preserved the above-average value of the computational time.

5 CONCLUSIONS

In this work we have investigated the performance of different GMM-based classification methods on the problem of the bearing-fault classification. The performance was evaluated on two publicly available datasets with bearing-vibrational data. To construct the classification datasets out of which the bearing faults can be classified, we used just simple processing techniques and only statistical features from the data are estimated. We opted out of using more complicated signal-processing techniques, for example, the wavelet transform, because of the pure simplicity. If the classification method can perform well on extracted simple statistics, it will perform even better on more sophisticated ones. Therefore, the general applicability is evaluated.

From the results obtained on both datasets one of the GMM-based classification method variants had the best results: on the CWRU dataset the *rebmix&EM* method and on the VSBD the *mclust* method. Although the variants of the GMM based classification did not yield the most satisfying results for the computational time, this was mostly caused by the slow convergence of the EM algorithm. Since there is a lot of effort in the research for speeding up the convergence of the EM algorithm **[24]**, this can be utilized to improve the computational dependency of the GMM-based classification methods. On other hand, standalone *rebmix* proved to be fast variant, although the other results were unsatisfying. As expected, the *nnet* method gave a good trade off between accuracy and computational time on both used datasets, which ultimately positioned it as a serious rival to GMM based methods.

We will end this article by providing some insights into our future work. Different speeding-up techniques of the EM algorithm along with different pre-processing techniques for the REBMIX algorithm will be tested. Finally, more datasets for the engineering-based classification tasks will be tested.

6 ACKNOWLEDGEMENTS

The authors acknowledge the financial support from the Slovenian Research Agency (research core funding No. 1000-18-0510).

7 REFERENCES

[1] Salehi, H., Burgueño, R. (2018). Emerging artificial intelligence methods in structural engineering. *Engineering Structures*, vol. 171, p. 170-189, **DOI:10.1016/j.engstruct.2018.05.084**.

[2] Nasiri, S., Khosravani, M.R., Weinberg, K., (2017). Fracture mechanics and mechanical fault detection by artificial intelligence methods: A review. *Engineering Failure Analysis*, vol. 81, p. 270-293, **DOI:10.1016/j.engfailanal.2017.07.011**.

[3] Rabcan, J., Levashenko, V., Zaitseva, E. Kvassay, M., Subbotin, S. (2019). Non-destructive diagnostic of aircraft engine blades by fuzzy decision tree. *Engineering Structures*, vol. 197, p. 109396, **DOI:10.1016/j.engstruct.2019.109396**.

[4] Hasni, H., Alavi, A.H., Lajnef, N., Abdelbarr, M. Masri, S.F., Chakrabartty, S. (2017). Self-powered piezo-floating-gate sensors for health monitoring of steel plates. *Engineering Structures*, vol. 148, p. 584-601, **DOI:10.1016/j.engstruct.2017.06.063**.

[5] Mangalathu, S., Jeon, J.-S. (2018). Classification of failure mode and prediction of shear strength for reinforced concrete beam-column joints using machine learning techniques. *Engineering Structures*, vol. 160, p. 85-94, **DOI:10.1016/j.engstruct.2018.01.008**.

[6] Pan, H., He, X., Tang, S., Meng, F. (2018). An improved bearing fault diagnosis method using one-dimensional CNN and LSTM. *Strojniški vestnik - Journal of Mechanical Engineering*, vol. 64, no. 7-8, p. 443-452, **DOI:10.5545/sv-jme.2018.5249**.

[7] Do, V.T., Nguyen, L.C. (2016). Adaptive empirical mode decomposition for bearing fault detection. *Strojniški vestnik - Journal of Mechanical Engineering*, vol. 62, no. 5, p. 281-290, **DOI:10.5545/sv-jme.2015.3079**.

[8] Nejad, A.R., Odgaard, P.F., Gao, Z., Moan, T. (2014). A prognostic method for fault detection in wind turbine drivetrains. *Engineering Failure Analysis*, vol. 42, p. 324-336, **DOI:10.1016/j.engfailanal.2014.04.031**.

[9] Liu, W.Y. (2013). The vibration analysis of wind turbine blade–cabin–tower coupling system. *Engineering Structures*, vol. 56, p. 954-957, **DOI:10.1016/j.engstruct.2013.06.008**.

[10] Cheng, L.Z., Liu, D.K., Wang, Y., Chen, A.Q. (2019). Load distribution and contact of axle box bearings in electric multiple units. *International Journal of Simulation Modelling*, vol. 18, no. 2, p. 290-301, **DOI:10.2507/IJSIMM18(2)475**.

[11] Edler, J.,Tic, V., Lovrec, D. (2019). 1-D simulation model of a progressive flow controller for hydrostatic bearings. *International Journal of Simulation Modelling*, vol. 18, no. 2, p. 267-278, **DOI:10.2507/IJSIMM18(2)472**.

[12] Liu, J. (2020). A dynamic modelling method of a rotor-roller bearing-housing system with a localized fault including the additional excitation zone. *Journal of Sound and Vibration*, vol. 469 p. 115144, **DOI:10.1016/j.jsv.2019.115144**.

[13] Lu, S., Zheng, P., Liu, Y., Cao, Z., Yang, H., Wang, Q. (2019). Sound-aided vibration weak signal enhancement for bearing

fault detection by using adaptive stochastic resonance. *Journal of Sound and Vibration*, vol. 449, p. 18-29, **DOI:10.1016/j.jsv.2019.02.028**.

[14] Li, Y., Yang, Y., Wang, X., Liu, B., Liang, X. (2018). Early fault diagnosis of rolling bearings based on hierarchical symbol dynamic entropy and binary tree support vector machine. *Journal of Sound and Vibration*, vol. 428, p. 72-86, **DOI:10.1016/j.jsv.2018.04.036**.

[15] Sobie, C., Freitas, C., Nicolai, M. (2018). Simulation-driven machine learning: Bearing fault classification. *Mechanical Systems and Signal Processing*, vol. 99, p. 403-419, **DOI:10.1016/j.ymssp.2017.06.025**.

[16] Loparo, K. (1998). Bearings vibration data set, Cast Western Reverse University, from: *https://csegroups.case.edu/bearingdatacenter/home*, accessed on 2019-12-12.

[17] Huang, H., Baddour, N., (2018). Bearing vibration data collected under time-varying rotational speed conditions. *Data in Brief*, vol. 21, p. 1745-1749, **DOI:10.1016/ j.dib.2018.11.019**.

[18] Bishop, C. (2006). *Pattern Recognition and Machine Learning,* Springer-Verlag, Berlin, Heidelberg.

[19] Fisher, R.A. (1936). The use of multiple measurements in taxonomic problems. *Annual Eugenics*, vol. 7, no. 2, p. 179-188, **DOI:10.1111/ j.1469-1809.1936.tb02137.x**.

[20] Rao, C.R. (1948). The utilization of multiple measurements in problems of biological classification. Journal of the Royal Statistical Society. Series B (Methodological), vol. 10, no. 2, p. 159-203, **DOI:10.1111/j.2517-6161.1948.tb00008.x**.

[21] Hastie, T., Tibshirani, R. (1996). Discriminant analysis by gaussian mixtures. Journal of the Royal Statistical Society. Series B (Methodological), vol. 58, no. 1, p. 155-176, **DOI:10.1111/j.2517-6161.1996.tb02073.x**.

[22] Escobar, M.D., West, M. (1995). Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, vol. 90, no. 430, p. 577-588, **DOI:10.1080/01621459.1995.10476550**.

[23] Roeder, K., Wasserman, L. (1997). Practical Bayesian density estimation using mixtures of normals. *Journal of the American Statistical Association*, vol. 92, no. 439, p. 894-902, **DOI:10.1080/01621459.1997.10474044**.

[24] McLachlan, G., Peel, D. (2000). *Finite Mixture Models*, John Wiley and Sons, New York, DOI:10.1002/0471721182.

[25] Dempster, A.P., Laird, N.M., Rubin, D.B., (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society, Series B (Methodological)*, vol. 39, no. 1, p. 1-38, **DOI:10.1111/j.2517-6161.1977.tb01600.x**.

[26] Melnykov, V., Melnykov, I. (2012). Initializing the EM algorithm in gaussian mixture models with an unknown number of components. Computational Statistics and Data Analysis, vol. 56, no. 6, p. 1381-1395, **DOI:10.1016/j.csda.2011.11.002**.

[27] Nagode, M. (2015). Finite mixture modeling via REBMIX. *Journal of Algorithms and Optimization*, vol. 3, no. 2, p. 14-28, **DOI:10.5963/JA0030200**.

[28] Ye, X.W., Xi, P.S., Nagode, M. (2019). Extension of REBMIX algorithm to von Mises parametric family for modeling joint distribution of wind speed and direction. *Engineering Structures*, vol. 183, p. 1134-1145, **DOI:10.1016/j.engstruct.2018.08.035**.

[29] Biernacki, C., Celeux, G., Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 7, p. 719-725, **DOI:10.1109/34.865189**.

[30] R Core Team, 2014, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, from: *http://www.R-project.org/*, accesed on 2019-12-12.

[31] Hastie, T., Tibshirani, R., Leisch, F., Hornik, K., Ripley, B.D. (2017). mda: Mixture and Flexible Discriminant Analysis, from: *https:// CRAN.R-project.org/package=mda*, accesed on 2019-12-12.

[32] Scrucca, L., Fop, M., Murphy, T.B., Raftery, A.E. (2016). mclust 5: Clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, vol. 8, no. 1, p. 205-233, **DOI:10.32614/RJ-2016-021**.

[33] Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition*, vol. 28, no. 5, p. 781-793, **DOI:10.1016/0031-3203(94)00125-6**.

[34] Scrucca, L., Raftery, A.E. (2015). Improved initialisation of model-based clustering using Gaussian hierarchical partitions. *Advances in Data Analysis and Classification*, vol. 9, no. 4, p. 447-460, **DOI:10.1007/s11634-015-0220-z**.

[35] Nagode, M., Panić B. (2019). Finite Mixture Modeling, Clustering & Classification, from: *https://cran.r-project.org/web/packages/rebmix*, accesed on 2019-12-12.

[36] Nagode, M., (2018), Multivariate normal mixture modeling, clustering and classification with the rebmix package. *ArXiv e-prints*, arXiv:1801.08788.

[37] Panić, B., Klemenc, J., Nagode, M. (2020). Improved initialization of the EM algorithm for mixture model parameter estimation. *Mathematics*, vol. 8, no. 3, p. 373, **DOI:10.3390/math8030373**.

[38] Cortes, C., Vapnik, V. (1995). Support-vector networks. *Machine Learning*, vol. 20, no. 3, p. 273-297, **DOI:10.1023/A:1022627411411**.

[39] Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F. (2018). e1071: Misc Functions of the Department of Statistics, Probability Theory Group, from: *https://cran.r-project.org/web/packages/e1071*, accesed on 2019-12-12.

[40] Chang, C., Lin, C. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, p. 27:1-27:27, **DOI:10.1145/1961189.1961199**.

[41] Ripley, B. (2017). Feed-Forward Neural Networks and Multinomial Log-Linear Models, from: *https://cran.r-project.org/web/packages/nnet*, accesed on: 2019-12-12.

[42] Venables, W.N., Ripley, B.D. (2002). Modern Applied Statistics with S, Springer, New York.

[43] Altman, N.S. (1992). An introduction to Kernel and nearest-neighbor nonparametric regression. The American Statistician, vol. 46, no. 3, p. 175-185, **DOI:10.1080/00031305.1992.10475879**.

[44] Beygelzimer, A., Kakadet, S., Langford, J., Arya, S., Mount, D, Li, S. (2018). fnn: Fast nearest neighbor search algorithms and applications, from: *https://cran.r-project.org/web/packages/FNN/*, accesed on: 2019-12-12.

[45] Hastie, T., Tibshirani, R., Friedman, J. (2009). *The Elements of Statistical Learning*, Springer New York, **DOI:10.1007/978-0-387-84858-7**.

[46] Sokolova, M., Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, vol. 45, no. 4, p. 427-437, **DOI:10.1016/j.ipm.2009.03.002**.

[47] Ferri, C., Hernández-Orallo, J., Modroiu, R. (2009). An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, vol. 30, no. 1, p. 27-38, **DOI:10.1016/j.patrec.2008.08.010**.

[48] Meyer, D., Leisch, F., Hornik, K. (2003). *Benchmarking Support Vector Machines*, Report Series, Vienna.

[49] Li, H., Liu, T., Wu, X., Chen, Q. (2019). Research on bearing fault feature extraction based on singular value decomposition and optimized frequency band entropy. *Mechanical Systems and Signal Processing*, vol. 118, p. 477-502, **DOI:10.1016/j.ymssp.2018.08.056**.

[50] Liu, J., Xu, Z, Zhou, L., Yu, W., Shao, Y. (2019). A statistical feature investigation of the spalling propagation assessment for a ball bearing. *Mechanism and Machine Theory*, vol. 131, p. 336-350, **DOI:10.1016/j.mechmachtheory.2018.10.007**.