UNIVERSITÀ DI PISA
Scuola di Dottorato in Ingegneria "Leonardo da Vinci"

Corso di Dottorato di Ricerca in
INGEGNERIA DELL'INFORMAZIONE

Tesi di Dottorato di Ricerca

ADVANCED VIBRATION ANALYSIS FOR THE DIAGNOSIS AND PROGNOSIS OF ROTATING MACHINERY COMPONENTS WITHIN CONDITION-BASED MAINTENANCE PROGRAMS

Sara Lioba Volpi

Anno 2011

UNIVERSITÀ DI PISA

Scuola di Dottorato in Ingegneria "Leonardo da Vinci"

Corso di Dottorato di Ricerca in
INGEGNERIA DELL'INFORMAZIONE

Tesi di Dottorato di Ricerca

ADVANCED VIBRATION ANALYSIS FOR THE DIAGNOSIS AND PROGNOSIS OF ROTATING MACHINERY COMPONENTS WITHIN CONDITION-BASED MAINTENANCE PROGRAMS

Autore:

Sara Lioba Volpi ……

Relatori:

Prof. Beatrice Lazzerini ……

Prof. Francesco Marcelloni ……

Anno 2011
SSD ING-INF/05

Nunquam invenietur,
si contenti fuerimus inventis[1].

- L.A. Seneca[2] -

[1] Nothing would ever be found, if we felt satisfied with our discoveries.
[2] Naturales Quaestiones, 6, 5, 2.

Acknowledgments

I would like to acknowledge my supervisor Prof. Beatrice Lazzerini, University of Pisa (Italy), who advised me during my Ph.D studies giving me full support in the development of my research and in my professional growth. I would really like to thank my co-supervisor Prof. Francesco Marcelloni for his support during my Ph.D studies.

I wish to express my sincere thanks to Prof. Oscar Cordón, European Centre for Soft Computing (ECSC, Mieres, Spain), who introduced me into the field of the cost-sensitive classification and gave me guidance and constructive suggestions during my visiting period in Mieres.

I owe my deepest gratitude to Prof. Dan C. Stefanescu, Suffolk University (Boston, U.S.), who contributed to a relevant part of my research on one-class classifier with valuable suggestions and discussions and generously spent his time and efforts to organize my visiting period in Boston.

I am very grateful to Prof. Piero Bonissone, Chief Scientist at GE Global Research (Niskayuna, NY, U.S.), Prof. Prof. Lakhmi Jain, University of South Australia, and Prof. Dan C. Stefanescu who kindly agreed to review my thesis and dedicated their time to provide deep feedbacks on the current work and valuable ideas for future extensions.

During my Ph.D, I shared my working and my leasure time with many people at the department of Information Engineering at the University of Pisa, at Suffolk University and at ECSC and, to them, go my deepest thanks. Thanks for the several discussions on topics not necessarily related to my research area which keep my mind open to different subjects. In particular I would like to thank Mario for his precious advices, Kate for her kindness and Charo for letting me know

how wonderful is Spain and the people who live there.

I would really like to thank Marco for his love. Thanks for the infinitive patience, costant support and encouragement that you gave and continue to give me every day of my life. Thank for always being there when I need you.

I am really grateful to my family for all the support they gave me all over my life.

I would like to express my thanks to all my friends and, especially, to Lorenzo for his precious friendship through so many years.

Finally, I wish to acknowledge Avio Propulsione Aerospaziale, via I Maggio, 99, Rivalta di Torino, Italy, for having provided the set of experimental data used for this work.

Sara

Pisa, 28 February 2011

A hundred times every day I remind myself
that my inner and outer life depend upon
the labors of other men, living and dead, and that
I must exert myself in order to give in the measure
as I have received and am still receiving.

- A. Einstein -

Vita

11-28-1983 Born, Castelfiorentino (FI), Italy

2011

Teaching Assistant
Department of Information Engineering
University of Pisa, Pisa, Italy

2010

Research Scholar
European Centre of Soft Computing, Mieres, Spain

Teaching Assistant
Department of Information Engineering
University of Pisa, Pisa, Italy

2009

Senior Lecturer and Research Scholar
Maths&Computer Science Department
Suffolk University, Boston (MA), United States

Teaching Assistant
Department of Information Engineering
University of Pisa, Pisa, Italy

2008

Teaching Assistant
Department of Information Engineering
University of Pisa, Pisa, Italy

Second-level Degree in Engineering
Final mark: 100/100 cum laude
Sant'Anna School, Pisa, Italy

2007 *Student Internship*
Fermi National Accelerator Laboratory
Batavia (IL), United States

Master Degree in Computer Engineering
Final mark: 110/110 cum laude
University of Pisa, Pisa, Italy

2005 *Student Internship*
Internazionale Marmi e Macchine Carrara Spa
Carrara (MS), Italy

Bachelor Degree in Computer Engineering
Final mark: 110/110 cum laude
University of Pisa, Pisa, Italy

Publications

Books

1. S.L. Volpi, *Introduction to Matlab and PRTools*, ed. S.E.U., Pisa 2010.

Journals

1. B. Lazzerini, S.L. Volpi, *Classifier ensembles to improve the robustness to noise of bearing fault diagnosis*, Pattern Analysis & Applications, pp. 1–17, 2011.

Conference proceedings

1. M. Cococcioni, E. D'Andrea, B. Lazzerini, S.L. Volpi, *Short-time forecasting of renewable production energy in solar photovoltaic installations*, Int. Conf. on Competitive and Sustainable Manufacturing, Products and Services (APMS'10), Como, Italy, October 2010.

2. S.L. Volpi, M. Cococcioni, B. Lazzerini, D. Stefanescu, *Rolling element bearing diagnosis using convex hull*, IEEE World Congress on Computational Intelligence (WCCI'10), vol. 1, pp. 1–8, Barcelona, Spain, July 2010.

3. B. Lazzerini, S.L. Volpi, *Noise assessment in the diagnosis of rolling element bearings*, Int. Conf. on Intelligent Computing and Cognitive Informatics (ICICCI'10), vol. 1, pp. 227–230, Kuala Lumpur, Malaysia, June 2010.

4. M. Cococcioni, B. Lazzerini, S.L. Volpi, *Rolling element bearing fault classification using soft computing techniques*, IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC'09), vol. 1, pp. 4926–4931, Hyatt Regency Riverwalk, San Antonio, Texas, U.S., October 2009.

5. M. Cococcioni, B. Lazzerini, S.L. Volpi, *Automatic diagnosis of defects of rolling element bearings based on computational intelligence techniques*, IEEE Int. Conf. on Intelligent Systems Design and Applications (ISDA'09), vol. 1, pp. 970–975, Pisa, Italy, November–December 2009.

6. S.L. Volpi, M. Antonelli, B. Lazzerini, F. Marcelloni, D. Stefanescu, *Segmentation and reconstruction of the lung and the mediastinum volumes in CT images*, IEEE 2nd Int. Symp. on Applied Sciences in Biomedical and Communication Technologies (ISABEL'09), vol. 1, pp. 1–6, Bratislava, Slovak Republic, November 2009.

7. S.L. Volpi, B. Lazzerini, D. Stefanescu, *Time Evolution analysis of bearing faults*, IASTED Int. Conf. on Intelligent Systems and Control (ISC'09), Cambridge, Massachusetts, U.S., November 2009.

8. M. Cococcioni, B. Lazzerini, S.L. Volpi, *Bearing condition monitoring using classifier fusion*, IASTED Int. Conf. on Artificial Intelligence and Soft Computing (ASC'09), vol. 1, pp. 131–137, Palma de Mallorca, Spain, October 2009.

Abstracts, presentations and seminars

1. S.L. Volpi, *Classifier ensembles to improve the robustness to noise of bearing fault diagnosis*, Ph.D. Workshop, Advances in Computer Systems and Networks, Pisa, Italy, November 2010.

2. S.L. Volpi, *The use of Matlab in decision support intelligent systems*, Faculty of Engineering, University of Pisa, April-May 2010.

3. S.L. Volpi, *Intelligent systems in a Matlab environment*, Faculty of Engineering, University of Pisa, March-April 2010.

4. S.L. Volpi, *Automatic diagnosis and time-evolution analysis of rolling element bearings defects using soft computing techniques*, Ph.D. Workshop, Advances in Computer Systems and Networks, Pisa, Italy, November 2009.

5. S.L. Volpi, *Segmentation and reconstruction of the lung and mediastinum fields in traditional CT and PET/CT images*, Ph.D. Workshop, Advances in Computer Systems and Networks, Pisa, Italy, November 2008.

Theses

1. S.L. Volpi, *Design and realization of a lung CAD system for PET/CT exams*, Second-level Degree in Engineering, Sant'Anna School of Advanced Studies of Pisa, Pisa, Italy, 2008.

2. S.L. Volpi, *Design and realization of the module for the extraction of the mediastinum and the lung area from DICOM images in a lung CAD system for CT exams*, Master Degree in Computer Engineering, University of Pisa, 2007.

3. S.L. Volpi, *Uniform Sampling Control*, Bachelor Degree in Computer Engineering, University of Pisa, 2005.

SOMMARIO

I macchinari utilizzati nell'ambito industriale sono soggetti a deterioramento nel tempo e per l'uso, per cui risulta di estrema importanza l'attuazione di un programma di manutenzione allo scopo di evitare il verificarsi di guasti che possono portare a conseguenze anche disastrose. La letteratura si è focalizzata sullo sviluppo di strategie di manutenzione ottimali come la "condition-based maintenance" (CBM, manutenzione basata sull'evento) per aumentare l'affidabilità, evitare guasti e ridurre i costi legati alla manutenzione stessa. La CBM si propone di identificare in tempo la presenza e la gravità di un guasto, di stimare quanto tempo manchi prima che un guasto si verifichi all'interno di un macchinario e di individuare le componenti che si stanno deteriorando. La CBM è stata largamente ed efficacemente applicata ai macchinari rotanti che, generalmente, basano il loro funzionamento sui cuscinetti. Il funzionamento continuo e affidabile dei cuscinetti è importante poiché il guasto di uno di essi può compromettere l'intero sistema. Quindi monitoraggio, prognosi e diagnosi di cuscinetti rappresentano task cruciali all'interno di programmi di manutenzione di tipo real-time.

In questa ricerca è stato effettuato uno studio completo delle tecniche di soft computing includendo la classificazione multi- e one-class e le strategie di combinazione basate su fusione e selezione di classificatori al fine di progettare e sviluppare metodologie accurate e robuste al rumore per la diagnosi e prognosi di guasti su cuscinetti a elementi rotanti.

Sono stati utilizzati segnali basati sulle vibrazioni registrati da quattro accelerometri su un dispositivo meccanico che includeva cuscinetti a elementi rotanti: i segnali sono stati registrati sia quando tutti i cuscinetti nel dispositivo erano "sani" sia quando uno di essi era stato sostituito con un cuscinetto danneggiato artificialmente. Sono stati considerati quattro tipi di guasti e tre livelli di gravità.

Questa ricerca ha portato al progetto e sviluppo di nuovi classificatori che, grazie agli alti livelli di accuratezza raggiunti, hanno dimostrato di rappresentare una valida alternativa ai classificatori tradizionali. Inoltre gli alti livelli di accuratezza e robustezza al rumore ottenuti dagli esperimenti provano l'efficacia delle metodologie proposte per effettuare automaticamente la prognosi e diagnosi delle componenti di macchinari rotanti all'interno di programmi di CBM.

ABSTRACT

Machines used in the industrial field may deteriorate with usage and age. Thus it is important to maintain them so as to avoid failure during actual operation which may be dangerous or even disastrous. The literature has focused its attention on the development of optimal maintenance strategies, such as condition-based maintenance (CBM), in order to improve system reliability, to avoid system failures, and to decrease maintenance costs. CBM aims to detect the early occurrence and seriousness of a fault, to estimate the time interval during which the equipment can still operate before failure, and to identify the components which are deteriorating. CBM has been widely and effectively applied to rotating machines, which usually operate by means of bearings. The reliable and continuous work of bearings is important as the break of one of them can compromise the work of the system. Thus the monitoring, prognosis and diagnosis of bearings represent crucial and important tasks to support real-time maintenance programs.

This research has carried out a complete analysis of advanced soft computing techniques ranging from the multi-class classification to one-class classification, and of combination strategies based on classifier fusion and selection. The purpose of this analysis was to design and develop high accurate and high robust methodologies to perform the detection, diagnosis and prognosis of defects on rolling elements bearings.

We used vibration signals recorded by four accelerometers on a mechanical device including rolling element bearings: the signals were collected both with all faultless bearings and after substituting one faultless bearing with an artificially damaged one. Four defects and three severity levels were considered.

This research has brought to the design and development of new classifiers which have proved to be very accurate and thus to represent a valuable alternative to the traditional classifiers. Besides, the high accuracy and the high robustness to noise, shown by the obtained results, prove the effectiveness of the proposed methodologies, which can be thus profitably used to perform automatic prognosis and diagnosis of rotating machinery components within real-time condition-based maintenance programs.

Contents

Introduction

One of the most important aspects in the domestic and industrial fields is the reliability of the used equipment such as rotating machines, which are increasingly employed in these two contexts. Furthermore the assessment of these machines is becoming more and more strict as they should meet more and more demanding performance criteria [1]. Unfortunately, these systems and machines may deteriorate with usage and age [135] bringing even to the breakdown of the system [21, 76, 141, 142].

Failures in these systems may result into catastrophic consequences depending on the field in which they operate. The most critical fields include, for example, the nuclear one where problems on machine may bring to serious consequences on people and the environment [21, 22, 142]. In the manufacturing field, breakdown problems can bring to unscheduled downtime causing reduction in the product quality, loss in the production and thus loss of money from the customers [150].

It is, therefore, extremely important to maintain and, if necessary, to repair systems and machines so as to avoid failure during the actual operation [143]. For this reason the *machine maintenance* has became an integral part of industrial systems with the aim of reducing costly machine downtime and ensuring production quality.

Maintenance is the set of activities aimed at maintaining a system in operable condition [5]. Although maintenance has been generally regarded as an unnecessary cost in many industries and organizations [129], in the last few years it has been considered more and more a profit-generating activity and, thus, a strategic issue [5, 129, 136]. Actually, a machine that is not properly maintained may result in speed losses, lack of precision, and reduction of the operating conditions up to the system breakdown [1, 4, 5]. Thus, the maintenance quality represents a key factor of the operating and environmental conditions that influence machine life [3]. Besides, only an effective maintenance policy

can improve machine performance and product quality [5].

For these reasons, researchers have increasingly focused their attention on the development of optimal maintenance strategies so as to improve system reliability, to avoid system failures, and to decrease maintenance costs themselves [143]. In particular, maintenance programs should guarantee that physical assets behave as expected at the minimum expense [129]. Thus, in the last years, the maintenance issue has been widely investigated in the literature [28, 90, 135, 143, 146, 151].

Maintenance activities can be classified in two main categories, namely *corrective maintenance* and *preventive maintenance*.

Corrective maintenance (CM), also called breakdown maintenance, refers to repairing faults after their occurrence. This kind of maintenance may imply high costs owing not only to the need of restoring the equipment operative condition, but also to possible production loss and/or safety consequences [129, 143]. All this is due to the fact that CM is not able to prevent any faults.

Preventive maintenance (PM), on the contrary, tries to prevent fault occurrence by identifying and correcting conditions that would cause breakdowns. When PM is performed through periodic inspections, it is called time-directed maintenance (TDM) [21, 129, 143, 154].

Although CM is usually considered more expensive than TDM, the indiscriminate use of overhaul or preventive replacement procedures in TDM programs may result in unnecessary waste of time and resources [129]. Furthermore no guarantee can be given regarding the proper work of the system between two subsequent checks.

To overcome the drawbacks of the above described maintenance processes another form of PM, called *condition-based maintenance* (CBM) [125, 129], has been widely adopted in the last years to detect the onset of a failure [21, 22, 23, 76, 141, 142]. CBM represents the ideal form of maintenance when a failure cannot be prevented, e.g., failures caused by random events. CBM is similar to TDM since it is performed at given intervals, however, since it is based on a continuous monitoring of the system, it needs to perform appropriate maintenance actions only when necessary. Thus, unlike TDM, CBM does not usually cause an intrusion into the equipment, and the actual preventive action is triggered by the detection of an incipient failure [129]. Therefore CBM aims to detect the early occurrence and seriousness of a fault, to es-

timate the time interval during which the equipment can still operate before failure, and to identify the components (e.g., bearings) that are deteriorating [141]. In particular, CBM exploits condition monitoring information, which consists of the continuous or periodic measurement and interpretation of data that help assessing the operating condition of each system component [141, 146, 147].

In this dissertation we focused on the condition-based maintenance of rotating machines [21, 141]. Actually, condition monitoring, fault diagnosis and fault prognosis of rotating machinery have become more and more important in many industrial fields from the safety-critical ones to the manufacturing and production ones [58, 61, 77], in order to guarantee the survival of the machines and the reliability of the involved processes. Since fault occurrence cannot be avoided in these machines, early detection and diagnosis of incipient failures can help prevent the machine breakdown by identifying the presence of symptoms such as increased vibrations, noise and temperature [146]. Thus CBM has been widely and effectively applied to rotating machines. Such machines usually operate by means of bearings, which represent a critical component whose state can be profitably used to represent the machine state [146]. The reliable and continuous work of all the bearings is very important as the break of one of them can compromise the proper work of the whole system [22, 23, 76, 141, 142]. However, bearings, even though well designed, often have to bear stress and heavy load which can deteriorate their performance [61] up to a point in which they can compromise the proper work of the machine inside which they are located, bringing to the system breakdown and possibly have catastrophic consequences depending on the field in which they operate [21, 76, 141, 142]. Besides, the faults occurring in rotating machines are often linked to bearing faults [77, 130]. For all these reasons the monitoring and the fault diagnosis of bearings have been widely studied in the literature, in order to make the fault detection as automatic as possible and thus to be able to implement real time maintenance [22, 23, 61, 114, 115, 141].

In this work some methodologies to perform the robust diagnosis and the prognosis of rolling element bearing defects are proposed, ranging from the multi-class classification to the one-class classification, paying attention to every "level" of a classification system. In particular, since there are four different approaches to build a classifier ensemble

[30, 73], namely, the *data level* where different data subsets are used, the *feature level* where different feature subsets are used, the *classifier level* where different base classifiers are used, and the *combination level* where different combiners are used [30, 73], the proposed methodologies make use of each of these approaches to create more robust and more accurate classification systems. Besides, some new developed one-class classifiers are introduced in this dissertation. The proposed classifiers have proved to be very accurate and to perform pretty better than the traditional multi-class and one-class classifiers, thus showing to be valuable alternatives to the traditional classifiers.

In this dissertation a real data set has been used. This data set has been provided by Avio Propulsione Aerospaziale (Rivalta di Torino, Italy).

This dissertation is organized as follows. Chapter 1 introduces an overview on the main elements of Pattern Recognition on which this thesis is based, Chap. 2 describes in details the issues dealt with in this dissertation as well as an overview on the state of the art on this field. Chapter 3 presents the designed and developed methodologies, algorithms and classifiers, Chap. 4 concerns the description of the used data set, while Chaps. 5–8 show the proposed methodologies and the related experimental results. In particular Chap. 5 refers to the classification and diagnosis of rolling element bearing faults, Chap. 6 deals with the prognosis issue of rolling element bearing faults, Chap. 7 considers the problem of creating algorithms robust to noise, and, Chap. 8 copes with the diagnosis of bearing faults as a one-class problem. Finally, Chap. 9 provides concluding remarks as well as a brief overview of the future work.

Chapter 1

Fundamentals of Pattern Recognition

We should make things
as simple as possible,
but not simpler.

- A. Einstein -

The problem of searching for patterns in data is fundamental in the real life [73]. *Pattern recognition* (PR) is about assigning *labels* (*classes*) to *objects* (*data*) [32, 73, 138]. Each object is described by a set of measurements (attributes) defined in a certain space (a one-dimensional space or more generally a multidimensional space). These attributes are called *features* [73, 138]. The space defined by the set of features is called *feature space* [73].

There are two major types of pattern recognition branches: *unsupervised* and *supervised* [73].

In the *unsupervised* category (known also as *unsupervised learning*), the goal is to unravel the underlying similarities, and to discover the structure of the data set if there is any [32, 73, 152]. This usually means that the aim of the learning process is to discover whether there are clusters in the data, and what characteristics make similar the objects inside the same cluster as well as different from the objects belonging to the other clusters [152]. In the literature, many clustering algorithms have been and continue to be proposed for unsupervised

learning [73]. The choice of an algorithm is basically guided by the designer's preference, since different algorithms, even though starting from the same set of data, may lead to different results [10, 73]. The issue in this branch of PR is that there is no way to evaluate the results but the indication of the user, which means that the result interpretation may also be based on subjective estimations [73]. One of the most used unsupervised learning algorithm is the *k-means* algorithm [10, 73], whose steps are described in Fig. 1.1. Fig. 1.3 shows an application of the k-means algorithm to the data set represented in Fig. 1.2. Besides, inside the unsupervised PR category the one-class classification, which will be fully described in Chap. 3, is also included.

1. Choose the number of clusters k to make and a similarity measure $\mathbb{S}(a, b)$ between two objects a and b. Initialize the k cluster centers (e.g., by randomly selecting k points from the data set D to be the centers). Go to step 2.
2. Label all points in D so that each point is assigned to the cluster with the most similar center. Go to step 3.
3. Calculate the new centre of each cluster as the mean of the points from D assigned to the specific cluster. Go to step 4.
4. Repeat step 2 until no change in the centers occurs.

Figure 1.1 *k-means algorithm steps.*

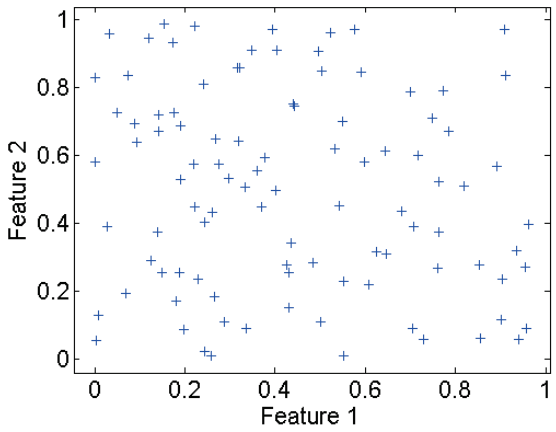
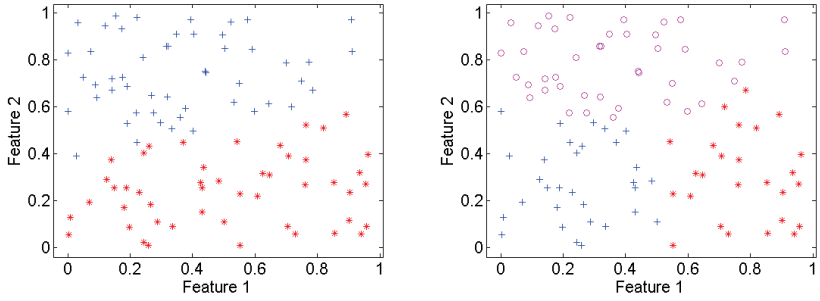
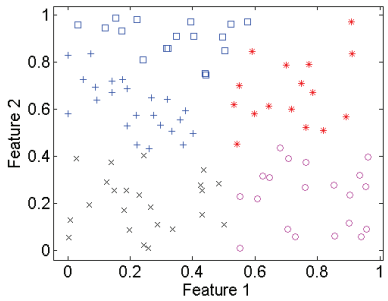


Figure 1.2 *Data set.*

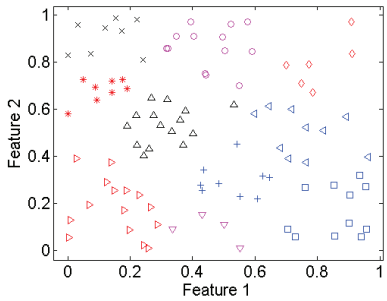


(a)k-means with k=2

(b)k-means with k=3



(c)k-means with k=5



(d)k-means with k=10

Figure 1.3 *Unsupervised learning. Application of the k-means algorithm to the data set represented in Fig. 1.2. (a)* $k = 2$*, (b)* $k = 3$*, (c)* $k = 5$*, and (d)* $k = 10$*. Each cluster is identified by a specific symbol, so that samples belonging to the same cluster are represented by the same symbol while samples from different clusters are represented by different symbols.*

The *supervised category* [13, 73] (known also as *supervised learning*) differs from the unsupervised learning since a priori known knowledge is available. More precisely each object in the data set is associated to a preassigned *label*. Each label identifies a class, so that each class is identified by a different label. Intuitively, a class contains similar objects (all associated to the same label), whereas objects from different classes are dissimilar [13, 73]. In most cases the labeling process cannot be described in an algorithmic form, this is why we generally supply the classification system with learning skills [73]. More precisely we provide the classification system with a set of labels in order to allow the system to learn how to distinguish objects belonging to different classes. This process is called *learning* or *training* process. What is really important in the learning process is to obtain a classification model with good generalization capabilities, i.e., a model that accurately predicts the class labels not only of the objects seen during the learning process but also of unknown objects [122].

Fig. 1.4 shows an example of supervised learning dealing with a 2-class classification problem. The classification is performed by four classifiers: a 3-NN, a Linear Discriminant Classifier, a Quadratic Discriminant Classifier, and a Multi-Layer Perceptron Neural Network.

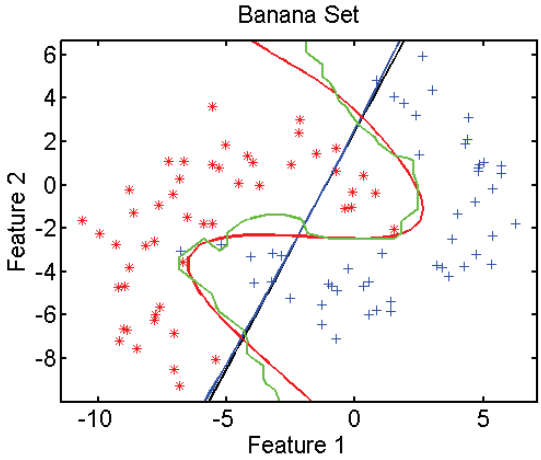


Figure 1.4 *Supervised learning. 2-class classification problem. First class (red stars), second class (blue plus). Classification performed by four classifiers: 3-NN (green line), Linear Discriminant Classifier (black line), Quadratic Discriminant Classifier (blue line), Multi-Layer Perceptron Neural Network (red line).*

1.1 Pattern Recognition concepts for supervised learning

1.1.1 Features

As stated above, objects are described by attributes called *features*. According to "The International Dictionary of Artificial Intelligence" [106] a feature can be defined as *a (usually) named quantity that can take on different values*. These values are the feature's domain and, in general, can be either quantitative or qualitative [41, 73].

The branch of PR which operates with *quantitative features* is called *statistical pattern recognition* [73]. In this branch the features are represented by numbers, such as integers or real numbers, for example the amplitude of a signal measured in dB. In particular numerical features values are arranged as an n-dimensional vector as represented in the following:

$$X = [x_1, x_2, \ldots, x_n] \qquad x \in \mathbb{R}^n$$

where each element of the vector corresponds to a specific feature, while the whole vector represents an object of the data set. The real space $\mathbb{R}^n$ is called the *feature space* and each axis corresponds to a specific feature.

The choice of a good set of features is a basic point to obtain good performance in the pattern recognition process [23, 76].

In this dissertation we make use of quantitative features and we will explain how we manage to obtain these features to be later used in the classification process.

1.1.2 Classes

As stated above, a *class* should contain similar objects while objects from different classes should be dissimilar [73]. However, the concept of similarity and dissimilarity are not always perfectly clear, sometimes classes are well defined and, in the simplest case, the classes are mutually exclusive [73], such as in the handwriting recognition where the classification system receives and interprets intelligible handwritten input from a certain source (paper documents, photographs, or other devices). In fact each hand-written symbol corresponds to one and

only one symbol stored in the computer, no matter if we are able to recognize the right matching symbol [73].

Nevertheless, in other classification problems, a distinct separation among the classes is not always simply identifiable. For example in the medical research there is an intrinsic variability that makes difficult to identify the classes, as well as to identify which are the most discriminant features [73]. Furthermore, there can be co-presence of more than one illness, which makes even more difficult to identify the specific illness we were interested in [73, 140].

1.1.3 Data sets

A data set is a collection of data (objects, elements, samples), usually presented in a matrix-like form, where each column represents a particular feature, while each row (feature vector) corresponds to a given sample of the data set. Thus a data set can be represented by an $N \times n$ matrix where N is the number of rows which corresponds to the number of objects composing the data set, and n is the number of features describing each object of the data set.

	First feature	Second feature	...	*n*-th feature
First object				
Second object				
...				
N-th object				

Figure 1.5 *Representation of a data set with N objects described by n features as an $N \times n$ matrix.*

A data set is described by several parameters which include the following ones:

- the number and types of the features,
- the number of samples,
- the number of classes,
- the vector of the class labels associated to each object.

Normally the order in which the samples are "listed" does not matter and thus the list of objects is unordered. Of course there are cases where the order is important such as in regression problems [19].

Data sets can be obtained in many ways. Besides there exist some data sets made available on Internet which can be used as benchmarks in the PR field. One of these repositories of data sets is the UCI Machine Learning Repository Database [40] at http://archive.ics.uci.edu/ml.

In this dissertation a real data set is used. This data set has been provided by Avio Propulsione Aerospaziale, via I Maggio, 99, Rivalta di Torino, Italy.

1.1.4 Classifiers

A classifier can be described by any function F:

$$F : \mathbb{R}^n \rightarrow \Omega$$

which, starting from the n-size vector of features values describing an object x ($x \in \mathbb{R}^n$), identifies the class ($\omega_i \in \Omega$) to which x belongs. A classifier can be considered as a set of *discriminant functions* F [117], each yielding a score (probability) for one specific class (thus one function f_i per each class ω_i). Each discriminant function f_i returns a value when applied to an object. More precisely each function f_i returns a value which specifies the confidence of the function in assigning the specific object x to the class ω_i [73]. Then, typically, the object x is assigned to the class with the highest score. Ties are broken randomly. Thus the classifier is the result of the application of the maximum rule to this set of discriminant functions (see Fig. 1.6). Therefore, generally, a classifier performs a mapping from an n-dimensional space $\mathbb{R}^n$ to a c-dimensional space, $\mathbb{R}^c$, where c is the number of classes [32, 73]. An example of classification performed by a Quadratic Discriminant Classifier (QDC) is represented in Fig. 1.7 [73].

There exist many types of classifiers such as linear, quadratic, and neural networks classifiers. Besides, more classifiers can be appropriately combined. We will deal with the combination of classifiers in the next chapters.

1.2 Classification process

1.2.1 Classification process steps

To perform a classification process, first of all we need data, that means that we need to make measurements of what we are interested in. This

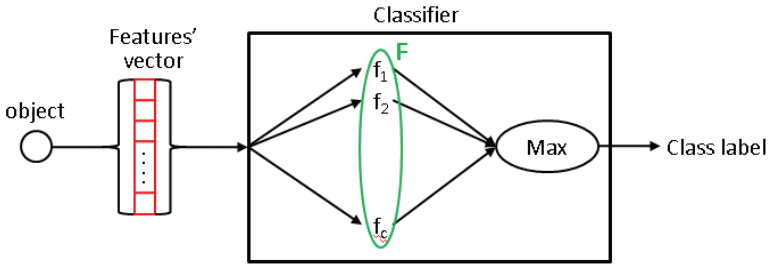


Figure 1.6 *Classification process. The object is described by the features' vector. The classifier has one discriminant function associated to each class. The maximum rule determines the winner class, i.e., the class to which the object will be assigned.*

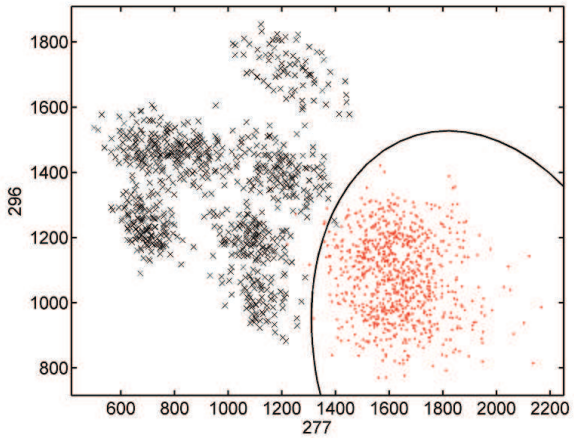


Figure 1.7 *2-class data set. First class: crosses, second class: points. Classification performed by a QDC classifier.*

can be done using sensors which are a fundamental part of every classification system.

Then the raw data, represented in a certain way depending on the type of the used sensors, should be processed in order to represent them in a more suitable way. For example transforming data from time to frequency domain by using the *Fast Fourier Transform* (FFT) [21, 22, 23, 76, 141, 142].

Once the data are preprocessed, we need to prepare the training set, i.e., to select/extract the objects to be used as training set. This process is called *prototype selection/extraction* depending on the type of algorithm used to perform this process [73].

Then another process to eliminate useless information and retain the most important information as much as possible should be performed. More precisely, data are represented by a set of features (for example each element of the FFT), and, since features are not all equally relevant, we need to identify which of these features are really useful at the aim of the classification. The process which selects/extracts the best features to represent the data with classification purpose is called *feature selection/extraction*. This process should be able to represent the data in the best way, which means to retain the most information (the best features) while removing unnecessary noise (useless features) [21, 22, 23, 73, 76, 141, 142].

Once the data have been "cleaned", they can be used to train a classification system, which will be successively used to classify "unknown" data (test set). The classification system may be composed of only one classifier or more classifiers appropriately combined. The combination level represents another crucial step since the choice of the combination strategy can affect all the classification process. Besides each classifier should be trained appropriately and the algorithm used to train the classifier represents another parameter which can further affects the performance of the final classification system.

Data sampling, prototype selection/extraction (choice of the algorithm to be used and its related parameters), feature selection/extraction (choice of the algorithm to be used and its related parameters), training (choice of the classifier to be used, choice of the classifier training algorithm and, if needed, choice of the combination strategy,...) compose the core of the supervised PR process.

The classification system can then be further tuned working on each of these steps but not necessarily all of them [73]. When a satisfactory accuracy on the training set has been reached, the training process can be stopped and the system can be used to process and classify new data [73]. Each of these subprocesses is fundamental to create the whole classification system. In the next chapters we will deal with all these subprocesses more in detail.

1.2.2 Classification performance indexes

Every time we train a classifier (classification system) we would like to know how good it is, i.e., to evaluate its performance, since we are interested in classifiers or better classification systems which are able

to reach reasonable performance.

The performance of a classifier can be described by many *performance indexes* such as *classification accuracy*, *confusion matrix*, and others. Of course we have to identify which of these indexes are the most important for the specific problem we are dealing with. Probably the most important and used performance index is the accuracy.

Classification accuracy

Generally the *classification accuracy* (shortly *accuracy*) is defined as the number of samples correctly classified divided by the total number of samples under classification as shown in eq. 1.1:

$$Accuracy = \frac{\text{number of samples correctly classified}}{\text{total number of classified samples}} \tag{1.1}$$

The *classification error* (shortly *error*) is then obtained subtracting the classification accuracy from 1 as described in eq. 1.2:

$$Error = 1 - Accuracy = \frac{\text{number of samples incorrectly classified}}{\text{total number of classified samples}} \tag{1.2}$$

True positives, False positives, True negatives, False negatives

There are other indexes related to the accuracy which are specific for 2-class data sets. When a 2-class data set is considered, generally, one class is called *positive class* (PC) while the other one *negative class* (NC). If we call

- class 1: Negative Class, *NC* (composed by the so-called *negative samples*),
- class 2: Positive Class, *PC* (composed by the so-called *positive samples*),

then, we can define the following four indexes:

- *TN*: *True Negatives*, i.e., the number of negative samples correctly classified,
- *FP*: *False Positives*, i.e., the number of negative samples incorrectly classified as belonging to the positive class,

- *TP*: *True Positives*, i.e., the number of positive samples correctly classified,
- *FN*: *False Negatives*, i.e., the number of positive samples incorrectly classified as belonging to the negative class.

Confusion matrix

The four indexes *TP*, *TN*, *FP*, *FN* can be represented in a *confusion matrix*. A confusion matrix [70] provides an easy, synthetic, and complete way to describe the knowledge about a classification system performance. It gives information about the *true labels* and the *estimated labels* assigned by a classification system to a specific data set. An example of a confusion matrix for a 2-class problem is represented in Fig. 1.8. Of course, a confusion matrix can be generalized to an N-class problem [138].

		Estimated Labels	
		NC	*PC*
True Labels	*NC*	*TN*	*FP*
	PC	*FN*	*TP*

Figure 1.8 *2-class confusion matrix.*

In a 2-class problem, starting from the confusion matrix (considering the two classes having the same priority) the accuracy percentage can be evaluated as in eq. 1.3

$$accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{1.3}$$

consequently the classification error is expressed by eq. 1.4:

$$error = 1 - accuracy = \frac{FN + FP}{TN + TP + FN + FP} \tag{1.4}$$

Specificity and Sensitivity

Using *TP*, *FP*, *TN*, and *FN* we can derive other two performance indexes called *sensitivity* and *specificity*.

In particular, *sensitivity* measures the proportion of actual positives which are correctly identified and is expressed by eq. 1.5.

$$sensitivity = \frac{TP}{TP + FN} \tag{1.5}$$

while the *specificity* index measures the proportion of actual negatives which are correctly identified and is expressed by eq. 1.6.

$$specificity = \frac{TN}{TN + FP} \tag{1.6}$$

The theoretical optimal prediction is to achieve 100% sensitivity and 100% specificity. However, for any test, there is usually a trade-off between these two measures.

ROC curve

Finally another metric used to measure the performance of a classifier is the Receiver Operating Characteristic curve (ROC curve) [37].

The ROC curve is a graphical plot of the *sensitivity* vs "1 - *specificity*", for a binary classifier system as its discrimination threshold is varied.

The ROC curve is a tool which allows to select possibly optimal models while discarding suboptimal ones independently from the cost context or the class distribution.

Fig. 1.9 shows some ROC curves related to different classifiers, each of which is represented by a different color.

1.2.3 Methods to test the classification system

If we were able to try the classifier on all possible input objects, we would know exactly how accurate the classifier is [73]. Unfortunately, this is hardly possible, since only a subset (generally a small subset) of all the possible inputs is available. Thus only an estimated accuracy can be evaluated [73]. When a classification process is performed the data are generally divided into two subsets, one called *training set* which is used during the training process to train the classifier and a *test set* which is used to evaluate the performance of the classifier on unseen data. There are a lot of different procedures to evaluate the performance of a classifier.

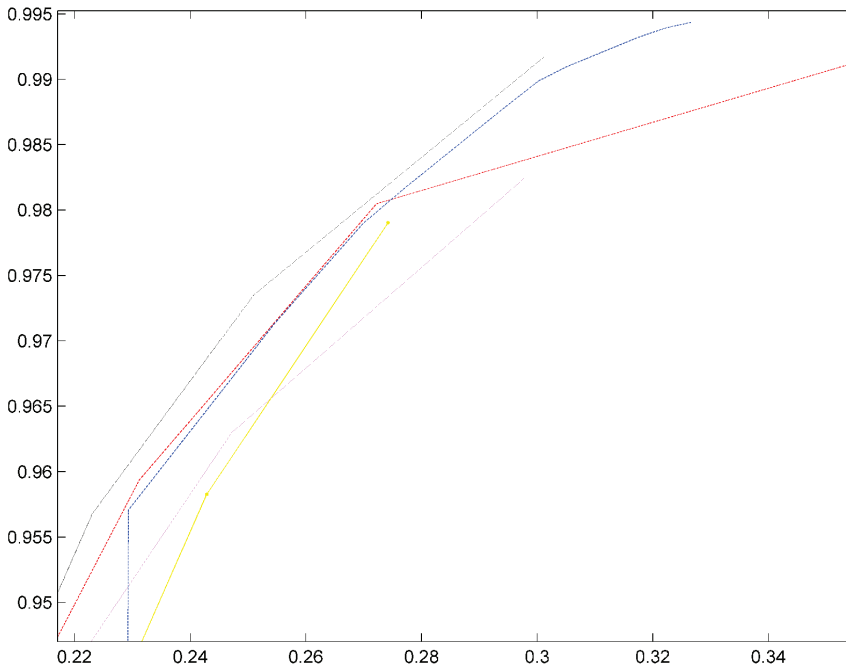


Figure 1.9 *ROC curves related to different classifiers, each of which is represented by a different color.*

Resubstitution error method

Denoting with D the whole data set, the simplest procedure to evaluate the performance of a classifier is based on the evaluation of the *resubstitution error*, i.e., the error on the training set. This procedure is called *resubstitution error method* (shortly R-method) and consists in training the classifier C using the whole data set D and then test the classifier again on D [73]. Thus the training and the test set coincide with D (Fig. 1.10).

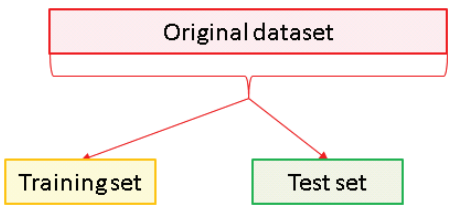


Figure 1.10 *R-method to create training and test sets. The training set coincides with the test set and both coincide with the whole original data set.*

Since training an algorithm and evaluating its performance on the same data may lead to an overoptimistic result [7, 75, 73], other testing methods were introduced such as the *cross-validation* (CV) algorithm. A general description of the CV strategy has been given by Geisser [43]. The basic idea is that a good estimate of the performance would be obtained using new data to test the classifier, i.e., data different from the training data [7, 43, 88]. The main idea behind CV is to split the data, once or several times, for estimating the performance of a specific model. Once split, a part of the samples is used to train the model, while the remaining samples are used to test the model and thus to estimate its performance. Compared to the R-method, CV avoids the overoptimistic result problem since the training samples are independent from the test samples. The major interest of CV lies in the universality of the data splitting heuristics. It only assumes that data are identically distributed, and training and test samples are independent, which can even be relaxed [7, 73]. This makes the CV algorithm suitable for many applications in many frameworks, such as regression [43, 118], density estimation [112, 118], and classification [8, 29] among many others. However, some CV procedures have been proved to fail for some model selection problems, depending on the goal of model selection, namely, estimation or identification [7, 73]. The issue of how to organize the training and test sets has been around for a long time [73, 131]. Hereafter we summarize some of the most used CV procedures.

Hold-out method

One of the simplest CV procedures is called Hold-out (shortly H-method) [29]. This method relies on a single split of the original data. Part of data (training set) is used for training the algorithm, and the remaining data (test set) are used to evaluate the performance of the algorithm [73]. However no study exists to define how to split the data, i.e., which is the optimal percentage of samples of the original data which should be used to create the training and the test sets (Fig. 1.11). However, generally, the training set is bigger than the test set [7, 73].

In most real applications, only a limited amount of data is available, thus the single split can result in either a training or a test set too small to be significant [73]. This leads to the idea of splitting the data more than once. The idea is that a single data split yields a validation

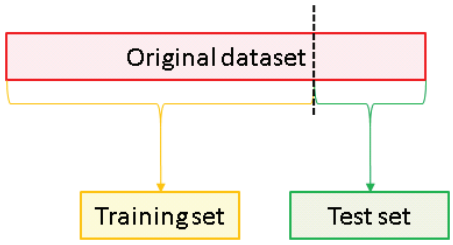


Figure 1.11 *H-method to create training and test sets. The original data set is split once. One part forms the training set and the other one the test set.*

estimate of the performance, then averaging over several splits yields a cross-validation estimate [7]. For example considering the H-method, we can average the performance results related to several experiments based on the H-method, where each experiment corresponds to a different data split [7].

K-fold cross-validation

K-fold cross-validation is one of the most used test procedure. The algorithm consists in choosing an integer K (preferably a factor of N, where N is the number of samples in the original data set D) and randomly divide D into K subsets of size $\frac{N}{K}$. Of the K subsets, one is retained as test set, while the others are used as training set. This procedure is repeated K times choosing, each time, a different subset as test set. At the end the whole accuracy is evaluated as the average of the K estimated accuracies.

The question of how to select K is still open, even though indications can be given towards an appropriate choice [73]. When the goal of model selection is estimation, it is often reported that the optimal K is between 5 and 10 [7, 49, 73]. Fig. 1.12 provides an example of K-fold CV with $K = 3$.

Leave-one-out

Leave-one-out [2, 43, 73, 118] is the most classical exhaustive CV procedure [7].

It consists in a K-fold cross-validation with $K = N$ where N represents the number of objects in the original data set D [73].

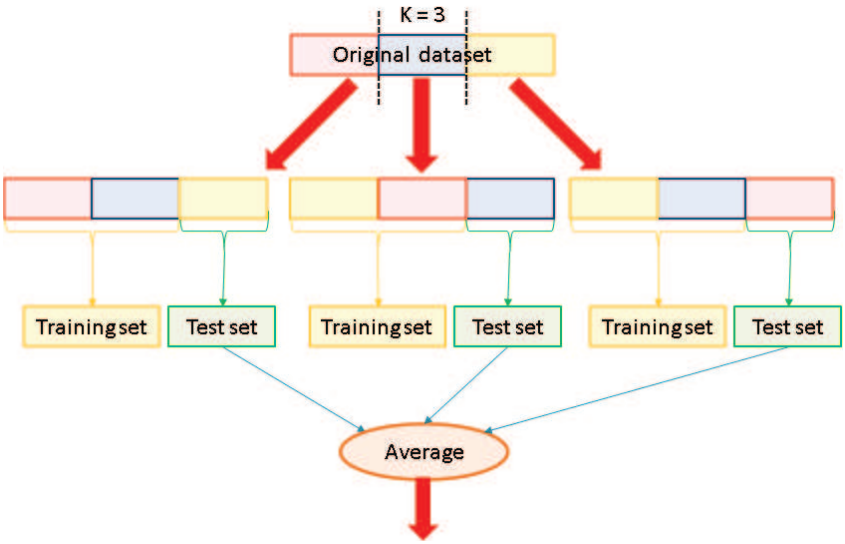


Figure 1.12 *K-cross validation method to create training and test sets (example with $K = 3$). The original data set D is randomly divided into 3 subsets of size $\frac{N}{3}$, with N the number of objects of the data set. Of the 3 subsets, one is retained as test set, while the others are used as training set. The procedure is repeated 3 times choosing, each time, a different subset as test set. The whole accuracy is evaluated as the average of the 3 estimated accuracies.*

Dietterich's 5x2cross validation

The Dietterich's 5x2cross validation method [30] consists in 5 repetitions of a 2-fold cross validation. This gives 10 pairs of error (accuracy) estimates which can be successively averaged to compute the final error (accuracy) [73].

1.2.4 Validation set

It is now becoming a common practice to divide the original data set into three sets instead of two [13, 73]:

- training set,
- validation set,
- test set.

As before, the test set remains unseen during the training process, while the validation data set acts as *pseudo-testing*.

Considering for example the training of a neural network classifier, the use of the validation set means to continue the training process until the performance improvement on the training set is no longer matched by a performance improvement on the validation set [13, 73]. At this point the training should be stopped so as to avoid overtraining. Fig. 1.13 represents the typical "evolution" of the training and validation errors of a neural network with the increase of the number of epochs in which the network is trained.

However, not all the data sets are large enough to allow for a validation set to be cut out [73].

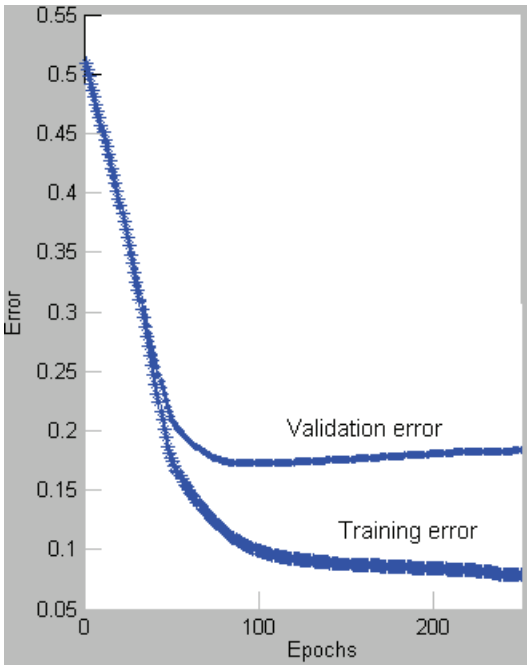


Figure 1.13 *The training process aims to reduce the resubstitution error. When a validation set is used, the training can be stopped when the performance improvement on the training set is no longer matched by a performance improvement on the validation set. The line composed by crosses represents the error on the training set while the line composed by points represents the error on the validation set. In the figure, at a certain point, while the error on the training set is still decreasing, the error on the validation set starts to increase.*

Chapter 2

The rolling element bearing diagnostic and prognostic issues

In the middle of difficulty
lies opportunity.

- A. Einstein -

One of the most important aspects in the domestic and industrial fields is the reliability of the used equipment such as rotating machines which are increasingly used inside these two fields. Furthermore the assessment of these machines is becoming more and more strict as they should meet increasingly demanding performance criteria [1].

Unfortunately, these systems and machines may deteriorate with usage and age [135] bringing even to the breakdown of the system [21, 76, 141, 142]. For these reasons, the maintenance of these systems represents a crucial process and thus the *machine maintenance* has became an integral part of industrial systems with the aim of reducing costly machine downtime and ensuring production quality.

In the last years, the maintenance issue has been widely investigated in the literature [28, 90, 135, 143, 146, 151] and in particular the condition-based maintenance (CBM) [125, 129] has been introduced and widely adopted to detect the onset of a failure [21, 22, 23, 76, 141, 142]. CBM represents the ideal form of maintenance when a failure cannot be prevented, e.g., failures caused by random events. Besides, CBM does not usually cause an intrusion into the equipment, and the actual

preventive action is triggered by the detection of an incipient failure [129]. CBM aims to detect the early occurrence and seriousness of a fault, to estimate the time interval during which the equipment can still operate before failure, and to identify the components (e.g., bearings) that are deteriorating [141].

In particular, CBM exploits condition monitoring information, which consists of the continuous or periodic measurement and interpretation of data that help assess the operating condition of each system component [141, 146, 147].

In this dissertation we focus on the CBM process of rotating machines [21, 141]. Actually, condition monitoring and fault diagnostics of rotating machinery have become more and more important in many industrial fields from the safety-critical ones to the manufacturing and production ones [58, 61, 77], in order to guarantee the survival of these machines and the reliability of the involved processes. Since fault occurrence cannot be avoided in the machine, early detection and diagnosis of incipient failures can help prevent the machine breakdown by identifying the presence of symptoms such as increased vibrations, noise and temperature [146]. Thus CBM has been widely and effectively applied to rotating machines [22, 61, 114, 141]. Such machines usually operate by means of bearings, which represent a critical component of the machines themselves so that their state can be profitably used to represent the machine state [146]. The reliable and continuous work of all the bearings is very important as the break of one of them can compromise the proper work of the whole system [22, 23, 76, 141, 142].

However, bearings, even though well designed, often have to bear stress and heavy load which can deteriorate their performance [61] up to a point in which they can compromise the proper work of the machine inside which they are located, bringing to the system breakdown and possibly have catastrophic consequences depending on the field in which they operate [21, 76, 141, 142]. Besides, the faults occurring in rotating machines are often linked to bearing faults [77, 130]. For all these reasons the monitoring and the fault diagnosis of bearings have been widely studied in the literature, in order to make the fault detection as automatic as possible with the purpose of implementing real time maintenance programs [22, 23, 61, 114, 115, 141].

While machine diagnosis comprises the automated detection and

classification of faults, machine prognosis concerns the automated estimation of how soon and likely a failure will occur, including the forecast of the remaining operational life, future state, or probability of reliable operation of an equipment based on the collected condition monitoring data [53, 128]. Prognostics is vital in order to significantly reduce expensive downtime, maintenance costs and safety hazard conditions [53]. However, prognostics is a very wide and new research area which has been little studied [53, 116] compared to the diagnosis issue.

2.1 State of the art

Several methods for the detection and diagnosis of faults in bearings have been developed in the last years [21, 22, 23, 76, 91, 114, 115, 141, 142]. These studies mainly focus on three issues, namely,

1 the choice of the data (i.e., signals) to be collected,

2 the feature selection/extraction algorithm to be adopted to appropriately decrease the dimensionality of the signal representation space,

3 the classification system used to distinguish between signals corresponding to faultless bearings and signals associated with faulty bearings, possibly identifying also the specific types and severity levels of defects.

The aim of the first issue is to find the signals that convey useful information regarding early indications of changes in the bearing state [4, 5]. As far as this issue is concerned, the literature presents studies such as vibration analysis, which uses vibration signals emitted by the mechanical system to trace the state of the system itself [120], and infrared thermography, which measures emissions of infrared energy in order to determine the operating condition of the system [125]. Vibration analysis is, however, the most used methodology, probably because it provides the most information from the collected data [58]. Besides, bearings are the best point in the machine where we can measure machine vibrations since they are the place where the basic dynamic loads and forces are applied [143]. Besides, as stated in [80], states vibration analysis is widely used in the bearing diagnosis field for the following reasons [78, 80]:

- vibration analysis can be performed during the normal operating

condition of the machine giving:

  - continuous information on the machine state,
  - continuous information on the machine bearings,

- incipient faults are quickly detected,
- it is economically justified,
- the equipment to collect vibration signals, i.e., mainly sensors, is relatively small and portable and thus not intrusive,
- it is generally accurate and reliable.

The three most commonly applied vibration techniques for bearing performance analysis [93] include time domain analysis [114, 115, 143], frequency domain analysis [21, 22, 23, 61, 76, 141, 142], and time-frequency domain analysis [91, 125].

Time domain analysis is usually based on performance indexes such as *Root Mean Square* (RMS), *Kurtosis*, and *Crest Factor* [56, 125], while frequency domain analysis is generally based on the *Fast Fourier Transform* (FFT) technique [24]. A mixture of the two is adopted in the time-frequency domain allowing for the analysis of signals which are transient or non-stationary. Time-frequency domain methods include *Short Time Fourier Transform* [68], the *Wigner-Ville Distribution* [79] and the *Wavelet Transform* [158].

The main advantage of frequency domain analysis over time domain analysis is its ability to easily identify and isolate certain frequency components of interest, such as the theoretical characteristic frequencies of the defects [20, 61]. Besides indexes such as *Crest factor* and *Kurtosis* start to increase as the spikiness of the vibration increases, but, as the damage increases, the vibration signal becomes more random and the values for the *Crest factor* and *Kurtosis* reduce to levels that are typical of normal bearings. Thus, the statistical analysis approach based on *Kurtosis* and *Crest factor* fails in detecting bearing defects at later stages of their development [92].

The second issue on which the literature has focused its attention is the choice of the most appropriate set of numeric characteristics, or features, used to represent the signals. For good classification, feature selection/extraction is a crucial step as the features represent the condition indicators in the classification process. Besides, not all features are meaningful and provide significant information about the machine

condition. Some of them may be useless or irrelevant [9, 146]. In particular feature selection/extraction aims to find the set of features with the highest discriminant power for classification purposes. Several methods have been proposed including Forward Feature Selection (FFS) [21, 22, 23, 76, 141, 142], genetic algorithms [57, 114, 115, 146], decision trees [120, 146], and Principal Component Analysis (PCA) [56, 121].

As regards the third issue related to the choice of the classification system, several methods have been adopted including statistical classifiers [21, 22, 23, 141], neural networks (e.g., Multi-Layer Perceptron neural networks (MLPs) or Radial Basis Function neural networks (RBFs)) [21, 22, 23, 76, 141], Support Vector Machines (SVMs) [109, 110, 157, 159], one class classifiers (e.g., convex hulls) [142], fuzzy-logic-based classifiers [72, 119], and decision tree (such as C4.5) [56, 119].

Besides classifiers can be appropriately combined using mainly two approaches which either select a classifier in a classifier ensemble (classifiers selection strategies) or appropriately combine all the classifiers in the ensemble (classifiers fusion strategies) [22].

As far as the prognostic issue is concerned, even though the literature on this area is continually growing [53], it still presents fewer works [53] compared to the diagnostic issue. Some works and reviews to help clarify this issue can be found in [15, 46, 53, 61, 116, 134, 153], where prognosis has been considered in several different fields and different types of machines, such as paper making machines [15], aircraft engines [153], rotating machines [53, 141].

Of course, prognostics can be considered superior to diagnostics in the sense that prognostics can prevent faults, and if impossible, be ready for the problems in terms of human resources and spare parts, and thus save extra unscheduled maintenance cost [61]. However, prognostics cannot completely replace diagnostics since in the reality there are always some types of fault which are not predictable [61]. In addition, prognostics, like any other prediction techniques, cannot be 100% trusted [61]. In the case of unsuccessful prediction, diagnostics represents a useful complementary tool to provide maintenance decision support [61]. Besides, diagnostics can be helpful to improve prognostics, i.e., the diagnostic information can be useful for preparing more accurate event data and hence building better CBM models for prognostics [61].

In the literature the methods for predicting rotating machinery failures can be grouped into the following three main categories [53]:

1 traditional reliability approaches,
2 prognostics approaches,
3 integrated approaches.

In particular, the first category, *traditional reliability approaches*, includes the event data based prediction approaches, in the second category, *prognostics approaches*, we can find the condition data based prediction methods, while the third category, *integrated approaches*, includes the prediction based on both event and condition data [53].

2.2 Discussion on the state of the art

Most diagnostic methods found in the literature consider the bearing fault diagnosis as a 2-class problem, since they just distinguish between faultless and damaged bearings independently of the type and/or the severity of the defect. However the knowledge of the type and severity level of the defect can bring valuable information to perform better and defect-oriented maintenance programs. This is why in this dissertation we propose classification techniques which are defect-oriented, i.e., able to recognize both the type and the severity level of the defect.

While the diagnostic problem represents a crucial issue in the maintenance process, also the prognostic problem represents a very important task. Actually, even though the diagnostic expert engineers have significant information and experience about machine failure and health states by continuously monitoring and analyzing the machine condition, in the literature, little attention has been paid to the study of the evolution of a defect, that is how a defect evolves over time if a fault component (such as a bearing) is not repaired or substituted by a faultless one [66, 123, 141]. An effective prognostics program gives the maintenance engineers more time to schedule a maintenance activity to repair and to acquire replacement components before the system further decreases its "health" state [66, 141]. This is the reason why we propose the study of the evolution of a defect as time passes in order to identify how the vibration signals change. This analysis can be profitably used to define prognostic program to detect as soon as possible any incipient defects, as well as to determine the time within which the maintenance, i.e., the substitution of the faulty bearing, should be performed before the

defect gets too serious.

Furthermore, many methodologies proposed in the literature are not tested on noisy data, so that several diagnostic techniques can perform well in a noise-free environment but very poorly in the presence of noise like in a real environment. For this reason, not only we test our classification techniques on noisy data in order to analyze their robustness to noise, but we also propose some techniques to increase the robustness to noise of the classification systems.

Finally, we also present a one-class classification study on the rolling element bearing diagnosis issue. In fact, in the literature, little attention has been dedicated to the main challenging problem regarding the diagnosis of mechanical equipment: the lack of a significant set of damaged data. In general, bearings can develop several types of faults that can be further divided in classes according to the level of severity. It is difficult to collect all types of faults and severities and, then, use them to train a classification system, since it is not possible to identify all the possible types of faults and severity levels, for example an indentation on the roll can be positioned in different parts of a roll and thus it may produce different (vibration) signals. Furthermore, as shown in [141], (vibration) signals can change as time passes making thinner and fuzzier the division of faulty bearings into categories. Thus it is impossible to collect and "catalog" an infinite number of faults and severities. Moreover it is often difficult or even impossible to collect data from faulty bearings as it requires to put damaged bearings into the rotating machine causing unwanted consequences. For all these reasons the creation of a training set for the damaged samples can be either expensive or impractical and thus difficult to achieve. On the other hand, it is relatively cheap and simple to obtain measurements from faultless bearings and thus from a normally functioning machine.

However, most of the techniques presented in the literature deal with the bearing fault diagnosis as a two-class problem involving only data associated to pre-specified faults and severities regardless of all the "possible" types of fault and levels of severity they were not able to collect. Unfortunately, in real and practical cases, it is quite unlikely that a trained classification system will have to cope with only the known faults and severities (i.e., faults and severities used to train the classification system). On the contrary, it is very common that the

classification system will have to cope with unknown faults and severities (i.e., faults and severities not used during the training process). Thus, even though some techniques can achieve very high accuracies on known faults/severities, they may perform very poorly on unknown faults/severities as shown in [142]. This is why, in contrast with most of the approaches proposed in the literature, we decided to develop a classification system which is able to generalize the problem and thus correctly classifies also damaged samples belonging to types of fault and levels of severity not used during the training process. For this reason we have decided to approach this classification problem as a one-class classification problem. This way, we achieve independence from the specific damaged samples we were able to collect to train our classification system and, thus, we are able to develop a more general classifier that can reach a good accuracy not only for known but also for unknown faults/severities.

Finally, as far as the prognostic issue is concerned, since prognosis involves projecting into the future and that the future cannot be determined with absolute certainty, assumptions and simplifications are generally inevitable when prognostics models are designed [53]. However, these assumptions should be carefully checked [53].

While the study of the bearing prognosis has been concentrated in finding out how to forecast the state of the machine basing the study on more or fewer assumptions, little attention has been given to analyze how faults can develop through the time, that is how a defect evolves over time if the damaged bearing is not substituted by a faultless one. This issue is extremely important since the presence of a fault does not necessarily means that the machine is not able to continue to work, at least for another amount of time, thus is can be useful to start from a condition in which we already know that the bearing is damaged and then use this information to forecast within which amount of time the bearing must be changed in order to avoid the machine downtime.

Thus, in this dissertation we will aim to find out how the vibration signals coming from a faulty bearing evolve if the bearing itself is not substituted immediately. This means, e.g., to study if the severity level of the bearing increases and, possibly, after how much time. Besides, we wish to analyze if, after a certain amount of time in which a damaged bearing continues to work, we can consider it equivalent to a bearing

with the same defect but with a higher level of severity.

Thus in this dissertation we present classification methodologies to perform, both in not-noisy and noisy environments, the diagnosis and prognosis of faulty bearings both when a lot of damaged data and when none or few damaged data have been collected.

Chapter 3

Pattern recognition theory

Find a bug in a program, and fix it,
and the program will work today.
Show the program how to find and fix a bug,
and the program will work forever.

- O.G. Selfridge -

3.1 Feature selection and feature extraction algorithms

Generally, when the data set presents a large number of features we should check if all these features are really useful at the aim of classification or if some features are redundant and/or irrelevant. This, of course, may hold for data sets characterized by a small number of features [73]. If the data set is characterized by "useless" features it may be better to perform a transformation to reduce the number of features describing the data and thus to reduce the feature space dimension. Transforming the input features set into another one can be done mainly into two ways: *feature selection* and *feature extraction*. Feature selection/extraction has been the focus of interest for quite some time and much work has been done [27].

Feature selection [27, 67, 82] is the technique of selecting a subset of relevant features from the original set of features for building robust learning models (see Fig. 3.1), i.e., to find the features that best help

separate data of different classes (class-based separation) [82]. Thus the aim is to select the least number of features that maximize the classification performance which is generally expressed in terms of accuracy.

Feature extraction [67, 87] is a technique that, from the original set of features, creates another set of features where some original features could have been merged or elaborated in "some ways" (see Fig. 3.1).

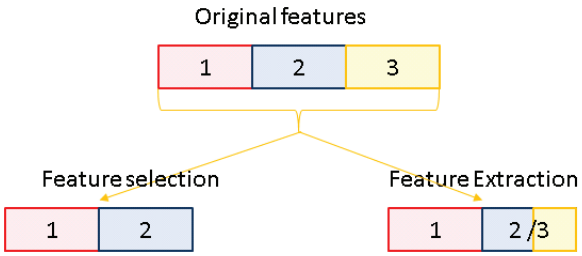


Figure 3.1 *Feature selection: selection of a subset of features from the original set of features. Feature extraction: creation of another set of features where some original features could have been merged or elaborated in "some ways". In the example a feature is maintained while the other two are "fused" together.*

If the selected/extracted features are carefully selected/extracted it is expected that the new feature set contains the (most) relevant information from the input data. Thus, by removing most irrelevant and redundant features from the data, feature selection/extraction generally helps improve the performance of learning models by:

- alleviating the effect of the curse of dimensionality by reducing the feature space dimensionality,
- enhancing generalization capability by increasing the classification accuracy,
- speeding up the learning process,
- reducing the memory needed for data representation,
- improving model interpretability.

3.1.1 Forward Feature Selection algorithm

There are many different algorithms which perform the feature selection and the *Forward Feature Selection* (FFS) method is one of the most used. FFS is a greedy algorithm that, starting with an empty set of features, adds the features one by one, selecting at each step the fea-

ture which decreases less the accuracy [21, 22, 23, 138, 141, 142]. The algorithm stops when a chosen stop-criterion is met. In particular the stop criterion can be the maximum number of features to be selected, i.e., the algorithm is stopped when the predefined number of features has been selected [67, 73, 82, 141].

3.1.2 Individual Feature Selection algorithm

Another commonly used algorithm to perform feature selection is the *Individual Feature Selection* (IFS) which computes the accuracy for each feature separately [138], then it selects as many features as specified by the user choosing the ones that achieve the highest accuracies. Another ways to select the features is to select all the features that achieve an accuracy higher than a chosen threshold. The advantage of this algorithm is its high speed. For this reason it is often used for pre-selection of a candidate feature subset from a larger set of features. However since individually poor features may reach high class separability when used together, this algorithm may discard potentially useful features.

3.1.3 Principal Component Analysis algorithm

The most known feature extraction algorithm is called *Principal Component Analysis* (PCA) [63, 98]. It involves a mathematical procedure that transforms the original set of possibly correlated features into a smaller number of uncorrelated features called *principal components*.

Figs. 3.2 and 3.3 show an example of feature selection and feature extraction, respectively, applied to the same 4-class 300-feature data set with a reduction of the feature space to a 2-dimension space.

In this dissertation we mainly use the FFS algorithm since it achieved the best results also compared to the PCA algorithm.

3.2 Multi-class Classifiers

In this chapter we introduce some classifiers we have used during our experiments. Some of these classifiers are already well-known while others were implemented during this work.

3.2.1 Linear discriminant classifier

The *linear discriminant analysis* (LDA) method dating back to Fisher's linear discriminant [39] is a method used in many fields such as pattern recognition [73]. LDA consists of searching some linear combinations

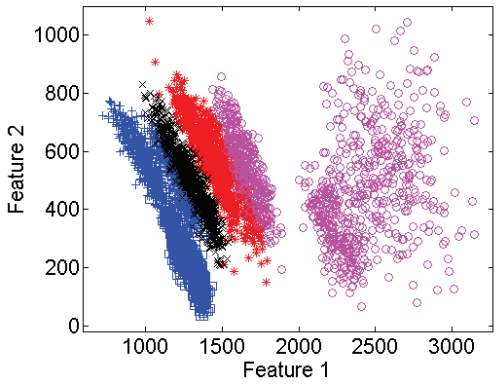


Figure 3.2 *Application of a feature selection algorithm (Forward Feature Selection). Reduction of the 4-class data set from a 300-dimension space to a 2-dimension space.*

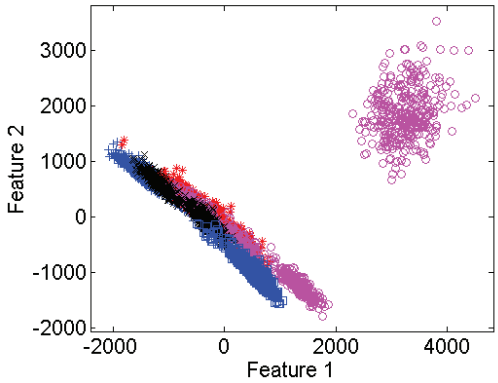


Figure 3.3 *Application of a feature extraction algorithm (Principal Component Analysis). Reduction of the 4-class data set from a 300-dimension space to a 2-dimension space.*

of the "input" variables (features), which provide the best separation between the considered classes [73]. The resulting combination may be used as a linear classifier, or, more commonly, for dimensionality reduction before later classification [21, 22, 73, 76, 142].

The *linear discriminant classifier* (LDC) is the linear minimum-error (Bayes) classifier used for normally distributed classes with equal covariance matrices [73]. LDC is simple to calculate from data and is a fast and reasonably robust classifier [32, 73, 81, 145] characterized by only one parameter r, called *regularization parameter* (one degree

of freedom), which we consider fixed to 0 for all the performed experiments.

An example of the use of an LDC to perform a classification of a data set characterized by two features and three classes (see Fig. 3.4) is reported in Fig. 3.5.

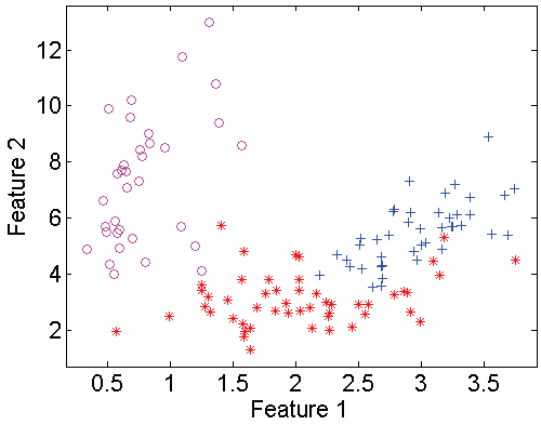


Figure 3.4 *Data set characterized by 2 features and 3 classes. Each class is represented by a different symbol.*

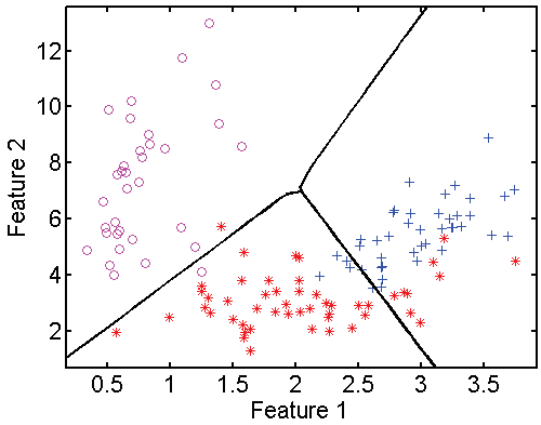


Figure 3.5 *Classification of the data set represented in Fig. 3.4 using an LDC classifier. The black line represents the LDC discriminant function.*

3.2.2 Quadratic discriminant classifier

A standard approach to supervised classification problems is the quadratic discriminant analysis (QDA), which, as well as LDA, is named after the type of discriminant functions it uses [73]. However, unlike LDA, in QDA there is no assumption that the covariance of each of the classes is identical.

In particular, the *quadratic discriminant classifier* (shortly, QDC) is the quadratic minimum-error (Bayes) classifier used for normally distributed classes with class-specific covariance matrices [32, 73, 145]. QDC, as LDC, is simple to calculate from data and is a fast and reasonably robust classifier, characterized by only one parameter r, called *regularization parameter* (one degree of freedom), which we consider fixed to 0 for all the performed experiments.

An example of the use of a QDC to perform a classification of the 3-class 2-feature data set of Fig. 3.4 is reported in Fig. 3.6.

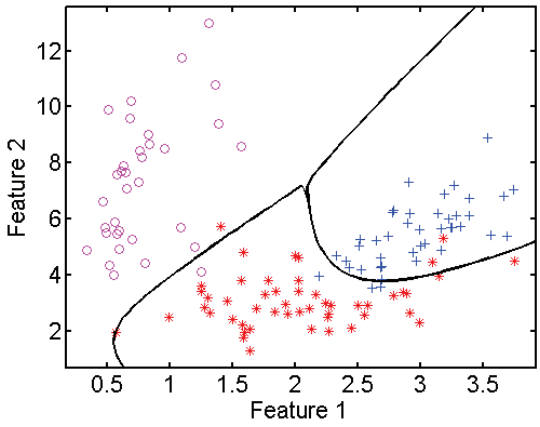


Figure 3.6 *Classification of the data set represented in Fig. 3.4 using a QDC classifier. The black line represents the QDC discriminant function.*

3.2.3 Neural Networks

Artificial neural networks were originally motivated by an interest in modelling the human brain [34, 52, 73, 85]. Literature on neural networks includes many publications including books, papers and so on [6, 12, 34, 36, 47, 50, 73, 83, 97, 107, 108].

Neural networks are made up of interconnecting artificial neurons

whose structure is represented in Fig. 3.7. The artificial neuron receives one or more inputs x_i and sums them to produce an output y. Usually the inputs are weighted using opportune weights w_i, and the sum is passed through a function f known as *activation function* or *transfer function*. There exist many types of transfer functions such as linear, non-linear, sigmoidal, piecewise, and so on. Finally there can be a threshold θ which decreases the activation of the neuron. Thus the output is evaluated as in eq. 3.1.

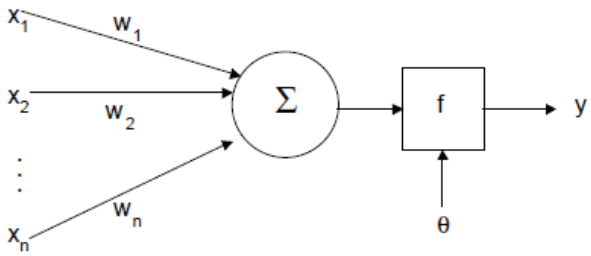


Figure 3.7 *Artificial neuron structure. x_i: inputs, y: output, w_i: weights, f: activation function, θ: threshold.*

$$y = f(\sum_{i=1}^{n} w_i x_i - \theta) \tag{3.1}$$

Two of the most popular neural network models are the *Multi-Layer Perceptron* (MLP) neural network [34, 73] and the *Radial Basis Function* (RBF) neural networks.

Rosenblatt's Perceptron

The *perceptron* is a type of artificial neural network. The original perceptron model and its famous training algorithm were developed by Frank Rosenblatt in 1958 [111]. The perceptron is implemented as in eq. 3.2 [34, 73, 111]:

$$\phi(\xi) = \begin{cases} 1, & \text{if } \xi \geq 0, \\ -1, & \text{otherwise,} \end{cases} \tag{3.2}$$

where ξ represents the activation.

The perceptron is a one-neuron classifier which is able to separate two classes in $\mathbb{R}^n$ by the linear discriminant function defined by $\xi = 0$ [34, 73].

Multi-Layer Perceptron neural network

A *Multi-Layer Perceptron* (MLP) neural network can be obtained connecting more perceptrons [34]. This is a feedforward structure because the outputs of the input layer and all intermediate layers are submitted only to the following layers. The generic model of a feedforward neural network classifier is shown in Fig. 3.8, where "layer" means a layer of perceptrons. There are three types of layers: the input layer, the hidden layer (which can be one or more), and the output layer [34]. Each layer is typically characterized by a specific transfer function.

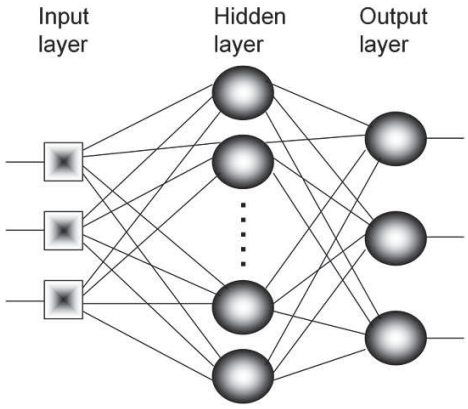


Figure 3.8 *MLP structure. "Layer" means a layer of perceptrons. There are three types of layers: the input layer, the hidden layer (which can be one or more), and the output layer.*

The number of hidden layers and the number of perceptrons at each hidden layer is decided by the designer of the MLP. The neurons in the hidden layers are often referred to as "hidden neurons" (HNs). However, it was shown that an MLP with a single hidden layer and threshold nodes can approximate any function with a specified precision [12, 107].

MLP can be trained using the well-known *backpropagation* training algorithm. Usually the input layers are characterized by identity transfer functions. Some works such as [6, 12, 34, 36, 47, 73, 83, 50, 97, 107, 108] and many others could usefully be consulted to gain insight into the MLP model, the backpropagation algorithm, and its variants.

An example of a 2-feature 2-class banana data set is represented in Fig. 3.9, while Fig. 3.10 shows the discriminant functions of some MLPs with one hidden layer characterized by a logarithmic sigmoidal transfer function in the hidden layer and trained using the backpropagation algorithm.

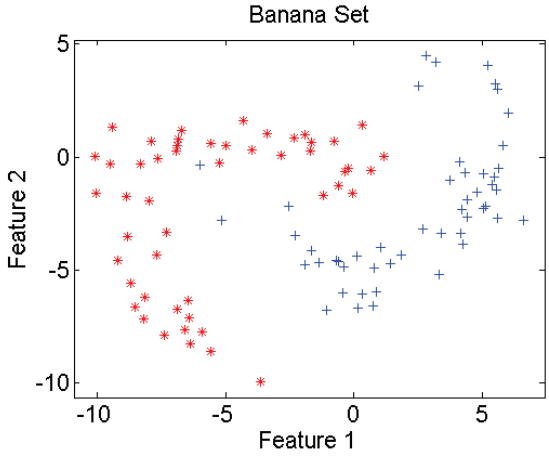


Figure 3.9 *2-feature 2-class banana data set.*

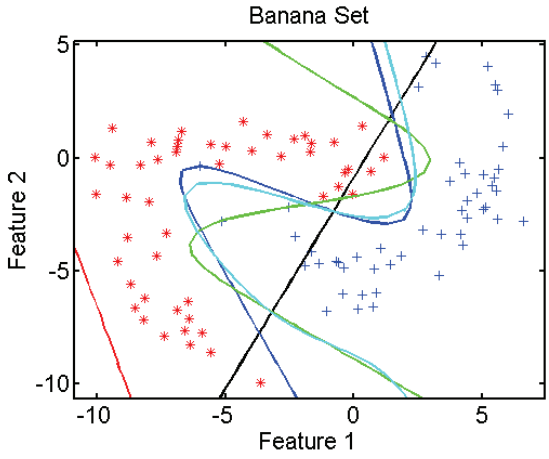


Figure 3.10 *Classification of the data set shown in Fig. 3.9 performed by some MLPs characterized by one hidden layer with a logarithmic sigmoid transfer function and trained using the backpropagation algorithm. These MLPs differ for the number of HNs: 1 (black), 2 (red), 3 (blue), 4 (green), 5 (cyan).*

Radial Basis Function neural network

A *Radial Basis Function neural network* (RBF) is a neural network that uses *radial basis functions* as activation functions. A radial basis function is a real-valued function whose value depends only on the distance from the origin as represented in eq. 3.3

$$f(\mathbf{x}) = \phi(\|\mathbf{x}\|) \tag{3.3}$$

or, alternatively, on the distance from some other point c, called *centre*, as in eq. 3.4

$$f(\mathbf{x}, \mathbf{c}) = \phi(\|\mathbf{x} - \mathbf{c}\|). \tag{3.4}$$

RBFs generally present three layers: an input layer, a hidden layer with a non-linear *radial basis function* activation function and a linear output layer. More information about RBF neural networks can be found in [12, 73, 83].

Fig. 3.11 shows the discriminant functions of some RBFs with one hidden layer characterized by a Gaussian transfer function and different numbers of HN classifying the data set in Fig. 3.9.

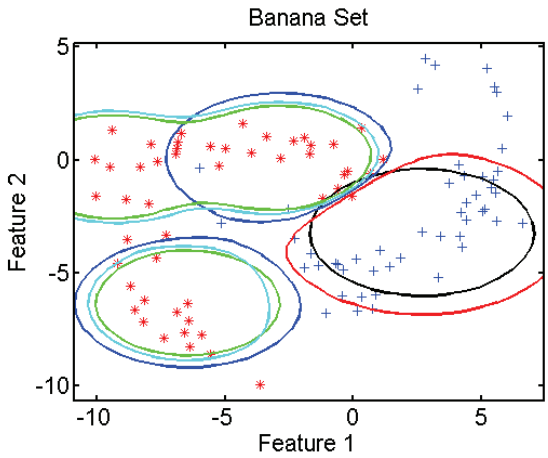


Figure 3.11 *Classification of the data set shown in Fig. 3.9 performed by some RBFs characterized by a Gaussian transfer function in the hidden layer. These RBFs differ for the number of HNs: 1 (black), 2 (red), 3 (blue), 4 (green), 5 (cyan).*

3.3 Classifier ensembles

A classifier ensemble is a set of classifiers that are jointly used to increase the classification accuracy [48, 67, 73].

According to Dietterich [30], a classifier ensemble can be more convenient than a single classifier for three main reasons [73]. First, when several good classifiers are available to solve a problem we might decide to use only one of them based on their training accuracy with the risk of not choosing the best in terms of generalization performance. A better choice would be to appropriately combine their outputs, thus avoiding adopting an inadequate classifier. Second, appropriate combination techniques may produce a better approximation of the ideal optimal classifier for the given data set. Third, an ensemble of simple classifiers, combined through a simple rule, may even outperform a single complex, generally more powerful, classifier.

In the literature different methods have been proposed to combine the classifier outputs depending on the type of these outputs. Xu et al. [151] distinguish three types of classifiers outputs:

- *abstract level*: each classifier output is a class label;
- *rank level*: each classifier output is the list of all class labels sorted according to the plausibility of each label to be the correct one [54, 73, 132];
- *measurement level*: each classifier output is a vector of as many elements as there are classes. The i–th component represents the degree of support to the hypothesis that the input to the classifier comes from the i–th class.

There are two main approaches to combine classifiers, namely, *fusion* [73, 74] and *selection* [26, 73, 104], which differ in the assumption regarding the knowledge of the feature space by the single classifiers. In the former case, all classifiers know the whole space, whereas in the latter case each single classifier is considered as an expert in a specific portion of such space [73].

3.3.1 Classifiers Fusion

The fusion of classifiers represents one of the most widely used approaches to combine classifiers.

The degrees of support for a given input x with $x \in \mathbb{R}^n$ can be interpreted in different ways, the two most common are *confidences* in

the suggested labels and *estimates of the posterior probabilities* for the classes [73]. Let

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\} \tag{3.5}$$

be the set of c class labels and

$$D = \{D_1, \ldots, D_L\} \tag{3.6}$$

the ensemble of L classifiers. Given an input feature vector x, $x \in \mathbb{R}^n$, each classifier D_i provides a degree of support $d_{i,j}(x)$ to the hypothesis that x comes from class ω_j. The value $d_{i,j}(x)$, typically ranging in the interval [0,1], can be considered as a confidence in the associated label or an estimate of the posterior probability for the class.

The L classifier outputs for an input x can be represented in a matrix, called *decision profile*, $DP(x)$:

$$\begin{bmatrix} d_{1,1}(x) & \cdots & d_{1,j}(x) & \cdots & d_{1,c}(x) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ d_{i,1}(x) & \cdots & d_{i,j}(x) & \cdots & d_{i,c}(x) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ d_{L,1}(x) & \cdots & d_{L,j}(x) & \cdots & d_{L,c}(x) \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{3.7}$$

where the i-th row,

$$\begin{bmatrix} d_{i,1}(x) & \cdots & d_{i,j}(x) & \cdots & d_{i,c}(x) \end{bmatrix}, \tag{3.8}$$

represents the output $D_i(x)$ of classifier D_i, while the j–th column,

$$\begin{bmatrix} d_{1,j}(x) & \cdots & d_{i,j}(x) & \cdots & d_{L,j}(x) \end{bmatrix}^T, \tag{3.9}$$

where T represents the transpose, is the support for class ω_j from classifiers D_i, $\cdots$, D_L.

We can use $DP(x)$ to find the overall support for each class and then

assign the input x to the class with the largest support [73]. One widely used approach consists in computing the overall support μ_j for the j-th class using only the j-th column of the matrix [73]. Combination strategies adopting this approach are called *class-conscious* [73, 74]. An example are the so-called *nontrainable combiners*, which compute $\mu_j(x)$ by simply applying a combination function (such as *simple mean*, *minimum*, *maximum*, *product*, etc.) to the single class supports, without requiring any further training after the base classifiers have been trained. Then the input x is assigned to the class with the maximum $\mu_j(x)$. The operators used in the present work as combination functions are the following [73]:

- *Simple mean*: $\mu_j(x) = \frac{1}{L}\sum_{i=1}^{L} d_{i,j}(x)$,
- *Maximum*: $\mu_j(x) = \max_i \{d_{i,j}(x)\}$,
- *Minimum*: $\mu_j(x) = \min_i \{d_{i,j}(x)\}$,
- *Product*: $\mu_j(x) = \frac{1}{L}\prod_{i=1}^{L} d_{i,j}(x)$.

3.3.2 Classifier Selection

In classifier selection, each single classifier is responsible for a specific portion of the feature space; thus different classifiers are used for different inputs [26, 73]. Actually, the key issue is to decide which classifier should be used to label a particular input. It can be shown that, if for each input x we choose the classifier that results to be the best in the region containing x, the selection scheme achieves at least the same accuracy as the best classifier in the ensemble [73].

In the literature several approaches to select the best classifier have been proposed, including static and dynamic ones. One approach consists in dynamically selecting the classifier by estimating the local competence of each classifier, like the decision-independent estimates strategy [45, 73], in which the competence is established based on the location of the input x without considering the classifiers outputs.

One such method is the *Direct K-NN Estimate* [73, 149], which simply and fast estimates the competence based on the accuracy achieved by the classifiers on the K nearest neighbors of x from either the training set or the validation set [73, 149]. K is, of course, a parameter of

the algorithm. In our experiment, we fix $K = 1$ to obtain the fastest algorithm.

When $K = 1$, the *Direct K-NN Estimate* performs the following steps: when the input x is submitted for classification, it looks for the nearest neighbor x_{nn} of x in the training set, then it evaluates the *decision profile* and selects as the classifier responsible for x, the classifier that returns the maximum value in the support for the class to which x_{nn} belongs. Ties are broken randomly. The steps performed, both in the training and test phases, by the Direct K-NN Estimate algorithm when $K = 1$ are detailed in Fig. 3.12.

Let TR be the training set and TS the test set.
Let c_i (with $i = 1$ to n, and n the total number of classes) represent the i-th class.

Train phase
Train a set of L classifiers using the data set TR.

Test phase
For each $\mathbf{x} \in TS$,

1. **Find the nearest neighbor $\mathbf{x}_{nn}$ of $\mathbf{x}$ inside TR.**
2. **Let $DP_{\mathbf{x}_{nn}}$ be the decision profile for the sample $\mathbf{x}_{nn}$ evaluated using all the L trained classifiers.**
3. **Let c_j be the class to which $\mathbf{x}_{nn}$ belongs. Let $DP^j(\mathbf{x}_{nn})$ be the j-th column of $DP_{\mathbf{x}_{nn}}$.**
4. **Find the row index (say k) of the element in $DP^j(\mathbf{x}_{nn})$ with maximum value. The k-th classifier is therefore considered the most accurate on the specific region containing $\mathbf{x}$ and $\mathbf{x}_{nn}$.**
5. **Classify $\mathbf{x}$ using the k-th classifier.**

Figure 3.12 *Steps performed by the Direct K-NN Estimate algorithm when $K = 1$. Train and test phases.*

3.4 One-class classification

One class classification [71, 84, 89, 99, 126] is an unsupervised classification strategy because it assumes that only information of one class, the *known class* (exhaustively sampled class), is available for training. Unlike two-class classification, one-class classification tries to describe the known class while learning nothing about the other class (*unknown*

class) during the training process [71, 84, 89, 99, 126]. Later, during the test process, the classifier distinguishes and thus classifies the *unknown samples* by means of their dissimilarity to the *known samples*.

The reasons for the absence of *unknown* samples, i.e., the samples belonging to the *unknown* class, can be several [64, 127]:

- the high measurement costs (e.g., need to put damaged bearings into a rotating machine causing unwanted consequences) [142],
- the low frequency of an event (e.g., a nuclear power plant failure or a rare medical disease),
- impossibility to collect and "catalog" all the possible types of *unknown* samples (e.g., to collect the infinite number of faults and severity levels of a faulty bearing in order to distinguish between a faultless bearing and a damaged one [142] or to identify all the possible human illnesses in order to distinguish a healthy person from an ill person).

Furthermore, even when available, *unknown* samples may not always be trusted, as they can be badly represented, for example, because they are affected by high level of noise.

On the other hand, it can be relatively cheap and simple to obtain measurements from a particular class (*known* class), e.g., to collect measurements from faultless bearings and thus from a normally functioning machine [142].

Reducing the classification problem of interest to a one-class problem brings to several advantages such as [64, 127]:

- increase of the independence from the collected data, i.e., more generalization ability, since the classification can be performed regardless of the lack of samples of one of the two classes;
- reduction of the computational time, since fewer data are processed (only *known* samples) during the training process. In the literature there are rarely references even to the order of magnitude of the computational time required by the proposed classification system to perform the classification process, however this is a crucial issue in real-time applications;
- reduction of the time required to collect data since fewer data (only *known* samples) should be collected;

- reduction of the required memory to store all the data since fewer data (only *known* samples) should be stored.

Therefore, the area of interest in one-class classification covers all the problems of detection by distinguishing a specified and exhaustively sampled *known* class from all kinds of "anomal" *unknown* class [64]. The applications are many such as fault detection [142, 156], rare illnesses [124], authorship verification [71], and so on.

The principles behind many two-class or multi-class classifiers can be used for solving one-class classification problems [12, 18, 55, 59, 62, 64, 69, 95, 96, 113].

The characterization of the *known* class can be performed by means of different approaches such as the *density estimation* approach (e.g., the Gaussian model), and the *boundary* approach (e.g., the nearest neighbor approach) [64, 127].

The first one, the *density estimation* approach, involves a density estimation of the *known* class and thus assumes that the *known* data is exhaustively sampled, and that low density areas in the training set indicate that these areas have a low probability of containing *known* objects [64]. If an exhaustive sampling is difficult to perform, other approaches should be used. More precisely, when only the data boundary is required we do not need to estimate a complete data density for a one-class classifier which might also be too demanding. This is why in the boundary methods only a closed boundary around the *known* set is optimized.

Finally, in this section, we introduce the one-class classifiers developed during this Ph.D work. The proposed classifiers are the convex hull classifier (CHC), the snake operator classifier (SOC), and the CSC and CSS classifiers which are obtained by the combination of the CHC and SOC. The first (CSC) alternates the use of more CHCs and SOCs starting from a CHC, while the second (CSS) consists in a CHC stretched starting from an SOC applied on a CHC. The two one-class classifiers, CSC and CSS, will be described in more details in the following.

3.4.1 Density estimation based one-class classifiers

Density estimators are the most popular and straightforward methods to obtain one-class classifiers [64, 124, 127].

The density methods perform an estimation of the complete prob-

ability density of the *known* class using the training set. Then in the test phase they decide the class to which a sample belongs on the base of a specific threshold and the estimated probability density [11, 18]: if the estimated probability is higher than the threshold the sample is classified as belonging to the *known* class, otherwise to the *unknown* class [64, 127].

This approach considers as hypothesis that the training size (*known* size) is "sufficiently" high [64, 127]. Thus, when the training set is "enough" this approach can reach good performance, generally approximating the *known* density by much simpler models and, thus, avoiding, to try to evaluate a full density probability estimation [64]. Generally, a flexible density model is used [127].

The drawback of the density estimation approach is that the estimation of the probability density might be a difficult task which is even harder when very few samples are available [31, 64, 127].

Gaussian density estimation one-class classifier

The simplest *statistical model* is the *normal* or *Gaussian* density [12]. According to the Central Limit Theorem [133], this model is correct when we assume that samples from one class originate from one prototype which is disturbed by a large number of small independent disturbances [127].

For this density model the conditional probability p_N for the *known* class that a new object x belongs to the *known* class [127] can be described as in eq. 3.10:

$$p_N\left(x;\mu,\sum\right) = \frac{1}{(2\pi)^N \| \sum \|^{1/2}} \exp\{-\frac{1}{2}(x-\mu)^T \sum^{-1}(x-\mu)\} \quad (3.10)$$

where μ is the mean and $\sum$ represents the covariance matrix.

The model is very simple. However it imposes a strict unimodal and convex density model on the data and suffers from very large covariance matrices, that are hard to estimate since the computation of their inverse becomes easily ill-defined [64, 127].

An example of application of the one-class Gaussian classifier on the data set in Fig. 3.13 is shown in Fig. 3.14. In these figures three Gaussian classifiers are represented. They are characterized by different thresholds θ_i ($\theta_i \in [0, 1]$). These thresholds define the fraction

of *known* samples which can be rejected. This means that not more than $(100 \times \theta_i)\%$ of the positive class (*known* class) can be misclassified (false negatives). That means that the fraction of false negatives will be $(100 \times \theta_i)\%$.

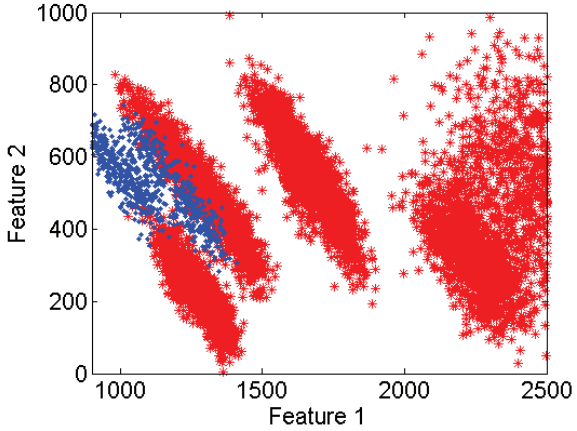


Figure 3.13 *2-feature 2-class data set. The known class is represented by the blue points while the unknown class is represented by the red stars.*

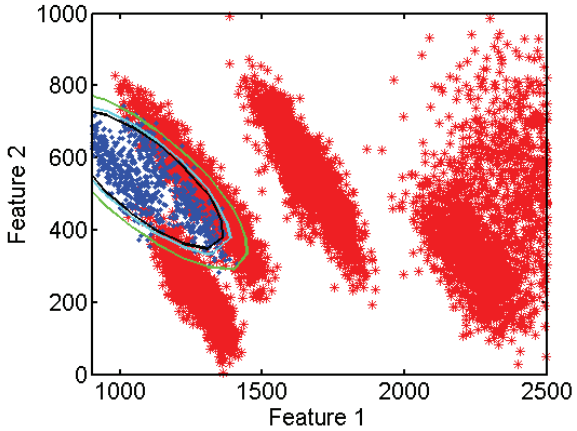


Figure 3.14 *Classification of the data set represented in Fig. 3.13 using three one-class Gaussian classifiers characterized by* $\theta_i = 0.01$ *(green),* $\theta_i = 0.05$ *(cyan),* $\theta_i = 0.09$ *(black).*

3.4.2 Boundary based one-class classifiers

When density estimation is not feasible (too few *known* samples), one can approximate the *known* class by a simpler model making use of the boundary-based approach. As stated in [137], when just a limited amount of data is available, one should avoid solving a more general problem as an intermediate step to solve the original problem. To solve this more general problem more data might be required than for the original problem. In our case this means that estimating a complete data density for a one-class classifier might also be too demanding when only the data boundary is required [64, 127].

Thus the idea at the base of the boundary-based strategy is to create a model which captures the data structure, i.e., only a closed boundary around the *known* set is optimized. Then new objects are projected onto this model. Although the objective of these algorithms is not to minimize the boundary around the *known* class, most methods have a strong bias towards a minimal volume solution. However how small the volume is, depends on the fit of the method to the data [64, 127].

Since the boundary methods heavily rely on the distances between objects, they tend to be sensitive to the scaling of the features. On the other hand the number of objects that is required is smaller than in case of the density methods. So, although the required sample size for the boundary methods is smaller than the density methods, a part of the burden is now put onto well-defined distances [64, 127].

Nearest Neighbor one-class classifier

The nearest neighbor one-class classifier (NNDD) [64, 127] is one of the simplest boundary-based one-class classifier. It can be derived from the nearest neighbor classifier of [31], which is a classifier which bases its decisions on the distance between the objects and their nearest objects (nearest neighbors) in the training set. Thus the NNDD classifier avoids to evaluate explicitly the complete density estimation [127].

The training phase in NNDD consists only of the storage of all the samples of the training set in memory. In the test phase, NNDD has to make an exhaustive search considering the set of training samples to find the nearest neighbor of each test sample. More precisely, a test object x is assigned to the *known* class when its local density (inverse of distance from its nearest neighbor in the training set $\mathrm{NN}(x)$) is larger or equal to the local density of its (first) nearest neighbor (inverse of

distance between NN(x) and its nearest neighbor NN(NN(x)) in the training set). Otherwise x is assigned to the *unknown* class. This is summarized in eq. 3.11.

$$f(x) = \begin{cases} \textit{known} \text{ class}, & \frac{\|(x,\mathrm{NN}(x)\|}{\|(\mathrm{NN}(x),\mathrm{NN}(\mathrm{NN}(x))\|} < 1 \\ \textit{unknown} \text{ class}, & \text{otherwise} \end{cases} \tag{3.11}$$

where NN(x) is the nearest neighbor of x in the training set, NN(NN(x) is the nearest neighbor of NN(x) in the training set and "$\|\cdot\|$" represents the *Euclidean distance* between two objects.

The NNDD classifier can of course be extended to a larger number $k \geq 1$ of nearest neighbors. Instead of taking the first nearest neighbor into account, the k-th neighbor should be considered such as in the well-know multi-class classifier k-NN. We will call k-NNDD the generalization to $k \geq 1$ of the NNDD classifier.

Two drawbacks of the NNDD classifier as well as the k-NNDD classifier are their need to store all the training samples that are used subsequently to classify unseen objects and to evaluate the distances of each unseen object from all the objects in the training set. This makes these algorithms *scale sensitive* in term of memory and computational time.

An example of application of the NNDD and k-NNDD classifiers on the data set in Fig. 3.13 are shown, respectively, in Fig. 3.15 and in Fig. 3.16. Three NNDDs and three k-NNDDs classifiers are represented. They are characterized by different thresholds θ_i ($\theta_i \in [0, 1]$). These thresholds define the fraction (%) of *known* samples which can be rejected. This means that not more than $(100 \times \theta_i)$% of the positive class (*known* class) can be misclassified (false negatives). That means that the fraction of false negatives will be $(100 \times \theta_i)$%.

3.4.3 Convex hull classifier

The convex hull (CH) for a set of points $\mathbb{S}$ in a real vector space $\mathbb{V}$ is defined as the minimal convex set containing $\mathbb{S}$ [16, 25]. In the literature, convex hulls are usually adopted in application domains such as computer visualization, verification methods and computational geometry problems, some of which are described in [38, 44]. The convex hull

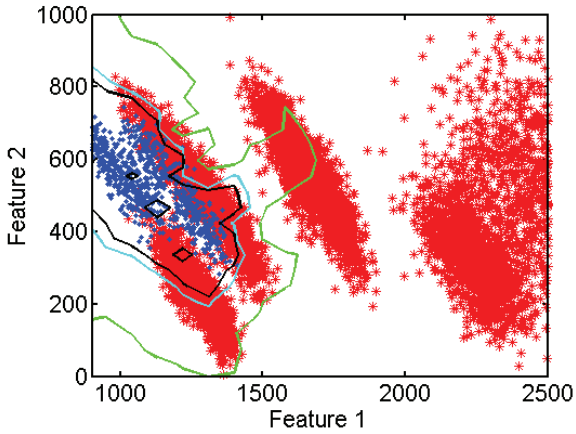


Figure 3.15 *Classification of the data set represented in Fig. 3.13 using three NNDD classifiers characterized by* $\theta_i = 0.01$ *(green),* $\theta_i = 0.05$ *(cyan),* $\theta_i = 0.09$ *(black).*

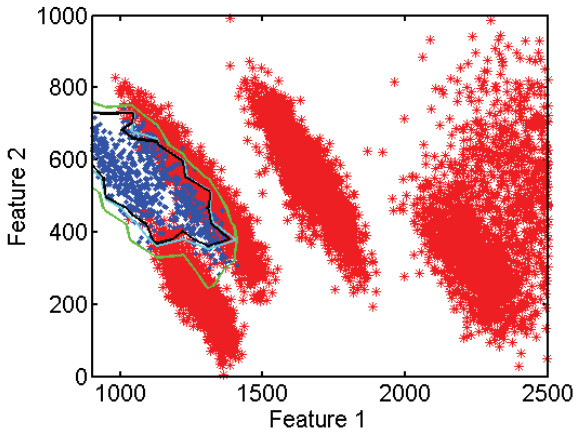


Figure 3.16 *Classification of the data set represented in Fig. 3.13 using three k-NNDD classifiers (k optimized using a leave-one-out density estimation) characterized by* $\theta_i = 0.01$ *(green),* $\theta_i = 0.05$ *(cyan),* $\theta_i = 0.09$ *(black).*

classifier (CHC) classifies the samples into two sets: the first (*known* class) consisting of the points inside the convex hull border and the second one (*unknown* class) consisting of the points outside the convex hull border (Fig. 3.17).

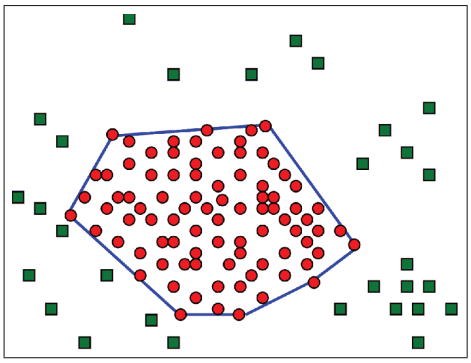


Figure 3.17 *Classification performed by a convex hull classifier (CHC): the known class consists of the points (dots) inside the convex hull border (spline) and the unknown class consists of the samples (squares) outside the convex hull border.*

3.4.4 Snake operator classifier

A snake operator [65] is an active contour model consisting of a spline that tries to minimize its energy basing itself on *internal constraint forces*, *external constraint forces* and *image forces* that pull it toward features such as lines and edges. In [65] these three types of energy are described in the following way:

- *internal constraint forces*: forces that impose to the snake to maintain a piecewise smoothness form,
- *images forces*: forces that push the snake towards salient image features like lines, edges, and subjective contours,
- *external constraint forces*: forces that are responsible for driving the snake near the desired local minimum.

Thus, if we represent the position of the snake $v(s,t)$, where s and t are, respectively, the spatial index and the time, defined on given open intervals Ω and T, then the snake energy E_{snake}, which should be minimized, can be written as follows in eq. 3.12:

$$E_{snake} = \frac{1}{2}\int_{\Omega} [E_{int}(v(s)) + E_{image}(v(s)) + E_{ext}(v(s))]\, ds \qquad (3.12)$$

where E_{int} represents the internal constraint forces, E_{image} the image forces, and E_{ext} the external constraint forces [65].

The snake operator is a valuable tool commonly used to segment images with a lot of applications such as in the medical image segmentation [86, 139, 140, 148].

In this dissertation, we use the snake operator as a classifier, interpreting the samples of our data set as an image to be segmented. The snake operator classifier (SOC) classifies the samples into two sets: the first consisting of the points inside the snake border and the second one consisting of the points outside the snake border. In our experiments we use the SOC in a two dimensional (2D) space.

One of the crucial aspects of the use of the snake operator is its initialization [139, 140], which is usually provided manually by an expert during the image segmentation [102]. Here we propose to initialize the snake operator using the curve resulting from the application of a 2D convex hull to the same data.

3.4.5 CSC classifier

The CSC classifier is a combination of the previously described one-class classifiers: convex hull and snake classifiers. This classifier operates as follows:

1 Build the convex hull containing all the training samples of the faultless class.

2 Consider this convex hull as the mask to initialize a snake operator. Let the snake evolve for a certain number N of epochs. N is generally a small number, optimized and chosen by the user. In our experiment we fix N to 300 (generally, the smaller N the more accurate the evolution is), however this number is strictly dependent on the specific context in which the snake operator is used.

3 Consider the faultless samples of the training set inside the snake, and build on them a new convex hull. If this convex hull is different from the previous one, then go to step 4, otherwise let the snake evolve for other N epochs and repeat step 3.

4 If the stop criterion is reached, stop the algorithm, otherwise repeat step 2.

The process can be stopped when the minimum acceptable accuracy for the faultless class is reached. More precisely, as the snake and the convex hull evolve, the accuracy of the faultless class continues to de-

crease (or remain unchanged), while the accuracy of the damaged class continues to increase (or remains unchanged). Thus, if the accuracy for the faultless class considering the first convex hull classifier is *acc*, then the accuracy for the faultless class for the following convex hull classifiers will be $acc - \epsilon$, with $\epsilon \geq 0$. Of course the accuracy for the damaged class will increase (or remains unchanged). The same happens between subsequent snake operators. Thus, the user can select the minimum acceptable accuracy T for the faultless class, then stop the process when this accuracy is reached. In this way we will obtain the maximum accuracy for the damaged class given the specific faultless accuracy T.

3.4.6 CSS classifier

The CSS classifier is a combination of the previously described one-class classifiers: CHC and SOC followed by a stretching process. This classifier works as follows:

1 Build the convex hull containing all the training samples of the faultless class.

2 Use this convex hull as a mask to initialize a snake operator. Let the snake operator evolve for a certain number M of epochs or until the snake reaches an "equilibrium state" corresponding to the equilibrium of the involved forces. M is generally a large number optimized and chosen by the user. In our experiments we fix M to 10000, however this number is strictly dependent on the specific context in which the snake operator is used.

3 Consider the faultless samples of the training set inside the snake and build on them a new convex hull.

4 Stretch this convex hull by enlarging it by a factor γ. γ is related to the specific classification problem. In particular the stretching is made so that the result is proportional to the "width of the specific feature" by a stretching factor called $stretch_{Perc}$, which is initialized to 1% in our experiments. By "width of a feature" we mean the difference between the maximum and the minimum values of that feature over all the faultless training samples. Let D_x and D_y be the widths of the faultless class features (only the samples inside the convex hull are considered), then the stretching is done proportional to D_x and D_y. Thus the factor of stretching

γ will be evaluated as in eq. 3.13:

$$\gamma = (\gamma_x, \gamma_y) = (stretch_{Perc} \times D_x, stretch_{Perc} \times D_y). \tag{3.13}$$

The stretching phase is performed in the following way. Each sample of the training set (composed only by faultless elements) is quadrupled. More precisely, let s be a generic training sample, and let x_s and y_s be the values of the two *discriminant features* (DFs) describing it. Then add to the training set four more elements $s1(x_{s1}, y_{s1})$, $s2(x_{s2}, y_{s2})$, $s3(x_{s3}, y_{s3})$, and $s4(x_{s4}, y_{s4})$, where:

- $x_{s1} = x_s + \gamma_x$ and $y_{s1} = y_s + \gamma_y$;
- $x_{s1} = x_s + \gamma_x$ and $y_{s1} = y_s - \gamma_y$;
- $x_{s1} = x_s - \gamma_x$ and $y_{s1} = y_s + \gamma_y$;
- $x_{s1} = x_s - \gamma_x$ and $y_{s1} = y_s - \gamma_y$.

Build on this new training set a new convex hull that will be larger than (stretched from) the previous one. Reset the training set to the original samples, but selecting only the samples inside the current convex hull. If this set is the same as that used to build the previous convex hull, then repeat step 4 with an increased $stretch_{Perc}$, namely $stretch_{Perc} = stretch_{Perc} + \gamma_0$ (γ_0 is set to 1% in the performed experiments), otherwise go to step 5.

5 If the stop criterion is reached stop the algorithm, otherwise set $stretch_{Perc}$ to 1% again and repeat step 4.

The process can be stopped when the minimum acceptable accuracy for the faultless class is reached. More precisely, the more the convex hull is stretched, the more the accuracy of the faultless class continues to increase (or remain unchanged), while the damaged accuracy continues to decrease (or remains unchanged). Thus, if the accuracy for the faultless class considering the first convex hull classifier is *acc*, then the accuracy for the faultless class for the following stretched convex hulls will be $acc + \delta$, with $\delta \geq 0$. Of course the accuracy for the damaged class will decrease (or remains unchanged). Thus the user can select the minimum acceptable accuracy T' for the faultless class, then stop the process when this accuracy is reached. In this way we will obtain the maximum accuracy for the damaged class for the given faultless

accuracy T'. An iteration of the process is represented in Fig. 3.18.

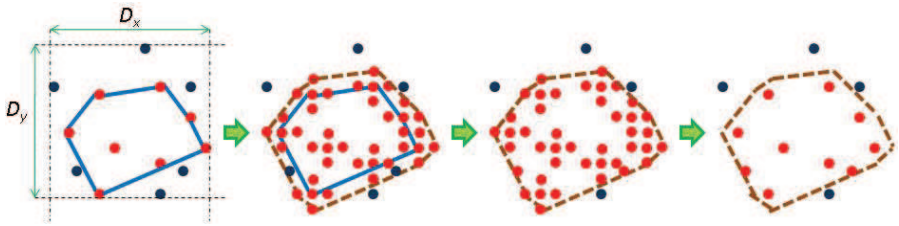


Figure 3.18 *An iteration of the evolution of the CSS classifier. In the leftmost subfigure the blue points represent the training samples outside the convex hull while the red points represent the training samples inside the convex hull. The blue dashed line represents the initial convex hull that should be stretched. In the second subfigure four new points are created starting from each of the training points inside the convex hull. In the third subfigure the new convex hull is created (brown dashed line). In the last subfigure the original training samples are restored and the artificially created ones are removed.*

Chapter 4

Rolling element bearing data set

Many of life's failures
are people who did not realize
how close they were to success
when they gave up.

- T. Edison -

In this chapter we describe the data set used in all the performed experiments. The data set, as already stated, has been provided by Avio Propulsione Aerospaziale, via I Maggio, 99, Rivalta di Torino, Italy.

The structure of a bearing consists mainly in four main components:

- the inner raceway,
- the outer raceway,
- the balls,
- the cage.

In particular the structure of a bearing is represented in Fig. 4.1.

Two approaches have been adopted by researchers for creating defects on bearings in order to study their vibration response [125]. The first one consists in letting the bearing running into the machine until failure while vibration signals are continuously collected and analyzed to detect vibration changes [103, 105, 125]. The other approach consists in artificially introducing defects in the bearings by the use of specific techniques such as mechanical indentation [125] and then measuring their vibration response and comparing it with that of faultless

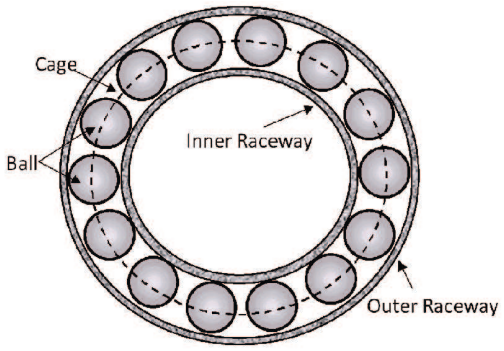


Figure 4.1 *General structure of a rolling element bearing. The four main components are the inner raceway, the outer raceway, the balls, and the cage.*

bearings [21, 76, 100, 125, 141]. In our case we have used the second approach.

The data set at our disposal consists of vibration signals coming from a rotating machine containing a certain number of bearings monitored by four accelerometers.

4.1 Data set description

4.1.1 Types of collected data

The signals were collected both with all faultless bearings and after substituting one faultless bearing with a damaged one. This bearing was artificially damaged and the data were collected before and after each damage. Thus we have data related both to faultless bearings and damaged bearings. In particular the following four types of defects were collected:

- indentation on the inner raceway,
- indentation on the roll (see Fig. 4.2),
- sandblasting of the inner raceway,
- unbalanced cage.

4.1.2 Subdivision of the data into classes and subclasses

Since we have data related both to faultless bearings and damaged bearings, we can divide our data into two main classes, i.e., *faultless class* and *damaged class*.

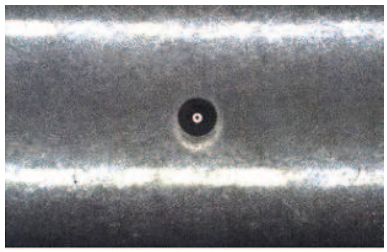


Figure 4.2 *Example of an indentation on the roll of a rolling element bearing.*

Then, since we have four different types of damage, we can further subdivide the damaged class into four classes, each of which corresponding to a specific type of fault.

Thus our data can be divided into five classes, namely, C1, C2, C3, C4, and C5, including one class for faultless bearings, C1, and one class for each damage: C2 (indentation on the inner raceway), C3 (indentation on the roll), C4 (sandblasting of the inner raceway), and C5 (unbalanced cage). The class subdivision is summarized in Table 4.1.

Table 4.1 *Classes subdivision*

Class	Type of class
C1	faultless bearing
C2	indentation on the inner raceway
C3	indentation on the roll
C4	sandblasting of the inner raceway
C5	unbalanced cage

In particular, the fault associated with class C2 consists of a 450 μm indentation on the inner raceway, while class C3, related to the indentation on the roll, can be further divided into three subclasses depending on the severity level of the damage, namely, *light* (C3.1, 450 μm indentation on the roll), *medium* (C3.2, 1.1 mm indentation on the roll), and *high* (C3.3, 1.29 mm indentation on the roll). The subdivision of class C3 into subclasses is summarized in Table 4.2.

Table 4.2 *Different levels of severity for class C3*

Subclass	Type of severity	Severity level
C3.1	indentation on the roll: 450 μm	light
C3.2	indentation on the roll: 1.1 mm	medium
C3.3	indentation on the roll: 1.29 mm	high

4.1.3 Data distribution

The data were recorded by the four accelerometers for time intervals of ten minutes. In particular, we considered a data set consisting of one-second signals with 2890 signals for class C1, 1770 for class C2, 4790 for class C3, 1520 for class C4, and 1770 for class C5. The distribution pf signals per class is shown in Table 4.3.

Table 4.3 *Distribution of signals for the classes C1, C2, C3, C4, and C5*

Class	Amount of signals per class (seconds)
C1	2890
C2	1770
C3	4790
C4	1520
C5	1770

In particular, the signals in class C3 are distributed in the following way: 1770 signals for the subclass C3.1, 1250 for C3.2, and 1770 for C3.3. The distribution of the signals for C3.1, C3.2, and C3.3 is presented in Table 4.4.

The distribution of the signals for each class and subclass is resumed in Fig. 4.3.

Actually for class C3.1 we have collected more data. More precisely the signals belonging to "light indentation on the roll" were collected for four subsequent days for a total of 7080 signals. However, if not differently specified, we will always refer to class C3.1 as the set of 1770

Table 4.4 *Distribution of signals for the subclasses C3.1, C3.2, and C3.3*

Subclass	Amount of signals per subclass (seconds)
C3.1	1770
C3.2	1250
C3.3	1770

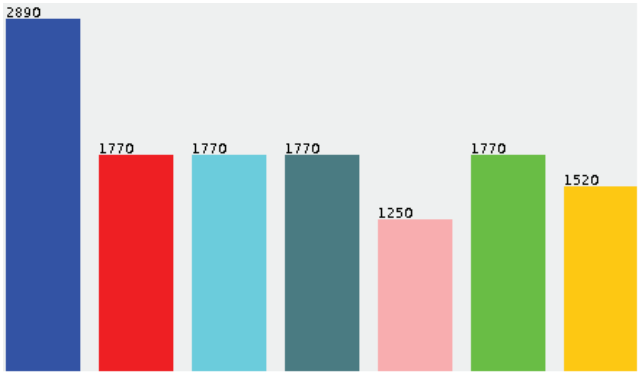


Figure 4.3 *Classes and subclasses distribution in the order C1, C2, C3.1, C3.3, C3.2, C4, and C5.*

signals belonging to the first day of collection of this type of fault. All the other data related to the other three days of collection will be used during a prognosis study described in Chap. 6. In particular for each of these four days of collection we have 1770 samples. Thus calling these other subclasses C3.1.1, C3.1.2, C3.1.3, and C3.1.4, their distribution can be represented as in Table 4.5.

4.2 Environment and software used

In the present work among other types of software and environments, the Matlab environment (see http://www.mathworks.com/products/matlab/ for more information) as well as the PRTools package [33] (see http://www.prtools.org/ for more information) have been widely used.

This work has also brought to the development of several functions

Table 4.5 *Distribution of the signals for the subclasses C3.1.1, C3.1.2, C3.1.3, and C3.1.4*

Subclass	Amount of signals per subclass (seconds)
C3.1.1	1770
C3.1.2	1770
C3.1.3	1770
C3.1.4	1770

to implement many pattern recognition functions. A brief description about the Matlab environment and the PRTools package can also be found in [138].

Chapter 5

Classification and diagnosis of rolling element bearings

It is not because things are difficult
that we do not dare,
it is because we do not dare
that they are difficult.

- L.A. Seneca -

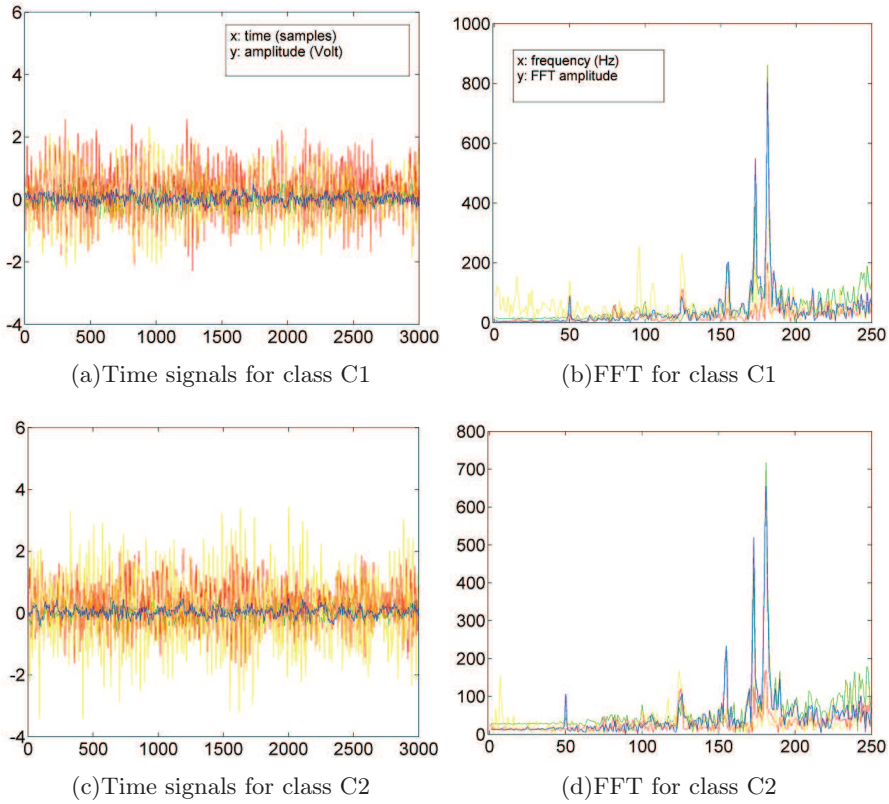
In this chapter we present the obtained results as well as the techniques and the methodologies proposed to deal with the bearing classification and diagnostic issue.

In particular in this chapter we present the performed experiments aimed at achieving the following objectives: given a mechanical object containing rolling bearings,

- to detect the presence of a defect,
- to recognize the specific type of defect,
- to recognize the severity level of the defect.

The experiments, performed on the vibration signals represented in the frequency domain, will show that the proposed classification methods are highly sensitive to different types and severity levels of the defects. The data set used in these experiments is described in Chap. 4.

To deal with the diagnostic issue, we consider the problem as a classification problem, adopting two statistical classifiers, namely the



(a)Time signals for class C1 (b)FFT for class C1

(c)Time signals for class C2 (d)FFT for class C2

Figure 5.1 *Examples of time signals and corresponding FFTs.*

LDC and the QDC, and MLP neural networks. In particular, we use LDC and QDC (with the *regularization parameter* fix to 0) to perform both feature selection and classification, whereas MLP neural networks perform classification of signals represented by means of the features selected by LDC and QDC.

Finally, to solve particularly difficult classification problems, we adopt classifier fusion.

We work in the frequency domain by transforming the time signals by using the Fast Fourier Transform (FFT). Unlike the classical approach, which identifies specific characteristic frequencies associated with given defects, we try to find out the frequencies able to discriminate among the different defects taken into consideration. Fig. 5.1 shows an example of the time signals and the corresponding FFT for classes C1 and C2.

We consider the frequency interval [1,250] Hz, sampled every 1 Hz. Therefore, each signal is represented by 250 frequency samples. As there are four accelerometers the total number of frequency samples for each element to be analyzed is $250 \times 4 = 1000$. In other words, each signal is represented in $\mathbb{R}^{1000}$. The 1000 frequency samples (referred to as *features* in the following) are obtained by concatenating the four groups of 250 frequency samples (i.e., features) relative to the four accelerometers.

As a final remark, we point out that for each experiment the data have been balanced using a random technique so that each class involved in the experiment contains the same number of samples as the least numerous one. Then the training set has been built by randomly choosing 70% of the total data, while the remaining data have been used as the test set. For the sake of clarity the set of all the damaged bearings will be referred to as class C6.

5.1 First series of experiments

5.1.1 Introduction to the first series of experiments

The proposed classification technique considers all the aspects of classification: feature selection, different base classifiers (two statistical classifiers, namely LDC and QDC, and MLP neural networks) and classifier fusion. The experiments, performed on the vibration signals represented in the frequency domain, have shown that the proposed classification method is highly sensitive to different types of defects and to different severity degrees of the defects.

5.1.2 Classification of C1 and C6

The goal of these experiments is to classify the signals into two classes: faultless bearings (C1) and damaged bearings (C6 = {C2, C3, C4, C5}).

Since each signal is represented in $\mathbb{R}^{1000}$, we need to decrease the space dimension. To this aim, we first divide the frequency interval [1,250] Hz into five sub-intervals consisting, respectively, of the first 50 frequencies, the second 50 frequencies, etc. In each sub-interval, each signal is represented by 200 features obtained by concatenating the four groups of 50 features associated with the four accelerometers (Fig. 5.2).

For each sub-interval, we look for the best *discriminating frequencies* (DFs), i.e., the frequencies that are able to provide the best accuracy

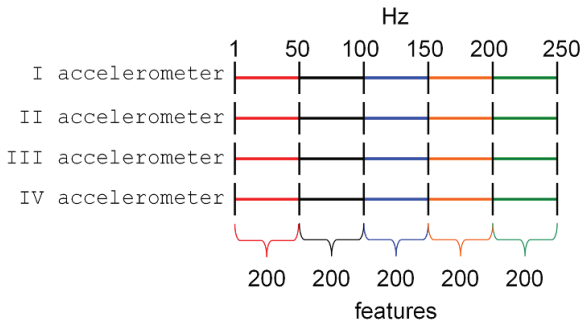


Figure 5.2 *Organization of the features considering the four accelerometers and the five frequency ranges.*

when used to represent the signals to be classified. In this way, besides decreasing the space dimension, we also identify the most significant frequency (sub-)interval for classification purposes. This step is performed using the forward feature selection (FFS). We chose to use FFS because it is a reasonable compromise between exhaustive search and random search. We adopted LDC and QDC to perform both feature selection and classification of the signals represented through the selected features. This choice stems from the fact that LDC and QDC are fast trainable classifiers.

We use 5 LDCs and 5 QDCs: each LDC/QDC works on a particular range of frequency, namely, the range [1,50] Hz, the range [51,100] Hz, etc. We experimentally verified that each classifier achieves the maximum classification accuracy with less than 200 features. The typical situation is represented in Fig. 5.3: we can notice that the accuracy increases with the number of features up to a point in which the accuracy remains almost constant and eventually decreases reaching a value that is equal to $1/n$, with n being the number of classes (we recall that we work with balanced classes). The value $1/n$ is considered the lowest accuracy that can be obtained, as, in the case the accuracy goes below this value, we could simply decide to classify all the elements as belonging to one class to increase again the accuracy up to $1/n$. This is the reason why we consider $1/n$ the lowest obtainable accuracy.

Considering all the 200 features and repeating the experiment 10 times, LDC and QDC have both selected as the best frequency range for this classification problem the fourth range, i.e., the range [151,200] Hz

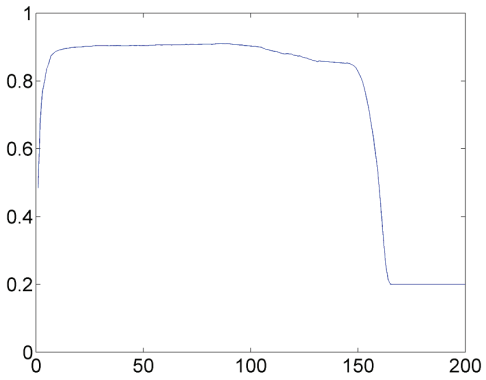


Figure 5.3 *Typical curve representing the classification accuracy (y-axis) versus the number of features (x-axis) for a five-class problem.*

as in this range we obtained the highest accuracies. In particular, we found that in the fourth frequency range LDC and QDC achieved a maximum accuracy of 99.66% with 33 features and 99.94% with 20 features, respectively.

Observing the curve that represents the classification accuracy versus the number of selected features, we noticed that just with the first 6 and 10 DFs, respectively, the two classifiers LDC and QDC achieve a performance close to the maximum in all the frequency ranges. We therefore decided to adopt only 6 and 10 DFs, respectively, to reduce the computation complexity. Indeed we can notice that each new added feature brought a negligible improvement after 6 and 10 features, respectively. In this way, the space dimension is reduced from $\mathbb{R}^{1000}$ to $\mathbb{R}^6$ and $\mathbb{R}^{10}$, respectively. In the following, we will refer to the DFs chosen to reduce the space dimension as *reduced discriminating features* (RDFs). The accuracies obtained by LDC and QDC considering only the RDFs for each frequency range are shown in Table 5.1. Table 5.2 shows the list of the RDFs for the fourth range.

Figure 5.4 shows the signals (both faultless and damaged) around the feature 181, i.e., the first DF selected by FFS in the fourth frequency range. Fig. 5.4 shows the good separation of the two classes performed using this RDF.

Table 5.1 *Classification of C1 and C6. Accuracy for LDC and QDC in the five frequency ranges (6 and 10 features respectively)*

Range	Frequency Range	Accuracy of LDC (mean over 10 trials)	Accuracy of QDC (mean over 10 trials)
1	[1,50] Hz	86.82%	85.47%
2	[51,100] Hz	86.82%	85.47%
3	[101,150] Hz	94.35%	93.94%
4	[151,200] Hz	**98.45%**	**99.76%**
5	[201,250] Hz	84.98%	89.45%

Table 5.2 *Classification of C1 and C6. List of the RDFs using the LDC and QDC classifiers for the fourth frequency range*

Classifier	RDFs
LDC	181, 23, 131, 138, 39, 32
QDC	181, 23, 131, 31, 118, 44, 5, 187, 190, 8

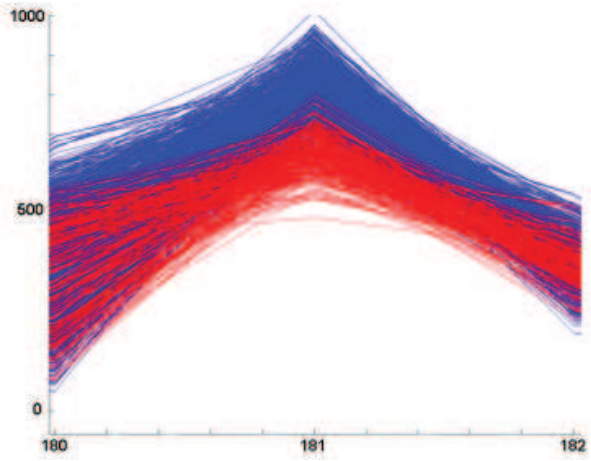


Figure 5.4 *Faultless (blue) and damaged signals (red) around the feature 181.*

5.1.3 Classification of C1, C3.1, C3.2, and C3.3

The goal of these experiments is to classify the signals into four classes C1, C3.1, C3.2, and C3.3. These experiments aim to distinguish be-

tween faultless and damaged bearings, and to recognize the different levels of severity of the same type of damage.

Repeating the experiment 10 times, once again, both LDC and QDC classifiers, using the FFS algorithm, select as the best range for this classification problem the fourth range, i.e., the range [151,200] Hz since in this range we succeed in obtaining the highest mean accuracies. In particular, considering all the 200 features, LDC and QDC achieved the maximum accuracy of 99.76% with 101 features and 99.93% with 89 features, respectively.

In this case, we consider 10 features as RDFs for both LDC and QDC. Actually, we wish to remark that, in order to adopt a uniform approach in all experiments, and taking into account the plots of accuracy versus number of DFs, we verify that choosing 10 RDFs is a good compromise both in this case and in the following cases. Indeed increasing the number of features brings to negligible improvements of the accuracy.

Table 5.3 shows, for each frequency range, the results obtained with the first 10 features (i.e., the RDFs); the accuracy is very close to the maximum one. Table 5.4 shows the list of the 10 RDFs for the fourth frequency range.

Table 5.3 *Classification of C1, C3.1, C3.2, and C3.3. Accuracy for LDC and QDC in the five frequency ranges (10 Features)*

Range	Frequency Range	Accuracy of LDC (mean over 10 trials)	Accuracy of QDC (mean over 10 trials)
1	[1,50] Hz	94.10%	97.43%
2	[51,100] Hz	90.70%	92.40%
3	[101,150] Hz	93.07%	95.50%
4	[151,200] Hz	**99.73%**	**99.87%**
5	[201,250] Hz	89.70%	90.96%

5.1.4 Classification of C1, C2, C3, C4, and C5

The goal of these experiments is to classify the signals into five classes C1, C2, C3, C4, C5. These experiments aim to recognize the different types of damage regardless of their severity levels.

Table 5.4 *Classification of C1, C3.1, C3.2, and C3.3. List of the 10 RDFs using LDC and QDC for the fourth frequency range*

Classifier	RDFs
LDC	131, 182, 181, 31, 23, 73, 130, 32, 123, 13
QDC	131, 181, 123, 182, 73, 31, 120, 138, 48, 132

Repeating the experiment 10 times, the LDC and QDC classifiers have both selected as the best frequency range for this classification problem the fourth range. Considering all the 200 features, we find that, in the fourth frequency range, LDC achieved a maximum accuracy of 94.30% using 86 features, while QDC obtained a maximum accuracy of 95.00% using 102 features. The accuracy obtained by LDC and QDC considering only the RDFs (10 also in this case) for each frequency range is shown in Table 5.5.

Table 5.5 *Classification of C1, C2, C3, C4, and C5. Accuracy for LDC and QDC in the five frequency ranges (10 Features)*

Range	Frequency Range	Accuracy of LDC (mean over 10 trials)	Accuracy of QDC (mean over 10 trials)
1	[1,50] Hz	61.72%	64.74%
2	[51,100] Hz	63.79%	63.90%
3	[101,150] Hz	56.21%	57.41%
4	[151,200] Hz	**91.01%**	**92.41%**
5	[201,250] Hz	58.88%	61.73%

The list of the 10 RDFs and an example of the related confusion matrices are shown, respectively, in Tables 5.6-5.8. From Tables 5.7 and 5.8 we notice that the main part of the error (48.25% for the QDC classifier, which achieves the best classification accuracy) is due to the misclassification of class C3, which is often recognized as C2 and vice versa. The following experiments will allow us to understand where this error is exactly placed, in other words we will expand the class C3 in its subclasses and then we will search the subclass(es) which account for most error (we wonder if C2 is misclassified equally with all the

elements of class C3 or perhaps only, or mainly, with the elements of a subclass of class C3, i.e., C3.1, C3.2, and C3.3).

Table 5.6 *Classification of C1, C2, C3, C4, and C5. List of the 10 RDFs using LDC and QDC for the fourth frequency range*

Classifier	RDFs
LDC	181, 182, 123, 137, 77, 131, 81, 82, 131, 105
QDC	181, 182, 123, 137, 77, 81, 131, 82, 132, 26

Table 5.7 *Classification of C1, C2, C3, C4, C5. LDC confusion matrix for the test set (10 features)*

		Estimated Labels					Total
		C1	C2	C3	C4	C5	
True Labels	C1	439	8	3	0	6	456
	C2	3	375	**60**	18	0	456
	C3	1	**44**	385	26	0	456
	C4	0	34	27	395	0	456
	C5	3	2	1	0	450	456
Total		446	463	476	439	456	2280

5.1.5 Classification of C1, C2, C3.1, C3.2, C3.3, C4, and C5

The goal of these experiments is to classify the signals into seven classes C1, C2, C3.1, C3.2, C3.3, C4, C5. These experiments aim to recognize not only the different types of fault but also the different degrees of severity.

LDC and QDC, using the FFS algorithm and repeating the experiment 10 trials, achieved the maximum performance again on the fourth frequency range. When using all the 200 features, LDC and QDC achieved the maximum accuracy of 95.30% with 110 features and 97.88% with 73 features, respectively. With 10 RDFs, the LDC and QDC classifiers achieved the accuracy shown in Table 5.9. We chose 10 RDFs to keep the complexity at an acceptable level. The 10 RDFs and

Table 5.8 *Classification of C1, C2, C3, C4, C5. QDC confusion matrix for the test set (10 features)*

		Estimated Labels					Total
		C1	C2	C3	C4	C5	
True Labels	C1	445	1	2	0	7	456
	C2	0	408	**33**	15	0	456
	C3	1	**64**	347	40	0	456
	C4	2	24	10	420	0	456
	C5	1	0	1	0	454	456
Total		450	497	393	475	461	2280

the related confusion matrices are shown, respectively, in Tables 5.10-5.12.

Table 5.9 *Classification of C1, C2, C3.1, C3.2, C3.3, C4, and C5. Accuracy for LDC and QDC in the five frequency ranges*

Range	Frequency Range	Accuracy of LDC (mean over 10 trials)	Accuracy of QDC (mean over 10 trials)
1	[1,50] Hz	69.84%	74.08%
2	[51,100] Hz	65.64%	69.45%
3	[101,150] Hz	65.68%	68.30%
4	[151,200] Hz	**91.10%**	**94.38%**
5	[201,250] Hz	60.93%	63.81%

From Tables 5.11 and 5.12, we can observe that the main part of the error (48.07% for the QDC classifier) is due to the misclassification of class C3.1, which is sometimes recognized as C2 (39.10%), and vice versa (8.97%). This means that the classification system cannot distinguish correctly between indentation on the inner raceway and light indentation on the roll.

On the other hand, these experiments aimed to the classification into 7 classes, namely, C1, C2, C3.1, C3.2, C3.3, C4, and C5, resulted in a better accuracy than the experiments aimed at the classification into 5

Table 5.10 *Classification of C1, C2, C3.1, C3.2, C3.3, C4, and C5. List of the 10 RDFs using LDC and QDC for the fourth frequency range*

Classifier	RDFs
LDC	131, 123, 181, 77, 182, 137, 81, 136, 73, 82
QDC	131, 123, 181, 77, 182, 81, 82, 31, 32, 173

Table 5.11 *Classification of C1, C2, C3.1, C3.2, C3.3, C4, and C5. LDC confusion matrix for the test set (10 features)*

		Estimated Labels							Total
		C1	C2	C3.1	C3.2	C3.3	C4	C5	
True Labels	C1	354	13	2	1	0	0	5	375
	C2	2	297	**56**	0	0	20	0	375
	C3.1	0	**60**	287	4	0	24	0	375
	C3.2	1	0	2	367	0	0	5	375
	C3.3	0	0	0	0	375	0	0	375
	C4	0	20	13	0	0	342	0	375
	C5	4	1	2	13	0	0	355	375
Total		361	391	362	385	375	386	365	2625

classes, namely, C1, C2, C3, C4, and C5. This suggests that one could achieve the objective of the classification into C1, C2, C3, C4, and C5 by appropriately exploiting the current experiment (classification into C1, C2, C3.1, C3.2, C3.3, C4, and C5). More precisely, we can classify the data into seven classes and then put together C3.1, C3.2 and C3.3 to obtain C3, thus returning to the five-class problem. In this way, considering the results obtained by the QDC classifier, the confusion matrix for the five-class problem becomes the one in Table 5.13 and the accuracy becomes 94.06%. This accuracy is higher than the one obtained before for the 5-class problem (92.41%). We remark that we still use the same number of RDFs.

Table 5.12 *Classification of C1, C2, C3.1, C3.2, C3.3, C4, and C5. QDC confusion matrix for the test set (10 features)*

		Estimated Labels							Total
		C1	C2	C3.1	C3.2	C3.3	C4	C5	
True Labels	C1	369	2	1	1	0	1	1	375
	C2	1	341	**14**	0	0	18	1	375
	C3.1	0	**61**	298	0	0	16	0	375
	C3.2	0	0	0	372	0	1	2	375
	C3.3	0	0	0	0	375	0	0	375
	C4	1	8	19	0	1	346	0	375
	C5	4	1	0	2	0	0	368	375
Total		375	412	332	375	376	382	372	2625

5.1.6 Classification of C2 and C3.1

Once identified where the main part of the error is (misclassification of C2 with C3.1 and vice versa rather than C2 with C3) we tried to cope with this problem with a dedicated classifier. The goal of this series of experiments is thus to classify the signals into two classes C2 and C3.1, so as to solve the main problem met in the previous series of experiments. Repeating the experiment 10 times, once again, LDC and QDC achieved the maximum performance in the fourth frequency range. When used with 10 RDFs, the LDC and QDC classifiers achieved the accuracy shown in Table 5.14.

To improve the results obtained by the LDC and QDC classifiers, we resort to classifier fusion. More precisely, we use different classifiers and then appropriately combine their responses. We use nine classifiers (Table 5.15). The MLPs used are characterized by one hidden layer and logarithmic sigmoid transfer functions. In particular, we also introduced another method of feature selection, IFS (Individual Features Selection). The nine classifiers were combined by means of the majority rule achieving the average accuracy of 94.35% over 10 trials (Table 5.15). Table 5.16 shows an example of the related confusion matrix.

We wish to point out that the obtained accuracy is higher than that

Table 5.13 *Classification of C1, C2, C3, C4, C5. QDC confusion matrix for the test set (10 features)*

		Estimated Labels					Total
		C1	C2	C3	C4	C5	
True Labels	C1	369	2	2	1	1	375
	C2	1	341	14	18	1	375
	C3	0	61	1045	17	2	375
	C4	1	8	20	346	0	375
	C5	4	1	2	0	368	375
Total		375	413	1082	382	383	2625

Table 5.14 *Classification of C2 and C3.1. Accuracy for LDC and QDC in the five frequency ranges*

Range	Frequency Range	Accuracy of LDC (mean over 10 trials)	Accuracy of QDC (mean over 10 trials)
1	[1,50] Hz	71.84%	73.21%
2	[51,100] Hz	65.81%	68.88%
3	[101,150] Hz	68.36%	67.14%
4	[151,200] Hz	**91.85%**	**91.95%**
5	[201,250] Hz	83.00%	83.99%

of the best of the nine classifiers and furthermore, in this way, we can also significantly increase the robustness of the resulting classification system.

5.1.7 Conclusions to the first series of experiments

In this first series of experiments we have presented a method, based on classification techniques and classifier fusion, for the automatic diagnosing of defects in rolling element bearings.

The proposed method has been applied to experimental data, registered by four accelerometers, and related to four different defects with different severity levels on rolling element bearings. The method has proved to be highly sensitive both to different defects and to different

Table 5.15 *Classification of C2 and C3.1. Classifier fusion*

Classifier	Neurons in the hidden layer	Number of features	Feature selection	Accuracy (mean)
LDC	—	20	—	91.97%
LDC	—	10	—	91.85%
QDC	—	20	—	92.16%
QDC	—	10	—	91.95%
MLP	20	20	FFS (LDC)	89.52%
MLP	20	20	FFS (QDC)	90.86%
MLP	20	15	FFS (QDC)	90.19%
MLP	40	10	IFS (LDC)	89.78%
MLP	40	10	IFS (QDC)	88.88%
9-classifier combiner	—	—	—	94.35%

degrees of severity for the considered defects. We achieved an accuracy on the test set greater than 94% for all the classification cases taken into consideration, sometimes reaching almost 100% accuracy.

5.2 Second series of experiments

5.2.1 Introduction to the second series of experiments

To futher improve the accuracy obtained with the previous experiments we decide to try also a different approach regarding which is the best range of frequency to be used at the aim of the classification.

We decide to enlarge the range of frequency to be analyzed, i.e., we decided to work in the frequency range [1,300] Hz instead of the previously used frequency range [1,250] Hz. Thus, we have a total of 300 features describing each sample for each accelerometer.

The Fourier spectrum related to the faultless class considering the third accelerometer and the first 300 features is represented in Fig. 5.5.

Figure 5.6 shows the Fourier spectrum related to each type of defects and severity, considering the third accelerometer and the first 300 features.

Table 5.16 *Classification of C2 and C3.1. Classifier fusion. Confusion matrix for the test set*

		Estimated Labels		Total
		C2	C3.1	
True labels	C2	489	42	531
	C3.1	18	513	531
Total		507	555	1062

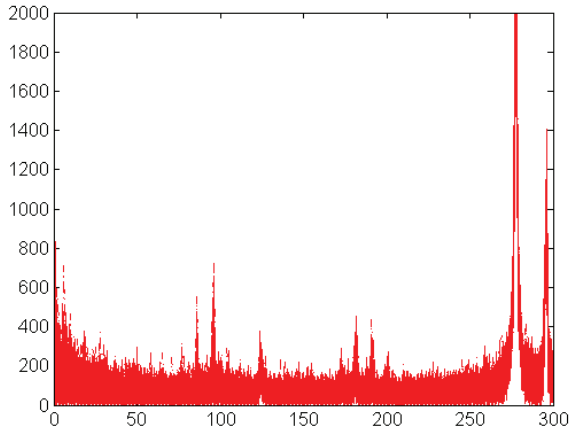
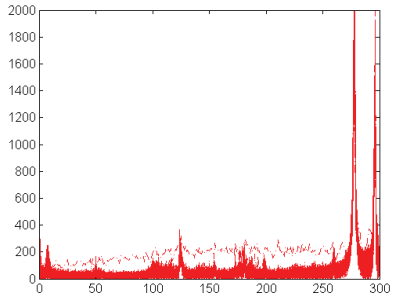


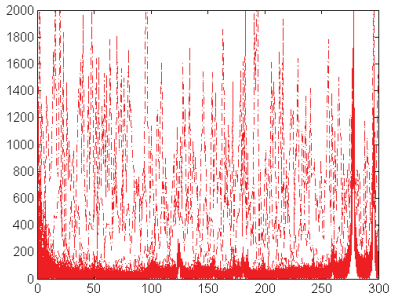
Figure 5.5 *Fourier spectrum for the signal belonging to the faultless class C1 in the frequency range [1,300] Hz for the third accelerometer.*

As the classifiers are concerned, whenever possible, simple classifiers, such as LDC and QDC, are used to perform both feature selection and classification. On the other hand, to solve particularly difficult classification problems we adopt Multi-Layer Perceptron neural networks and multi-classifier systems. In particular, three different types of methods of classifier fusion are analyzed: *maximum*, *minimum* and *average*. The proposed method shows a good sensitivity since it provides accuracy higher than 99% in all the experiments related to the recognition of different defects and different severity levels of the defects.

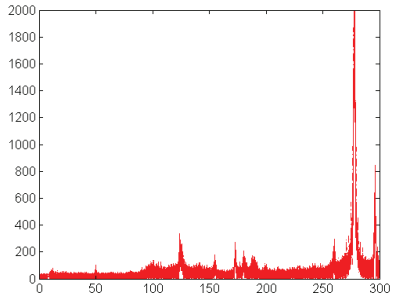
As there are four accelerometers, up to $300 \times 4 = 1200$ features (obtained by concatenating the four groups of 300 features related to



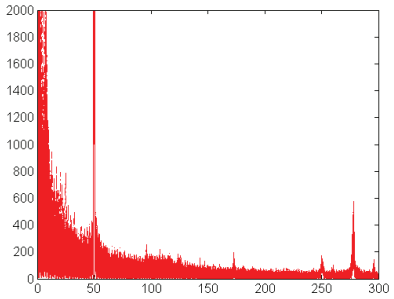
(a)Signals belonging to class C2 (indentation on the inner raceway)



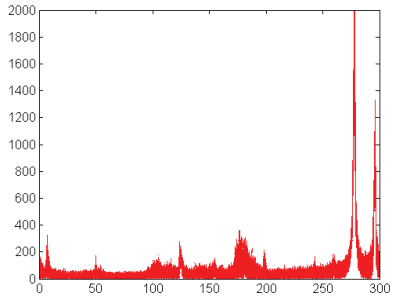
(b)Signals belonging to class C3.1 (light indentation on the roll)



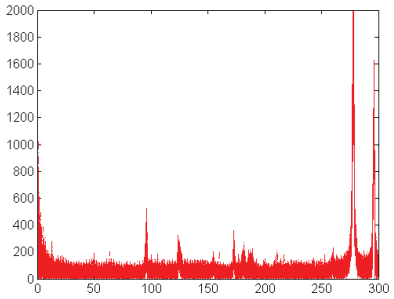
(c)Signals belonging to class C3.2 (medium indentation on the roll)



(d)Signals belonging to class C3.3 (high indentation on the roll)



(e)Signals belonging to class C4 (sandblasting of the inner raceway)



(f)Signals belonging to class C5 (unbalanced cage)

Figure 5.6 *Fourier spectra for the signals belonging to each type of defects and severity levels in the frequency range [1,300] Hz for the third accelerometer.*

the four accelerometers) can be used to represent each signal (Fig. 5.7). In other words, each signal can be represented in $\mathbb{R}^n$ with $n \leq 1200$.

To reduce the computational time, however, we decided to consider each accelerometer singularly.

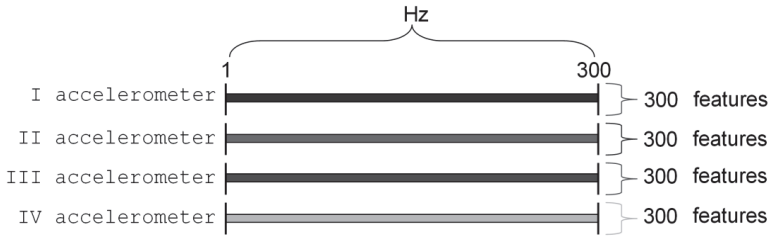


Figure 5.7 *Organization of the features considering the four accelerometers.*

5.2.2 Classification of C1 and C6

In this experiment we try to classify the signals into two classes: faultless bearings C1 and damaged bearings C6 (C2, C3, C4, C5). As previously stated, the set of all damaged bearings will be referred to as class C6. We consider the four accelerometers separately from each other. For each accelerometer, we look for the best DFs. This step is performed again using the FFS algorithm. Also this time we adopt LDC and QDC (with the *regularization parameter* set to 0) to perform both feature selection and classification of the signals represented through the selected features.

We use one LDC and one QDC for each accelerometer.

Fixing a maximum number of features equal to 10 so as to keep the computational complexity at a low level, we repeated the two-class experiment 10 times for each accelerometer.

Then, for each accelerometer and each classifier, we selected the best m ($m \leq 10$) features that are exactly the same (and in the same order) for all the trials. We will refer to these feature as *stable features* (SFs). Then using the SFs we computed the accuracy over 10 more trials. We then compared the four accelerometers based on the classification accuracy, expressed in the form (mean $\pm$ standard deviation) (Table 5.17). Please note that N.A. stands for "Not Applicable", which means that, in the specific case, no SFs have been selected.

From Table 5.17, we can notice that the QDC classifier applied to

Table 5.17 *Classification of C1 and C6*

Acc.	LDC		QDC	
	Number of SFs	Accuracy (Mean±Std.Dev.)	Number of SFs	Accuracy (Mean±Std.Dev.)
1	1	(96.04±0.97)%	1	(98.16±0.42)%
2	1	(85.43±0.68)%	0	N.A.
3	1	(94.77±0.43)%	**2**	**(99.69±0.14)%**
4	1	(92.21±0.68)%	2	(96.63±0.25)%

the third accelerometer's features achieves the best accuracy with a very small standard deviation. Actually, with just two features we obtain a good trade-off between accuracy and complexity. Table 5.18 shows the two stable features for the third accelerometer, listed in the order in which the FFS algorithm found them.

Table 5.18 *Classification of C1 and C6. List of the SFs*

QDC, Accelerometer 3
277, 296

With the chosen SFs we obtain a very high accuracy (99.69%) with a very low standard deviation (0.14%). This means that possible additional features, changing during the experiments, bring a negligible improvement. Thus, considering only two features we succeed in reducing the complexity to an optimal level from $\mathbb{R}^{1200}$ to $\mathbb{R}^2$. Fig. 5.8 shows the separation of the two classes performed by QDC using the stable features 277 and 296 of the third accelerometer.

In all our experiments we decided, based on heuristic reasoning, to consider "good" an accuracy of at least 99.00% and a standard deviation close to 0.10%. In particular, such a small range of acceptance for the standard deviation is justified by the fact that we want not only a good classifier (high accuracy) but also a stable classifier (low standard deviation). Taking these two thresholds into account, we can affirm that, for this classification problem, the choice of the two stable features is an optimal choice and thus we will not investigate further this

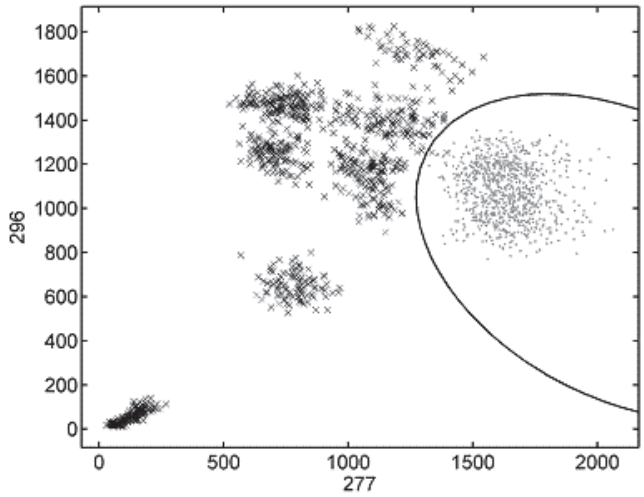


Figure 5.8 *Separation of C1 (gray dots) and C6 (black crosses) for the test set performed by QDC using the SFs of the third accelerometer.*

classification problem.

5.2.3 Classification of C1, C2, C3, C4, and C5

The goal of this experiment is to classify the signals into five classes C1, C2, C3, C4, and C5 so as to recognize the different types of damage regardless of their severity levels. Adopting the same criterion as before, i.e., repeating the experiment 10 times for each accelerometer, using the LDC and QDC classifiers, we identify, as the most promising accelerometers for this classification problem, the second and the third accelerometers (Table 5.19).

For both the second and third accelerometers, the best performance is obtained using the LDC classifier. Table 5.20 shows the list of the SFs for both the second and third accelerometers. From Table 5.19, we can see that for both accelerometers, the fixed accuracy thresholds are not respected. Thus we try to improve performance making use of the combination-level approach to building classifier ensembles, i.e., we use the fusion of classifiers.

Considering the obtained results, we decide to fuse an LDC applied to the features 296, 295 and 277 of the second accelerometer and an LDC applied to the features 277, 296 and 96 of the third accelerometer.

Table 5.19 *Classification of C1, C2, C3, C4, and C5*

Acc.	LDC		QDC	
	Number of SFs	Accuracy (Mean±Std.Dev.)	Number of SFs	Accuracy (Mean±Std.Dev.)
1	0	N.A.	0	N.A.
2	**3**	**(98.26±0.27)%**	2	(96.66±0.28)%
3	**3**	**(96.87±0.36)%**	2	(94.92±0.38)%
4	3	(95.85±0.58)%	3	(96.69±0.37)%

Table 5.20 *Classification of C1, C2, C3, C4, and C5. List of the SFs*

LDC, Accelerometer 2	LDC, Accelerometer 3
296, 295, 277	277, 296, 96

To combine the output of each classifier we used three different methods: maximum (*max*), minimum (*min*) and average (*mean*), and then we compared the results. Besides we associate different importance with each classifier. Actually we consider three types of importance: the first considers all the classifiers with the same importance, the second associates a different importance (weight) to each classifier proportional to the training accuracy before applying the combinational operators (*max*, *min* and *mean*), the third associates a different importance (weight) to each classifier proportional to the value $log(acc_i/(1-acc_i))$ (as suggested in [73]), where acc_i is the accuracy on the training set for the classifier i, before applying the combiners. The best results are obtained with the third choice. The results over 10 more trials are shown in Table 5.21.

From Table 5.21 we can see that no combiner respects both the fixed accuracy thresholds. For this reason we resort to the use of MLPs that are more complex classifiers compared to the statistical ones.

In this case we decided to combine four classifiers, namely, the LDC applied to the features 296, 295 and 277 of the second accelerometer, the LDC applied to the features 277, 296 and 96 of the third accelerometer, and two more classifiers consisting of two MLPs applied, respectively,

Table 5.21 *Classification of C1, C2, C3, C4, and C5. Classifiers and SFs used in the classifier fusion and related accuracy*

Classifier	Accelerometer, (Features)	Accuracy Mean±Std.Dev.
LDC	2, (296,295,277)	(98.26±0.27)%
LDC	3, (277,296,96)	(96.87±0.36)%
Combiner (*max*)	—	(98.88±0.16)%
Combiner (*min*)	—	(99.47±0.21)%
Combiner (*mean*)	—	(99.10±0.16)%

to the features 296, 295 and 277 of the second accelerometer and to the features 277, 296 and 96 of the third accelerometer. The used MLPs have one hidden layer and all neurons are characterized by a logarithmic sigmoid transfer function. We try different numbers of hidden neurons for each MLP, i.e., 10, 15, 20, 25, 30 and different number of epochs, i.e., 600, 800, 1000, 1200 and 1400. The best results were obtained by the use of 20 hidden neurons and 1200 epochs for each MLP. Table 5.22 shows the results obtained over 10 trials using 20 hidden neurons and 1200 epochs for each MLP. Besides we associate a different importance with each classifier inside the fusion of classifiers. Actually we try again the three types of importance mentioned above. Also in this case the best results were obtained using the weights $log(acc_i/(1-acc_i))$. From Table 5.22 we can notice that both the combiners *min* and *mean* respect the fixed accuracy thresholds, so they can be considered optimal choices for this classification problem.

5.2.4 Classification of C3.1, C3.2, and C3.3

The goal of this experiment is to classify the signals belonging to class C3 into three sub-classes C3.1, C3.2, and C3.3. This experiment aims to distinguish among the different levels of severity of the same type of damage.

Performing the FFS algorithm with LDC and QDC classifiers applied to the features of each accelerometer, we identify, over 10 trials, as the optimum classifier, the LDC applied to the features of the second accelerometer (Table 5.23). Table 5.24 shows the SFs for this accelerom-

Table 5.22 *Classification of C1, C2, C3, C4, and C5. Classifiers and SFs used in the classifier fusion and related accuracy*

Classifier	Hidden neurons	Accelerometer, (Features)	Accuracy Mean±Std.Dev.
LDC	—	2, (296,295,277)	(98.26±0.27)%
LDC	—	3, (277,296,96)	(96.87±0.36)%
MLP	20	2, (296,295,277)	(98.12±0.30)%
MLP	20	3, (277,296,96)	(98.09±0.27)%
Combiner (*max*)	—	—	(98.93±0.15)%
Combiner (*min*)	—	—	**(99.42±0.08)%**
Combiner (*mean*)	—	—	**(99.32±0.09)%**

eter. Fig. 5.9 shows the separation of the three classes performed by LDC using the SFs.

Table 5.23 *Classification of C3.1, C3.2, and C3.3*

Acc.	LDC		QDC	
	Number of SFs	Accuracy (Mean±Std.Dev.)	Number of SFs	Accuracy (Mean±Std.Dev.)
1	1	(92.47±1.17)%	1	(95.50±1.02)%
2	2	**(100.00±0.00)%**	1	(93.49±0.55)%
3	1	(99.97±0.10)%	0	N.A.
4	2	(98.90±0.16)%	1	(96.08±0.42)%

5.2.5 Classification of C1, C2, C3.1, C3.2, C3.3, C4, and C5

This experiment aims to classify the signals into seven classes C1, C2, C3.1, C3.2, C3.3, C4, and C5 in order to recognize both the different types of fault and the different levels of severity.

The previous three experiments can be considered *base experiments* as they classify the data in faultless and damaged ones, in different types of defects, and in different types of severity levels, respectively.

Table 5.24 *Classification of C3.1, C3.2, and C3.3. List of the SFs*

LDC, Accelerometer 2
296, 173

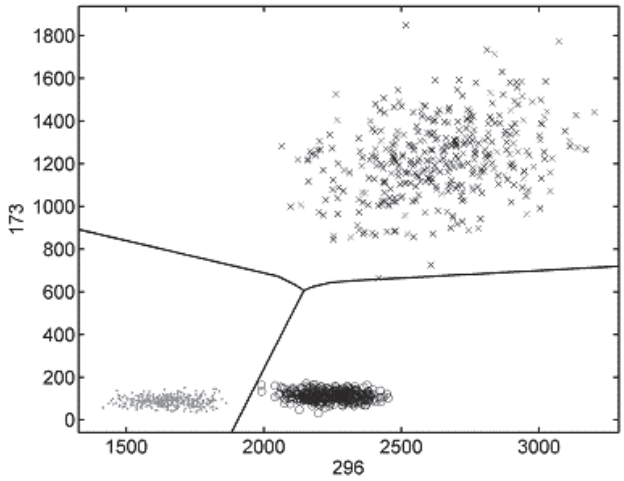


Figure 5.9 *Separation of C3.1 (gray dots), C3.2 (black circles), and C3.3 (gray crosses) for the test set performed by LDC.*

This experiment, instead, can be considered a non-base classification problem since it tries to classify the signals considering both the types of defects and the levels of severity. For this reason we can see this experiment like the fusion of the last three experiments.

We also wish to use this experiment to prove the sensitivity and the reliability of the proposed method adopted in the last three experiments. In practice, we start using statistical classifiers, then, if the accuracy thresholds are not respected, first we resort to the combination of these statistical classifiers, then we adopt fusion of the statistical classifiers with neural networks.

Let us call optimal configurations the three sets of SFs and classifiers that were chosen in the last three experiments (base experiments) to solve the correspondent classification problems. Table 5.25 shows the optimal configurations found in the last three experiments (which use 300 features for each accelerometer), respectively.

Table 5.25 *Optimal configuration for the base experiments*

Base experiments	Optimal configuration	
	Classifier	Accelerometer (SFs)
1^{st} (C1, C6)	QDC	3 (277, 296)
2^{nd} (C1, C2, C3, C4, C5)	LDC	2 (296, 295, 277)
	LDC	3 (277, 296, 96)
3^{rd} (C3.1, C3.2, C3.3)	LDC	2 (296, 173)

To prove the sensitivity and the reliability of the proposed method we decide to use only the configurations chosen in the base experiments and to fuse them to solve the current classification problem. Actually, regarding the 5-class classification, we decide to use only one LDC classifier, instead of two, applied to the union of the features, i.e., 296, 295, 277 and 173 shown in Table 5.25. Thus the classifiers used in this classification problem are the ones listed in Table 5.26.

Table 5.26 *Classification of C1, C2, C3, C4, and C5. Classifiers and SFs used in the classifier fusion and related accuracy*

Classifier	Accelerometer, (Features)	Accuracy Mean±Std.Dev.
QDC	3, (277, 296)	(98.10±0.10)%
LDC	2, (296, 295, 277, 173)	(98.45±0.08)%
LDC	3, (277,296,96)	(98.59±0.13)%
Combiner (*max*)	—	(99.30±0.26)%
Combiner (*min*)	—	(99.04±0.16)%
Combiner (*mean*)	—	(98.85±0.16)%

Besides, also in this case, we associate a different importance with each classifier inside the fusion of classifiers. Actually we consider the three types of importance mentioned above. The best results were obtained again using the weights $log(acc_i/(1-acc_i))$. The results obtained fusing the classifiers with the three combiners *max*, *min* and *mean* are

shown in Table 5.26. From Table 5.26 we can see that no combiner respects both the two fixed accuracy thresholds. For this reason we resort to the use of MLPs.

In this case we decide to combine six classifiers, that are the three mentioned above, i.e., QDC applied to the features 277 and 296 of the third accelerometer, LDC applied to the features 296, 295, 277, and 173 of the second accelerometer, and an LDC applied to the features 277, 296, and 96 of the third accelerometer, and other three classifiers consisting of three MLPs applied, respectively, to the features 277 and 296 of the third accelerometer, to the features 296, 295, 277, and 173 of the second accelerometer, and to the features 277, 296 and 96 of the third accelerometer. The used MLPs have one hidden layer and the neurons are characterized by a logarithmic sigmoid transfer (activation) function. We try different numbers of hidden neurons for each MLP, i.e., 10, 15, 20, 25, 30, and different number of epochs, i.e., 600, 800, 1000, 1200 and 1400. The best results are obtained by the use of 20 hidden neurons and 1200 epochs for each MLP. Again, we study the behavior of the combiners changing the importance associated with each classifier. Actually we consider the three types of importance mentioned above and the best results were obtained using the weights $log(acc_i/(1 - acc_i))$.

Table 5.27 shows the results obtained over 10 trials using 20 hidden neurons and 1200 epochs for each MLP. As we can see, the combiners *max* and *mean* respect the fixed accuracy thresholds.

We wish to remark that, in this classification problem, we have not performed any feature selection but we have based our classification only on the information collected with the previous base experiments. This means that the proposed method can be adopted as a methodology to develop an automatic system for bearing fault detection and recognition.

5.2.6 Conclusions to the second series of experiments

In this second series of experiments we have tested a proposed automatic method, based on soft computing classification techniques and classifier fusion, for diagnosing defects of rolling element bearings.

The method has been applied to data, collected by four accelerometers, pertinent to four different defects on rolling bearings with three different levels of severity. The method has proved to be highly sen-

Table 5.27 *Classification of C1, C2, C3.1, C3.2, C3.3, C4, and C5. Classifiers and SFs used in the classifier fusion and related accuracy*

Classifier	Hidden neurons	Accelerometer, (Features)	Accuracy Mean±Std.Dev.
QDC	—	3, (277,296)	(98.10±0.10)%
LDC	—	2, (296,295,277,173)	(98.54±0.08)%
LDC	—	3, (277,296,96)	(98.59±0.13)%
MLP	20	3, (277,296)	(89.39±2.70)%
MLP	20	2, (296,295,277,173)	(96.16±2.37)%
MLP	20	3, (277,296,96)	(98.34±0.16)%
Comb. (*max*)	—	—	**(99.27±0.13)%**
Comb. (*min*)	—	—	(98.34±1.80)%
Comb. (*mean*)	—	—	**(99.06±0.12)%**

sitive both to different defects and to different degrees of severity for the considered defects. We achieved an accuracy on the test set greater than 99% and a standard deviation close to 0.1% in all the experiments (sometimes reaching even 100% accuracy with zero standard deviation).

Chapter 6

Time evolution analysis of rolling element bearing faults

The only thing in life
achieved without effort
is failure.

- Anonymous -

The process of detection and diagnosis of faults in a mechanical equipment deals only with the assessment of the health of the system at the current time, thus it does not provide any information regarding the remaining useful life of the equipment [123, 141].

The ability to accurately predict the remaining useful life of a machine system represents a crucial issue in the maintenance process of a system which can even improve the productivity and increase the system safety [66, 141].

An effective prognostics program gives to the maintenance engineers more time to schedule a maintenance activity to repair and to acquire replacement components before the system further decreases its "health" state [66, 141].

Nowdays, even though the expert diagnostic engineers have significant information and experience about machine failure and health states by continuously monitoring and analyzing the machine condition, in the literature, little attention has been paid to the study of the evolution of a defect, that is how a defect evolves over time if a fault component

(such as a bearing) is not repaired or substituted by a faultless one [66, 123, 141].

Actually, in this chapter we describe the performed experiments whose aim is to find out how the vibration signals coming from a faulty bearing evolve when the faulty bearing is not substituted immediately. This means, e.g., to study if the severity level of the bearing increases and, possibly, after how much time. Besides, we wish to analyze if, after a certain amount of time in which a damaged bearing continues to work, we can consider it equivalent to a bearing with the same defect but with a higher level of severity [141].

In particular in this chapter we present the performed experiments aimed at achieving the following objectives: given a mechanical object containing rolling bearings,

- to distinguish between faultless and damaged bearings,
- to recognize the severity level of the defect,
- to analyze the evolution of a particular defect comparing it with same defects with higher severity levels,
- to continue to recognize a defect as time passes.

To this aim, we have dealt with the problem as a classification problem, adopting the statistical classifier QDC, MLPs and RBFs. In particular, we use QDC to perform both feature selection and signal classification, whereas MLP and RBF neural networks only perform classification of the signals represented by means of the features selected by QDC.

To this aim we consider the data subdivided into faultless class C1 and damaged class (call it C6). The data at our disposal consists of 2890 signals for the faultless class C1 and 10100 signals for the damaged class C6. The distribution of the signals for class C1 and C6 is represented in Table 6.1. The damaged class C6 is composed by the three classes C3.1, C3.2, and C3.3.

Table 6.1 *Distribution of the signals for classes C1 and C6*

Class	Number of signals (sec)
C1	2890
C6	10100

In this series of experiments, as stated in Chap. 4, we consider the class C3.1 as composed by the signals collected for four subsequent days during which no changes were made to the system. Each day of collection refers to a specific class, i.e., C3.1.1, C3.1.2, C3.1.3, and C3.1.4.

Table 6.2 shows the distribution of the signals for the classes C3.1, C3.2, and C3.3, while Table 6.3 represents the distribution of the signals for class C3.1 along the four days of data collection.

For the sake of clarity we will also call C7 the class composed by the following classes: C3.1.1, C3.2, and C3.2.

Table 6.2 *Distribution of the signals for classes C3.1, C3.2, and C3.3*

Class	Number of signals (sec)
C3.1	7080
C3.2	1250
C3.3	1770

Table 6.3 *Distribution of the signals for classes C3.1.1, C3.1.2, C3.1.3, and C3.1.4*

Class	Number of signals (sec)
C3.1.1	1770
C3.1.2	1770
C3.1.3	1770
C3.1.4	1770

6.1 Methodology

We work in the frequency domain by transforming the signals by the Fast Fourier Transform (FFT). Unlike the classical approach, which identifies specific characteristic frequencies associated with given defects, we tried to automatically find out the frequencies able to discriminate among the different classes taken into consideration. We would like to stress that we also tried the feature extraction algorithm Prin-

cipal Component Analysis (PCA) which, however, was not able to give results as accurate as the Forward Feature Selection (FFS).

We considered the frequency interval [1,300] Hz, sampled every 1 Hz, since this was proved to be more useful then the interval [1,250] Hz. Since from the previous experiments we noticed that the second and third accelerometers were the one which provide better results we decide to focus our attention only on these accelerometer. For the sake of clarity, these accelerometers will be referred as the first and the second accelerometer in this chapter. Thus since we consider only two accelerometers, up to $300 \times 2 = 600$ features, obtained by concatenating the two groups of 300 features related to the two accelerometers, can be used to represent each signal (Fig. 6.1). In other words, each signal can be represented in $\mathbb{R}^n$ with $n \leq 600$.

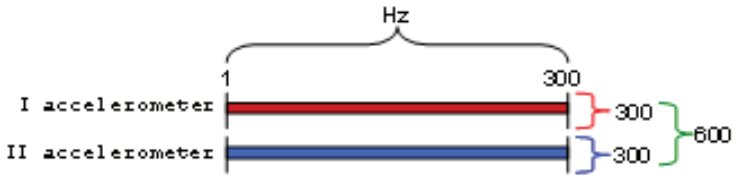


Figure 6.1 *Organization of the features considering the two accelerometers.*

Again, for each experiment we balance the data using a random technique so that each class involved in the experiments contains the same number of samples as the least numerous one.

Finally, the training and test sets have been built using the H-method by randomly choosing 70% of the total data as training set and the remaining 30% as test set. We chose this algorithm since it is very fast, very simple, it has empirically been shown to be one of the most effective resampling methods, and we can decide exactly how many elements should be removed [72, 141, 142].

In the following we will also indicate the required time for each experiment with reference to a computer machine Pentium Dual-Core 2.50 GHz, with 4 GB of RAM, for a 32 bit Matlab environment.

First of all we need to choose how to represent the signals. To this aim, we take two different requirements into account: the memory necessary for signal representation (number of features) and the time needed to perform the classification. Both requirements should be kept

as low as possible.

As our signals are represented with 600 features (300 related to the first accelerometer and 300 related to the second accelerometer) we need to decrease the space dimension in order to reduce the complexity of the classification problem. In fact, any feature set may include not only useful features but also irrelevant or redundant features. Besides the use of all the features of a feature set can make the classification process slower and the classification accuracy lower. Thus to simultaneously improve the classification accuracy and reduce the computational burden of the classifier, few features that are able to characterize the bearing status need to be selected from the original feature set. For this reason, the first step to perform is a feature selection. To this aim we choose to use the FFS algorithm. Thus, with the FFS, we look for the best *discriminating frequencies* DFs, i.e., the features that are able to provide the best accuracy when used to represent the signals to be classified.

As regards the classifier choice, we adopt QDC to perform both feature selection and classification of the signals represented through the selected features.

6.2 Experiments and Results

6.2.1 Classification of C1 and C7

This experiment aims to classify the signals into two classes: faultless bearings (C1) and damaged bearings (C6), in order to check whether we are able to identify a defect, whatever its severity is, as soon as it appears. We therefore decide to consider just C7 (i.e., the union of C3.1.1, C3.2 and C3.3) and not the whole class C3, which includes also the samples collected in the days beyond the first. Table 6.4 shows the classes involved in the experiment. Please note that the acronyms TR and TS identify the classes used in the training process and in the test process respectively.

Performing the FFS with the QDC classifier, we obtain the DFs in the order in which FFS selected them and the classification accuracy shown in Table 6.5. The accuracy represents the average over 30 trials using the selected DFs. From Table 6.5, we can notice that we found the optimum classifier for this classification problem, reducing the complexity of the problem to an optimal level from $\mathbb{R}^{600}$ to $\mathbb{R}^2$.

Table 6.4 *Classification of C1 and C7. Training and test sets*

C1	C6					
	C3.1				C3.2	C3.3
	C3.1.1	C3.1.2	C3.1.3	C3.1.4		
TR-TS	TR-TS				TR-TS	TR-TS

Table 6.5 *Classification of C1 and C7. Accuracy and DFs*

DFs	Accuracy (Mean±Std.Dev)
577, 265	(100.00±0.00)%

The separation of the two classes C1 and C7 performed by QDC is shown in Fig. 6.2, while Table 6.6 shows the time (msec) required to perform the feature selection (to select 2 features), to train and test the QDC classifier.

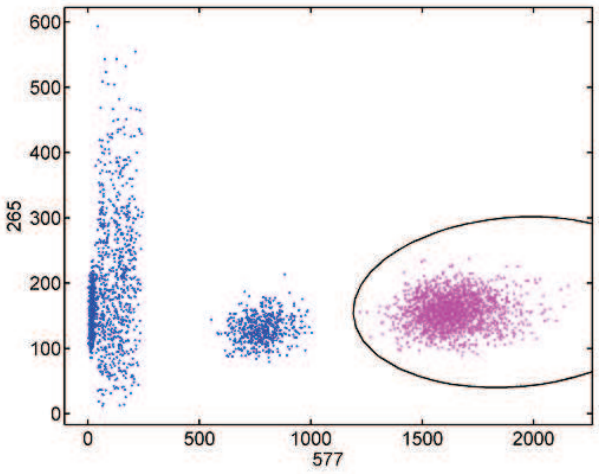


Figure 6.2 *Separation of C1 (red dots) and C7 (blue dots) for the test set performed by QDC and the two DFs.*

6.2.2 Classification of C1 and C6

The goal of the second experiment is to classify the signals into two classes C1 and C6 so as to recognize faultless and damaged bearings.

Table 6.6 *Classification of C1 and C7. Mean computational time*

	Feature Selection	Training	Testing
Computational time	39.46 msec	12.22 msec	16.72 msec

Unlike the previous experiment, in this case we take into account all the samples of C3.1 pertinent to the first, second, third and fourth days of collection. Thus we are interested in checking if we can identify a defect not only on its appearance but also later. In practice, we aim to detect how the evolution of a defect over time can affect the corresponding vibration signals. The classes involved in the experiment are shown in Table 6.7.

Table 6.7 *Classification of C1 and C6. Training and test sets*

C1	C6					
	C3.1				C3.2	C3.3
	C3.1.1	C3.1.2	C3.1.3	C3.1.4		
TR-TS	TR-TS	TR-TS	TR-TS	TR-TS	TR-TS	TR-TS

Table 6.8 shows the DFs in the order in which FFS selected them and the classification accuracy obtained repeating 30 times the experiment using only these DFs.

Table 6.8 *Classification of C1 and C6. Accuracy and DFs*

DFs	Accuracy (Mean±Std.Dev)
577, 296, 188	(99.94±0.04)%

From Table 6.8, we can notice that we obtained a very high accuracy, 99.94%, with a very small standard deviation, 0.04%. Thus, with only three DFs, we can reduce the complexity to an optimal level from $\mathbb{R}^{600}$ to $\mathbb{R}^3$. Table 6.9 shows the confusion matrix resulting from the average of all the confusion matrix over the 30 trials for this classification problem while Table 6.10 specifies the computational time required to perform the feature selection, and the time to train and test the QDC classifier.

Table 6.9 *Classification of C1 and C6. Confusion matrix for the test set using the three DFs and the QDC classifier*

		Estimated Labels		Total
		C1	C7	
True labels	C1	867	0	867
	C6	2	865	867
Total		869	865	1734

Table 6.10 *Classification of C1 and C6. Mean computational time*

	Feature Selection	Training	Testing
Computational time	59.59 msec	12.77 msec	17.03 msec

6.2.3 Classification of C3.1.2, C3.1.3, and C3.1.4

From a practical point of view, with reference to a specific defect, it would be important to be able to train the classifier with the samples collected during the first day of defect occurrence and successfully recognize the samples representing the same defect over the following days. This means that we train the classifier on two classes, namely C1 and C7, and then we test it on C3.1.2, C3.1.3, and C3.1.4. We wish to verify if our classifier is able to recognize these classes as belonging to the damaged class without using these data during the training process. Table 6.11 shows the classes involved in the experiment.

Table 6.11 *Classification of C3.1.2, C3.1.3, and C3.1.4. Training and test sets*

C1	C6					
	C3.1				C3.2	C3.3
	C3.1.1	C3.1.2	C3.1.3	C3.1.4		
TR-TS	TR-TS	TS	TS	TS	TR-TS	TR-TS

Considering the DFs selected in the first experiment, i.e., 577 and 265, we achieve the accuracy and the average confusion matrix obtained

over the 30 trials shown in Table 6.12. In particular, Fig. 6.3 shows the separation of the two classes C1 and C6 performed by QDC.

Table 6.12 *Classification of C3.1.2, C3.1.3, and C3.1.4. Confusion matrix for the test set using the three DFs and the QDC classifier*

		Estimated Labels		Total	Accuracy
		Faultless	Damaged		
True labels	C3.1.2	0	1770	1770	100.00%
	C3.1.3	0	1770	1770	100.00%
	C3.1.4	227	1543	1770	87.17%
Total		227	5083	5310	95.72%

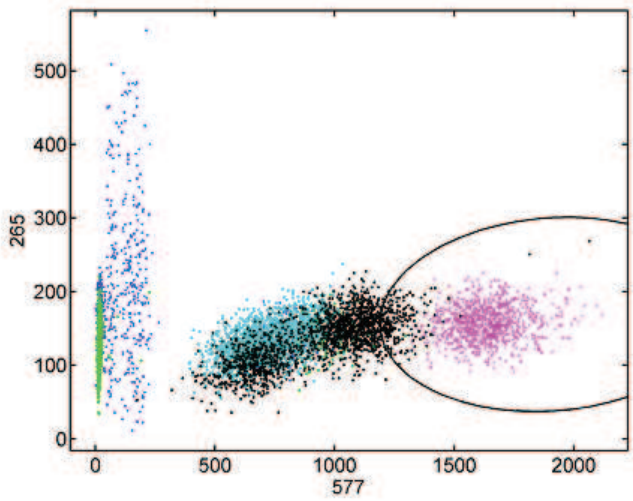
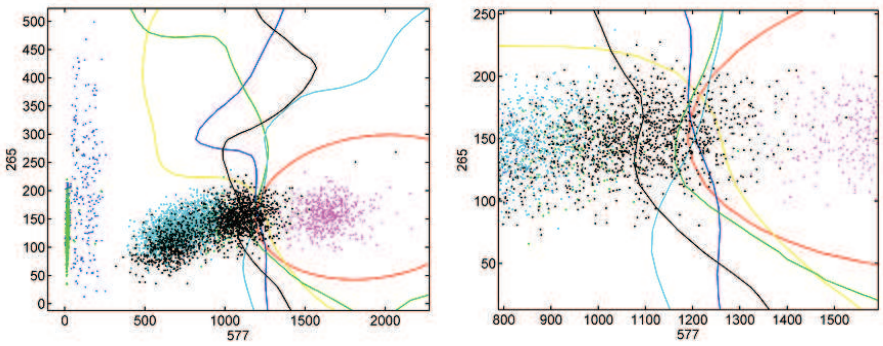


Figure 6.3 *Separation of C1 and C6. C1 (red dots), C3.1.1, C3.2, C3.3 (blue dots), C3.1.2 (green dots), C3.1.3 (cyan dots), C3.1.4 (black dots) for the test set performed by QDC and the two DFs.*

From Table 6.12 we can notice that the total accuracy, equal to 95.72%, is not satisfying. In particular, while classes C3.1.2 and C3.1.3 are perfectly classified as damaged, class C3.1.4 is not properly classified; indeed the 12.83% of the elements of class C3.1.4 is misclassified as belonging to the faultless class. It seems that as time passes the



(a) (b)

Figure 6.4 *(a) Separation of faultless and damaged for the test set performed by QDC and MLPs using two DFs. Classes: C1 (red dots), C3.1.1, C3.2, C3.3 (blue dots), C3.1.2 (green dots), C3.1.3 (cyan dots), C3.1.4 (black dots). Discriminant functions: QDC (red), MLP with 15 h.n. (green), MLP with 25 h.n. (blue), MLP with 35 h.n. (yellow), MLP with 45 h.n. (cyan), MLP with 60 h.n. (black). (b) zoom of (a).*

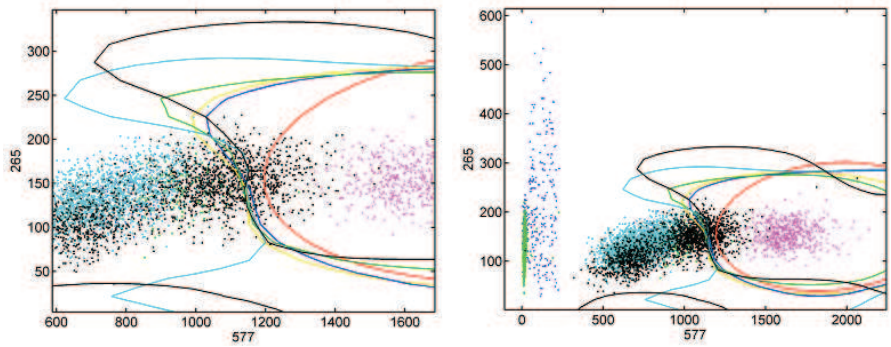
defect becomes less easily detectable. This problem may derive from the fact that we have reduced the set of features too much. Actually class C3.1.4 may differ from class C1 for other features, which we have not selected as DFs. This agrees with the fact that in the first and second experiments we found different DFs, (577, 265) and (577, 296, 188), respectively.

Thus, we try to use more complex but also more powerful classifiers such as MLP and RBF neural networks. We use MLPs with one hidden layer and neurons characterized by a logarithmic sigmoidal transfer function. We try different numbers of hidden neurons, i.e., 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, and 60. Fig. 6.4 shows the separation into faultless and damaged samples performed by QDC and some MLPs. In all the following figures, h.n. stands for hidden neurons.

We use RBFs with one hidden layer and neurons characterized by a Gaussian transfer function. We try different numbers of hidden neurons (10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60) with a spread value of 1.

The separation into faultless and damaged samples performed by QDC and some RBFs is shown in Fig. 6.5.

We can notice that neither MLPs nor RBFs achieve higher accuracy



(a) (b)

Figure 6.5 *(a) Separation of faultless and damaged for the test set performed by QDC and RBNs using two DFs. Classes: C1 (red dots), C3.1.1, C3.2, C3.3 (blue dots), C3.1.2 (green dots), C3.1.3 (cyan dots), C3.1.4 (black dots). Discriminant functions: QDC (red), RBN with 15 h.n. (yellow), RBN with 25 h.n. (blue), RBN with 35 h.n. (green), RBN with 45 h.n. (cyan), RBN with 60 h.n. (black). (b) zoom of (a).*

than that of the QDC classifier. However from Figs. 6.4 and 6.5 we can observe that, even though QDC gives the best performance, sometimes the neural networks are able to classify correctly some elements that QDC misclassifies. More precisely, all these types of classifiers do not make the same errors. This is a key feature that allows us to combine these classifiers in order to increase the whole performance [73].

The best average performances among the used MLPs and RBFs are obtained by the MLPs, in particular by the MLPs with 15, 25, 35 and 45 hidden neurons, respectively. Thus we decide to combine the QDC classifier with these four MLPs adopting the *unanimity vote*: one signal is considered as belonging to class C1 only when all these classifiers classify it as belonging to class C1. So doing, we reduce the space associated with the faultless bearings.

In this way we succeed in significantly increasing the accuracy while drastically decreasing the number of misclassifications of the elements of class C6. Table 6.13 shows the average over 30 trials of the confusion matrix considering the combination of the five classifiers.

Table 6.13 *Classification of C3.1.2, C3.1.3, and C3.1.4. Confusion matrix for the test set using the two DFs and combine the QDC classifier with four MLPs*

		Estimated Labels		Total	Accuracy
		Faultless	Damaged		
True labels	C1	1770	0	1770	100.00%
	C3.1.1	0	1770	1770	100.00%
	C3.1.2	1	1769	1770	99.94%
	C3.1.3	0	1770	1770	100.00%
	C3.1.4	141	1629	1770	92.03%
Total		1912	6938	8850	98.39%

6.2.4 Time evolution of a defect

The aim of this experiment is to analyze how a damaged bearing evolves over time. In this case we make a three-class classification, training a QDC classifier with classes C3.1.1, C3.2, and C3.3, and then testing it on classes C3.1.2, C3.1.3, and C3.1.4. In this way we wish to verify how the signals belonging to the light indentation on the roll (C3.1) evolve as time passes. Table 6.14 shows the classes involved in the experiment.

Table 6.14 *Time evolution analysis. Training and test sets*

C1	C6					
	C3.1				C3.2	C3.3
	C3.1.1	C3.1.2	C3.1.3	C3.1.4		
	TR	TS	TS	TS	TR	TR

Using the FFS we select the DFs listed in Table 6.15. Repeating the experiment 30 times we obtain 100.00% accuracy with 0.00% standard deviation as shown in Table 6.15 when classifying classes C3.1.1, C3.2 and C3.3.

The separation of the three classes (C3.1.1, C3.2 and C3.3) performed by the QDC using the two DFs (578, 350) is shown in Fig. 6.6.

Considering the DFs 578, 350, we have now to see how the classes C3.1.2, C3.1.3 and C3.1.4 are classified. The obtained results over 30

Table 6.15 *Classification of C3.1.1, C3.2, C3.3. QDC classifier. Accuracy and DFs*

DFs	Accuracy (Mean±Std.Dev)
578, 350	(100.00±0.00)%

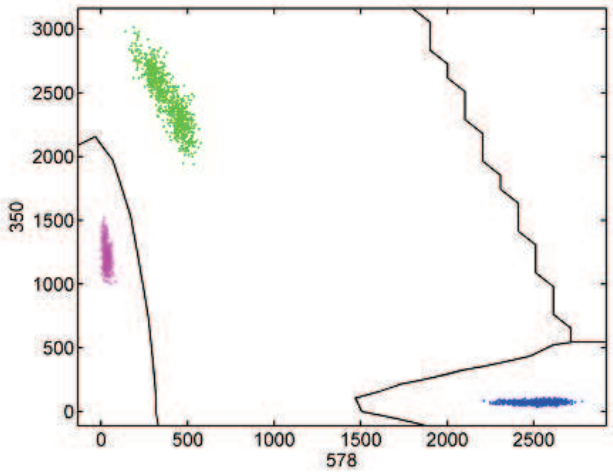


Figure 6.6 *Separation of C3.1.1 (red dots), C3.2 (blue dots) and C3.3 (green dots) for the test set performed by QDC and the two DFs.*

trials are given in Table 6.16, while Fig. 6.7 shows the separation of the three classes performed by QDC with the two DFs.

This experiment suggests that for the selected DFs some elements of C3.1.2 are misclassified with samples of class C3.2, while all the elements of classes C3.1.3 and C3.1.4 are misclassified with samples of class C3.2. This means that after one day the severity of the defect has already increased considerably.

This result is very important to define which is the time limit (deadline) within which it is necessary to substitute the faulty bearing. In this case the deadline is actually one day as, after one day, the vibrations coming from the faulty bearing (C3.1) become similar to those with higher severity (C3.2).

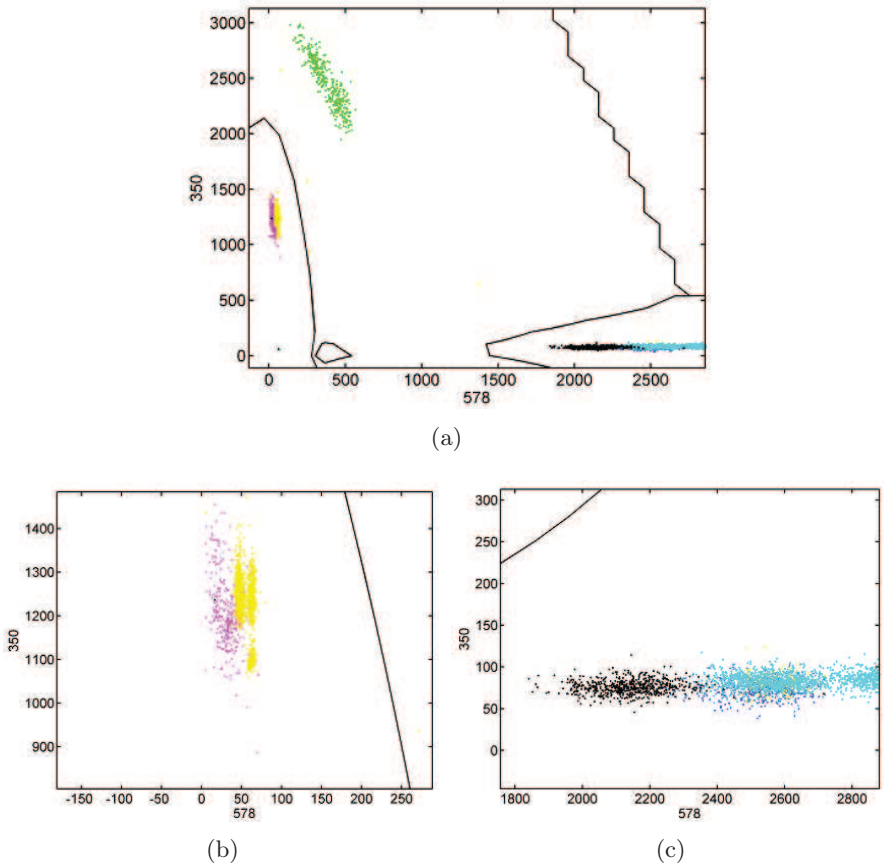


Figure 6.7 *(a) Separation of C3.1.1 (red dots), C3.1.2 (yellow dots), C3.1.3 (cyan dots), C3.1.4 (black dots), C3.2 (blue dots) and C3.3 (green dots). Training set consisting on C3.1.1, C3.2, C3.3. (b) and (c) zoom of (a).*

Table 6.16 *Classification of C3.1.2, C3.1.3, C3.1.4. Confusion matrix for the test set using the two DFs and the QDC classifier*

		Estimated Labels			Total
		C3.1.1	C3.2	C3.3	
True	C3.1.2	1628	138	4	1770
labels	C3.1.3	0	1770	0	1770
	C3.1.4	1	1769	0	1770
Total		1629	3677	4	5310

6.3 Conclusions to the prognosis issue

In this chapter we have proposed the use of classification techniques and classifier fusion to automatically detect both the presence of a defect on a rolling element bearing and its severity level. We used experimental data consisting of vibration signals represented in the frequency domain by means of the FFT, registered by two accelerometers. We took into account one defect with three types of severity.

As the data related to the lowest severity level were collected in four subsequent days, we have also performed an analysis aimed to identify how the vibration signals of a damaged bearing evolve over time. We observed that, as time passes the signals representing the least severe damage get more similar to those related to the same defect but with a higher severity level. This analysis can be profitably used to define prognostic program to detect as soon as possible any incipient defects, as well as to determine the time within which the maintenance, i.e., the substitution of the faulty bearing, should be performed before the defect gets too serious.

Chapter 7

Noise Analysis

A person who never made a mistake
never tried anything new.

- A. Einstein -

As stated in Chap. 2, many methodologies proposed in the literature are not tested on noisy data, so that several diagnostic techniques can perform well in a noise-free environment but very poorly in the presence of noise like in a real environment.

For this reason, we would like to analyze how our methodologies will perform in presence of noise, and how to improve, as much as possible, this performance.

7.1 Noise signal creation

There are two main ways to obtain signals affected by noise. The first one is to affect the machine containing the bearing with noise, then to collect the signals at all the required levels of noise, the second one is to collect the signals in a free-noise environment and then to add an artificial noise.

However, the first one requires that the machine is let working in a noisy environment (which is not necessary) which can affect the operating condition of the machine itself, while in the second way no additional collection of signals is required and thus the machine is not subjected to not necessary noise. Besides, in this way every time a new level of noise is required it is just required to artificially create a new

noise signal without requiring to use again the machine to collect new data, which can require to stop the machine which can be a problem in normal working environment where the production should be stopped as few times as possible. For these reasons we decide to artificially add a noisy signal to the free-noise signals we have already collected.

The used artificial noise signal is a white Gaussian noise (stochastic process with zero mean and unity variance), which is multiplied by an increasing positive coefficient, called noise level, to increase its power.

To create a data set affected by noise we add, in the time domain, the Gaussian noise signals to the signals not affected by noise. Then we compute the signal-to-noise ratio (SNR) as the ratio between the average power of the signals of all classes and the noise power. More formally, the power of the i-th signal s_i is computed as in eq. 7.1:

$$P_{s_i} = \frac{1}{N_{sam}} \sum_{j=1}^{N_{sam}} s_{i_j}^2 \tag{7.1}$$

where N_{sam} represents the total number of samples for each signal, and s_{i_j} is the j-th temporal sample of the signal s_i. The average power of the signals of all classes is described by eq. 7.2:

$$\overline{P}_{s_i} = \frac{1}{N_{sig}} \sum_{i=1}^{N_{sig}} P_{s_i} \tag{7.2}$$

where N_{sig} is the total amount of signals in all the classes.

The process of noise is repeated for each sample of the signals, each time generating a new process of random noise n_k, $k = 1, ..., N_{sam}$.

We create ten data sets each of which obtained by adding a different increasing level of noise to the data set of noise-free signals. To achieve this we multiply each stochastic noise by an increasing positive integer NL. In the experiments, we use ten noise levels NL_h, with $h = 1, \ldots, 10$:

$$NL_h = \{5, 10, 15, 20, 25, 30, 40, 60, 80, 100\}$$

The k-th extracted Gaussian noise gives origin to ten different noise

signals as expressed in eq. 7.3:

$$n_{kh} = NL_h \cdot n_k \qquad h = 1, \ldots, 10, \tag{7.3}$$

consequently, the power of the generic noise signal n_k becomes as described in eq. 7.4:

$$P_h = \frac{1}{N_{sam}} \sum_{k=1}^{N_{sam}} n_{kn}^2. \tag{7.4}$$

Therefore, for each noise level NL_h, we compute the SNR as expressed in eq. 7.5:

$$SNR_h = 10 \log_{10} \frac{\overline{P}_s}{P_h}. \tag{7.5}$$

7.2 First assessment of the robustness to the noise

7.2.1 Introduction to the experiments for the first assessment of the robustness to noise

In this subsection we present a method, based on classification techniques, for the automatic detection and diagnosis of defects of rolling element bearings. We use the data set described in Chap. 4. We also assess the degree of robustness of our method to noise by analyzing how the classification performance varies at the variation of the signal-to-noise ratio.

This series of experiments aims to achieve the following objectives: given a mechanical object containing rolling bearings,

- to detect the presence of a defect,
- to recognize the specific kind of defect,
- to recognize different severity levels of the same fault as belonging to the same class,
- to provide the diagnostic system with high robustness to noise, or, better, to different levels of noise.

In this first noise assessment we do not make any change to the proposed methodology or to the classification technique to improve the robustness to noise, since this first analysis aims to find the basic robustness to noise of our algorithms. Successively, we will analyze how and how much we can improve this robustness making appropriate changes.

7.2.2 Methodology

The proposed methodology includes four main steps:

- data preparation,
- feature selection,
- training,
- test.

The *data preparation* step consists in processing the data and preparing them to be given in input to the feature selection algorithm. Since we decided to work in the frequency domain, we transform the time vibration signals by means of the *Fast Fourier Transform* (FFT). As already stated, unlike the classical approach, which identifies specific characteristic frequencies associated with given defects, we try to automatically find out the frequencies able to discriminate among the different defects taken into consideration.

From an operation point of view, we first balance the data using the random undersampling technique in order to have classes with the same number of samples. We chose this algorithm since it is very fast, very simple, it has empirically been shown to be one of the most effective resampling methods, and we can decide exactly how many elements should be removed [72, 141, 142].

Based on heuristic considerations, for each of the four accelerometers, we consider the frequency interval [1,300] Hz, sampled every 1Hz. Within this interval, we take into account six frequency ranges: [1,50] Hz, [51,100] Hz, [101,150] Hz, [151,200] Hz, [201,250] Hz, and [251,300] Hz. As there are four accelerometers, up to $300 \times 4 = 1200$ frequency samples (obtained by concatenating the four groups of 300 frequency samples relative to the four accelerometers) could be used to represent each signal (Fig. 7.1). In other words, each signal could be represented in $\mathbb{R}^n$ with $n \leq 1200$. Hereafter, each frequency sample will be referred to as *feature*.

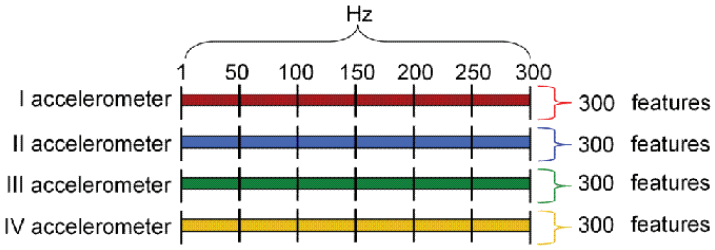


Figure 7.1 *Organization of the features considering the four accelerometers.*

On the other hand, we also desire to rank the accelerometers according to their contribution to classification accuracy so as to identify the most significant accelerometer(s). We therefore consider the four accelerometers separately from each other.

As second step we perform a *feature selection* process in order to decrease the feature space dimension and to reduce the training and test time as well as to identify the most significant and useful features (*discriminant frequencies*, i.e., the frequencies that are able to provide the best accuracy when used to represent the signals to be classified.). This step is performed by means of the Forward Feature Selection (FFS) algorithm. We continue to use this algorithm since it has proved to be a very strong and valuable method in this bearing fault classification problem. We adopt LDC and QDC (with the *regularization parameter* fix to 0) to perform both feature selection and classification of the signals represented through the selected features.

We use 4 LDCs and 4 QDCs: each LDC/QDC works on the frequency range [1,300] Hz of a particular accelerometer.

The feature selection and the training and test processes are performed through the following steps:

1 First, fixed a maximum number of features equal to 10 (based on heuristic considerations) so as to keep the computational complexity at a low level, we perform the FFS to extract the discriminant frequencies. To this aim, in order to guarantee more stable and reliable results [73], for each accelerometer, we repeat the FFS for a reasonable number t of times using both the LDC and QDC classifiers. In each trial we apply the Hold-out method [73] to generate different training and test sets (70% and 30%,

respectively, of the total data for training and test sets). In our case we set t equal to 30; this number is suggested by [73] to be a typical value used in simulation.

2 We identify the *stable features* (SFs), i.e., the features that are the same and selected in the same order, in all the trials, by the FFS. Indeed, the features selected by the FFS may vary from one trial to another. In order to guarantee a higher level of generalization, we are, therefore, interested in identifying the features that are significant for all the training sets.

3 Once identified the SFs, we use them to compute the classification accuracy expressed in the form (mean±standard deviation) for each accelerometer over the 30 different test sets previously generated. We then compare the four accelerometers and identify the best one(s).

4 Based on design specifications, we consider good, and thus acceptable, an accuracy higher than a threshold θ ($\theta = 99.00\%$ in our case). If at least one accelerometer meets this requirement, we consider the best of them. Otherwise, we try to improve the performance making use of the feature-level approach to building classifier ensembles going to step 5.

5 Let us use the term *configuration* to refer to the following three elements: a classifier (either LDC or QDC), an accelerometer, and the related SFs. Considering the two best configurations, we identify the *stable ranges*, i.e., the frequency ranges containing the selected SFs. There are six possible stable ranges, namely, [1,50] Hz, [51,100] Hz, [101,150] Hz, [151,200] Hz, [201,250] Hz, and [251,300] Hz. Once identified the stable ranges, we consider the union of these stable ranges and, on them, we perform again the FFS using both the LDC and QDC classifiers. We repeat the feature selection 30 times (on 30 different training sets, as before), and again we collect the new SFs. Using them we evaluate the classification accuracy on the 30 new different test sets. If at least one configuration provides an accuracy higher than θ, then we consider the best of them. Otherwise, we try to improve the performance making use of the classifier-level approach to building classifier ensembles considering more complex classifiers such as neural networks. Finally, if no classifier achieves the required

accuracy, we resort to the combination-level approach to build classifier ensembles.

7.2.3 First series of experiments

The aim of this series of experiments is to classify the signals into two classes C1 and C7, where C7 identifies the set of classes C2, C3.1, C4, and C5. With this experiment we aim to detect a faulty bearing as soon as the damage occurs, that is why we used in the training process only the lowest severity (C3.1) of the indentation on the roll (C3) and not the higher levels of severity C3.2 and C3.3.

Performing the steps of the classification methodology described in the previous section, we evaluate the accuracy (Table 7.1) for each classifier after identifying the SFs. Please note that in Table 7.1, like in all the following ones, the standard deviation will be referred to as "Std.Dev".

Table 7.1 *Classification of C1, C7. Accuracy and number of SFs for the four accelerometers*

Acc.	LDC		QDC	
	Num. of SFs	Accuracy (Mean±Std.Dev.)	Num. of SFs	Accuracy (Mean±Std.Dev.)
1	2	(89.44±0.08)%	2	(90.14±0.08)%
2	2	(98.29±0.21)%	2	(98.32±0.25)%
3	1	(97.32±0.04)%	**2**	**(99.65±0.08)%**
4	2	(95.42±1.27)%	2	(97.11±0.16)%

From Table 7.1 we find out that only the third accelerometer with two SFs extracted by the QDC classifier (bold text in Table 7.1) achieves an accuracy (99.65%) higher than the threshold θ ($\theta = 99.00\%$), thus we consider the configuration consisting of the QDC classifier applied to the SFs of the third accelerometer shown in Table 7.2 as the optimal configuration. With this configuration we also achieve a very low standard deviation (0.08%). This means that the use of an additional feature would bring to a negligible improvement. Thus, considering only two features we reduce the complexity to an optimal level from $\mathbb{R}^{1200}$ to $\mathbb{R}^2$. Fig. 7.2 shows the separation of the two classes performed

by QDC using the SFs 277 and 296 of the third accelerometer, while Table 7.3 shows a typical confusion matrix for this problem.

Table 7.2 *Classification of C1, C7. List of the SFs for the best configuration*

QDC, Accelerometer 3
277, 296

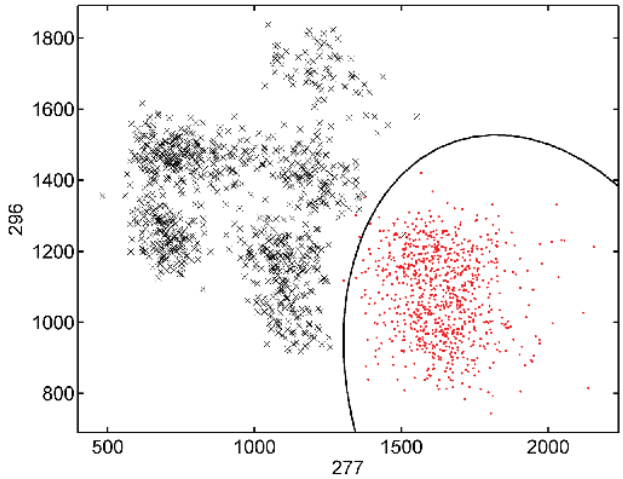


Figure 7.2 *Separation of C1 (dots) and C7 (crosses) for the test set performed by QDC using the SFs of the third accelerometer.*

Although this experiment was primarily aimed at detecting a fault as soon as it appears, we would also be interested in recognizing higher levels of severity. In other words, we want to classify signals of classes C3.2 and C3.3 as belonging to the damaged class as well. On the other hand the choice not to include C3.2 and C3.3 in the training test is motivated by the fact that we cannot expect to have always all the different types of severity of a particular defect at our disposal to train a classifier. Thus we want to check whether the classifier, trained using only the *basic defects* (i.e., the defects at their lowest level of severity), can correctly recognize also the *derived defects* (i.e., the defects at higher levels of severity). As already stated, this is crucial to make our method a practical tool in real applications.

Using the optimal configuration previously found we test our clas-

Table 7.3 *Classification of C1 and C7. Confusion matrix for the test set*

		Estimated Labels		Total
		C1	C7	
True labels	C1	863	4	867
	C7	1	866	867
Total		864	870	1734

sifier on a test set composed not only by C1, C2, C3.1, C4, and C5 but also by C3.2 and C3.3. Over 30 trials, we obtain an average accuracy of $(99.69 \pm 0.14)\%$. Fig. 7.3 shows the separation of the two classes C1 and C6 (we remind that C6 is the set including C2, C3, C4, and C5) performed by QDC using the SFs 277 and 296 of the third accelerometer, while Table 7.4 shows a typical confusion matrix for this problem. Thus our classifier trained with the basic defects is able to recognize also the derived defects with a very high accuracy and a very low standard deviation.

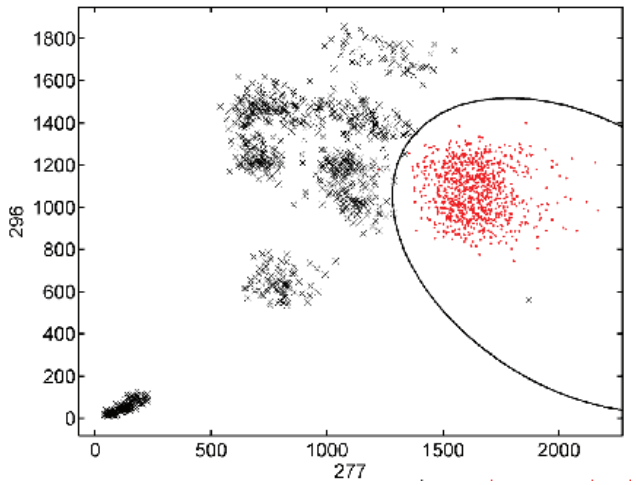


Figure 7.3 *Separation of C1 (dots) and C6 (crosses) for the test set performed by QDC using the SFs of the third accelerometer.*

Table 7.4 *Classification of C1 and C6. Confusion matrix for the test set*

		Estimated Labels		Total
		C1	C6	
True labels	C1	866	1	865
	C6	4	863	867
Total		870	864	1734

7.2.4 Second series of experiments

The aim of this series of experiments is to classify the signals into five classes C1, C2, C3.1, C4, and C5, i.e., to recognize the different types of faults, but considering only the lowest level (C3.1) of fault severity for class C3. From Table 7.5, considering the accuracies, we identify, as the most promising accelerometers, the second and the third using the LDC classifier.

Table 7.5 *Classification of C1, C2, C3.1, C4, and C5. Accuracy and number of SFs for the four accelerometers*

Acc.	LDC		QDC	
	Num. of SFs	Accuracy (Mean±Std.Dev.)	Num. of SFs	Accuracy (Mean±Std.Dev.)
1	1	(62.85±0.51)%	2	(62.26±0.49)%
2	**3**	**(98.13±0.22)%**	2	(96.54±0.12)%
3	**3**	**(96.88±0.22)%**	2	(94.70±0.61)%
4	3	(95.70±0.41)%	2	(96.55±0.16)%

Since no configuration reaches an accuracy higher than θ (θ = 99.00%), thus not meeting the criteria described in the methodology section, we try to improve the performance making use of the feature-level approach to building classifier ensembles. To this aim, we consider the SFs for the two best accelerometers (bold text in Table 7.5), the second and third in our case, using the LDC classifier (Table 7.6).

Then we identify the stable ranges, i.e., the frequency ranges containing the stable features above. In this case, we obtain the range

Table 7.6 *Classification of C1, C2, C3.1, C4, and C5. List of the SFs for the two best configurations*

LDC, Accelerometer 2	LDC, Accelerometer 3
296, 295, 277	277, 296, 96

[251,300] Hz for the second accelerometer, and the ranges [251,300] Hz and [51,100] Hz for the third accelerometer. Finally, we consider the union of these three ranges and perform the FFS on them with LDC and QDC. We repeat the feature selection 30 times and again we collect the SFs and the accuracy (Table 7.7).

Table 7.7 *Classification of C1, C2, C3.1, C4, and C5. Accuracy and number of SFs*

Acc. (Freq. range)	LDC		QDC	
	Num. SFs	Accuracy (Mean±Std.Dev.)	Num. SFs	Accuracy (Mean±Std.Dev.)
2 [251, 300] 3 [251, 300]	4	(99.46±0.07)%	4	(99.82±0.06)%

Table 7.8 shows the SFs for the optimal configuration with an accuracy of (99.82±0.06)% obtained by the QDC classifier. Please note that in Table 7.8 the features ranges [1,50], [51,100], [101,150] correspond, respectively, to the range [251,300] Hz of the second accelerometer, and the ranges [51,100] Hz and [251,300] Hz of the third accelerometer. Thus the SFs 127 and 146 correspond, respectively, to the features 277 and 296 of the third accelerometer while the SFs 46 and 45 correspond, respectively, to the features 296 and 295 of the second accelerometer.

From Table 7.7 we can see that the accuracy obtained by the QDC classifier (the best of the two) meets our design specifications. Table 7.9 shows a typical confusion matrix using the QDC classifier and the selected SFs. In this experiment, we managed to reduce the space dimension and thus the complexity to an acceptable level (signals are

Table 7.8 *Classification of C1, C2, C3.1, C4, and C5. List of the SFs for the best configuration*

QDC, Accelerometer 2, 3
127, 146, 46, 45

represented only in $\mathbb{R}^4$), drastically decreasing the memory and time required for training and classification.

Table 7.9 *Classification of C1, C2, C3.1, C4, and C5. Confusion matrix*

		Estimated Labels					Total
		C1	C2	C3.1	C4	C5	
True Labels	C1	456	0	0	0	0	456
	C2	0	455	0	1	0	456
	C3.1	0	0	456	0	0	456
	C4	0	1	0	455	0	456
	C5	0	0	0	0	456	456
Total		456	456	456	456	456	2280

In this experiment, like in the previous one, we worked using only the lowest level of fault severity, i.e., C3.1. Again, as we are interested not only in identifying the defect as soon as possible, but also in recognizing higher levels of severity, we decide to check if our classification system is able to classify signals of classes C3.2 and C3.3 (derived defects) as belonging to the same class C3.1 (basic defect), as they represent the same category of defect, namely, indentation on the roll.

Using the previously found configuration (Table 7.8), we test our classifier on a test set composed not only by C1, C2, C3.1, C4, and C5 but also by C3.2 and C3.3. Over 30 trials, we obtained an accuracy of (99.80±0.02)%. A typical confusion matrix for this problem is shown in Table 7.10. From Table 7.10 we can notice that all the elements of class C3 were correctly classified.

Table 7.10 *Classification of C1, C2, C3, C4, and C5. Confusion matrix*

		Estimated Labels					Total
		C1	C2	C3	C4	C5	
True Labels	C1	456	0	0	0	0	456
	C2	0	455	0	1	0	456
	C3	0	0	456	0	0	456
	C4	0	1	0	455	0	456
	C5	0	0	0	0	456	456
Total		456	456	456	456	456	2280

7.2.5 Third series of experiments

This series of experiments aims to assess the robustness of the proposed method to noise. To this aim, we repeat the previous experiment (training on C1, C2, C3.1, C4 and C5, and test on C1, C2, {C3.1, C3.2, C3.3}, C4 and C5) using the optimal configuration previously found (Table 7.8). We train the QDC classifier with a training set consisting of signals not affected by noise, then we test it on signals affected by different levels of noise. In particular we use ten levels of noise characterized by $NL_h = \{5, 10, 15, 20, 25, 30, 40, 60, 80, 100\}$ as stated in the Methodology section.

Table 7.11 shows the appreciable level of robustness to noise of our classification system over 100 trials. We obtained very good results up to SNR=16.52 db (accuracy higher than 90.00%, red text in Table 7.11) and acceptable results (accuracy between 80.00% and 90.00%, blue text in Table 7.11) up to SNR=9.59 db.

In order to increase the robustness to noise, we adopted MLP and RBF neural networks, and compared them with the QDC classifier. We use MLPs with one hidden layer and all neurons characterized by a logarithmic sigmoidal transfer function. We try different numbers of hidden neurons (10, 15, 20, 25, 30, 35, 40, 45, 50). We adopt RBFs with one hidden layer and all neurons characterized by a Gaussian transfer function. We try different numbers of hidden neurons (10, 15, 20, 25, 30, 35, 40, 45, 50) and different spread values (0.3, 0.4, 0.5, 0.6, 0.7, 0.9, 1, 1.1, 1.2, 1.3).

Table 7.11 *QDC classifier. Test set affected by noise*

NL	SNR (db)	Accuracy (Mean±Std.Dev.)	NL	SNR (db)	Accuracy (Mean±Std.Dev.)
5	40.55	(99.35±0.22)%	30	9.59	(86.20±1.01)%
10	28.62	(97.93±0.15)%	40	44.59	(77.41±1.25)%
15	21.47	(96.24±0.47)%	60	-2.75	(62.44±1.78)%
20	16.52	(92.50±0.77)%	80	-7.61	(51.55±1.49)%
25	12.45	(89.75±0.74)%	100	-11.35	(44.28±0.90)%

The inputs to the MLPs and RBFs are the SFs previously selected by QDC, i.e., the features 296 and 295 of the second accelerometer, and the features 277 and 296 of the third accelerometer.

The MLP with 50 hidden neurons provides the best performance among all the MLPs (Table 7.12), while, among the RBFs, the best performance is obtained by the RBF with 45 hidden neurons and a spread value of 1.2 (Table 7.13.

Table 7.12 *MLP with 50 hidden neurons. Test set affected by noise*

NL	SNR (db)	Accuracy (Mean±Std.Dev.)	NL	SNR (db)	Accuracy (Mean±Std.Dev.)
5	40.55	(99.62±0.01)%	30	9.59	(91.63±0.84)%
10	28.62	(99.32±0.21)%	40	4.59	(84.73±0.65)%
15	21.47	(98.48±0.26)%	60	-2.75	(72.67±0.81)%
20	16.52	(96.51±0.47)%	80	-7.61	(63.75±1.75)%
25	12.45	(93.87±0.76)%	100	-11.35	(56.00±1.24)%

Comparing the results in Tables 7.11, 7.12, and 7.13, we can see that the MLP and RBF do increase the robustness to noise. In particular, both of them improve both the good results (accuracy higher than 90.00%) up to 9.59 db, and the acceptable results (accuracy higher than 80.00%) up to 4.59 db.

Considering the results in greater detail, we can affirm that the RBF

Table 7.13 *RBF with 45 hidden neurons. Test set affected by noise*

NL	SNR (db)	Accuracy (Mean±Std.Dev.)	NL	SNR (db)	Accuracy (Mean±Std.Dev.)
5	40.55	(99.63±0.07)%	30	9.59	(92.11±0.59)%
10	28.62	(99.43±0.18)%	40	4.59	(84.16±1.05)%
15	21.47	(99.05±0.19)%	60	-2.75	(69.41±1.10)%
20	16.52	(97.13±0.33)%	80	-7.61	(55.92±1.63)%
25	12.45	(94.21±0.76)%	100	-11.35	(47.00±1.19)%

obtains a higher accuracy and a lower standard deviation for the first levels of noise (5, 10, 15, 20, 25, and 30) compared to both the QDC and the MLP. However, for the subsequent levels of noise (40, 60, 80, and 100), the performance of the RBF starts to decrease and becomes quite similar to the one achieved by the QDC classifier. The MLP provides better accuracy for all the different levels of noise compared to the QDC, concerning both the average accuracy and the standard deviation, showing to be more stable to the noise compared to the QDC classifier. Furthermore, even though the RBF is better than the MLP for the first levels of noise, the MLP offers a more graceful performance degradation for high levels of noise.

Thus, we can affirm that for acceptable levels of noise, the best results and, consequently, the best robustness are obtained by the RBF, while, for higher levels of noise, the MLP results to be the best. Figs. 7.4 and 7.5 clarify the comparison among these three classifiers (QDC, MLP and RBF).

7.2.6 Conclusion to the experiments for the first assessment of the robustness to noise

In this section we have presented an automatic method, based on classification techniques, for diagnosing faults in rolling element bearings.

The proposed method has been applied to experimental data, registered by four accelerometers, and related to four different defects on rolling bearings, namely, indentation on the inner raceway, indentation on the roll, sandblasting of the inner raceway, and unbalanced cage, and

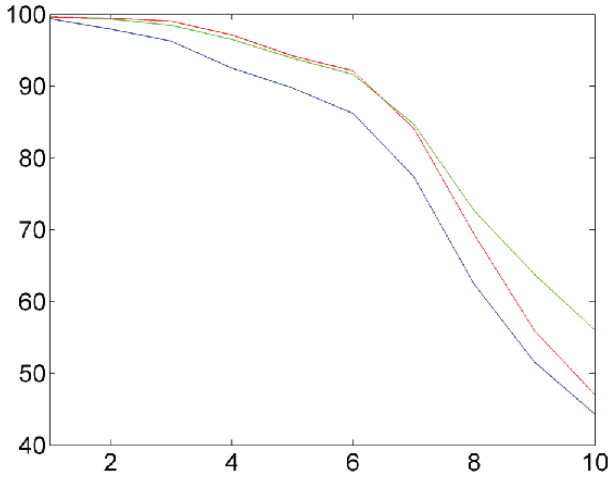


Figure 7.4 *Evolution of the accuracies of QDC (blue), MLP (green), RBF (red) at the increase of the noise level. x-axis: level of noise, y-axis: accuracy.*

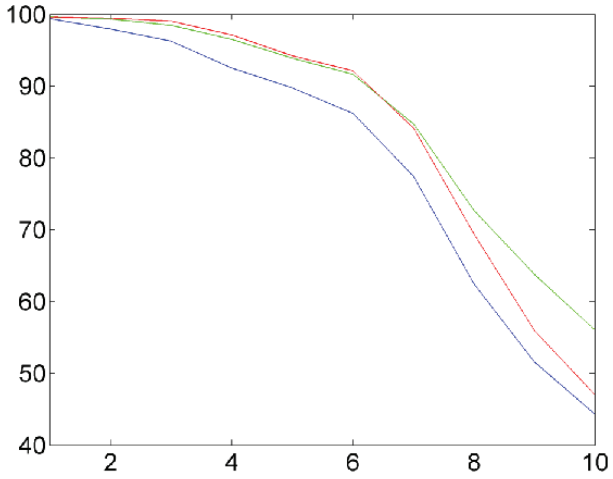


Figure 7.5 *Zoom of Fig. 7.4 for the first levels of noise.*

different levels of severity for one of them, namely, light, medium and high severity for the indentation on the roll. The method has proved to be highly sensitive to identify both different defects and levels of severity for the considered defects. We achieved an accuracy on the test set always higher than 99.00%.

We have also performed a noise analysis to assess the robustness of our methodology to noise, comparing the behaviors of the different

classifiers varying the level of noise by which signals were affected. In particular, we have classified the noisy signals by means of a classifier trained on signals without noise. We succeeded in increasing robustness to noise making use of more complex classifiers such as MLP and RBF neural networks.

The appreciable levels of robustness to noise achieved could be even increased if, in practical cases, we could filter noise out of the acquired signals before their classification. Alternatively, or in addition, we could train the classifier both with signals without noise and with signals with added noise, choosing the noise level to be added depending on the specific situation of interest.

7.3 Methods to increase the robustness to noise

7.3.1 Introduction to the developed methods to increase the robustness to noise

In this section we perform a noise analysis to assess the degree of robustness to noise of a neural classifier aimed at performing multi-class diagnosis of rolling element bearings. More precisely this section aims to achieve the following objectives: given a mechanical equipment containing rolling bearings,

- to detect the presence of a defect,
- to recognize the specific kind of defect,
- to provide the diagnostic system with high robustness to different levels of noise.

We make an analysis using ten levels of noise, each of which characterized by a different signal-to-noise ratio ranging from 40.55 db to -11.35 db. We classify the noisy signals by means of a neural classifier initially trained on signals without noise, then we repeat the training process with signals affected by increasing levels of noise. We show that adding noisy signals to the training set we manage to significantly increase the classification accuracy.

Finally we apply the two most used strategies to combine classifiers: *classifier fusion* and *classifier selection*, and show that, in both cases, we can significantly increase the performance of the single best classifier and thus improve the robustness to noise.

The analysis presented in this section will also show how to identify both the type of classification system (e.g., single classifier or classifier ensemble) and how many and which noise levels should be used in the training phase in order to maximize the classification accuracy and the robustness to noise in the application domain of interest.

7.3.2 Methodology

We deal with a multi-class classification problem aimed at classifying the data into five classes: one class for the faultless samples and four damaged classes, each related to a different fault.

We also perform noise analysis aimed at finding out how we can increase the robustness of the classification system to noise.

Since the best results in term of classification accuracy and robustness to noise were obtained using the second and third accelerometers, as shown in the previous section, we have decided to use only these two accelerometers in the present section. Thus we will refer the original second and the third accelerometers, respectively, as the first and second accelerometers.

We carry out the following steps considering independently the samples belonging to the two accelerometers. We have to deal with class imbalance, i.e., the presence of significant differences in class prior probabilities, which may seriously worsen the performance of many classification systems that assume relatively balanced data distribution [42, 51]. Examples are decision trees, backpropagation neural networks, Bayesian networks, nearest neighbor, support vector machines [17, 60, 122]. For this reason the first step we perform is to balance the data before each experiment in order to obtain classes with the same cardinality as the least numerous one. We use a random undersampling method to balance class distribution by eliminating elements of all classes but the least numerous one until all classes have the same cardinality. We chose this algorithm since it is very fast, very simple, it has empirically been shown to be one of the most effective resampling methods, and we can decide exactly how many elements should be removed [72, 141, 142].

The second step of the proposed methodology is to find out which is the best way to represent the signals. We decided to work in the frequency domain by transforming the signals by the Fast Fourier Transform (FFT). Since the phenomenon originated by the considered faults

produces an effect mainly in the spectral interval [1,300] Hz, we consider that frequency interval. Furthermore we select the spectral sampling equal to 1 Hz so as to describe in sufficient details the most relevant features related to the faults. Each frequency sample will be referred to as *feature*. Thus, we have a total of 300 features from which we will select the DFs (i.e., the features that are able to provide the best accuracy when used to represent the signals to be classified) making use of the Forward Feature Selection (FFS) algorithm with a QDC classifier. The FFS is thus performed on the range [1,300] Hz using the QDC classifier. We consider the *regularization parameter* fixed to 0 for all the performed experiments. By means of the FFS, we reduce the feature space by removing the noisy features and retaining only the features that provide more information. We only consider as DFs the first M features, $f_1, \ldots f_M$, $M \leq 300$, that significantly increase the accuracy. By "significantly" we mean an accuracy improvement of at least 1%. In other words, f_i is considered only if the accuracy obtained by the set $\{f_1, \ldots, f_i\}$ is at least 1% higher than the accuracy achieved by the set $\{f_1, \ldots, f_{i-1}\}$, $i = 2, \ldots, 300$.

Once identified the DFs, we use a more complex, but also more flexible classifier, namely, an MLP to perform the classification process using the extracted DFs.

We assess the robustness to noise of the MLP taking ten different increasing levels of noise into account starting from level 1, which corresponds to the lowest noise level, up to level 10, which is the highest noise level.

We perform eleven experiments, each time adding a higher level of noise to the training set. More precisely, the i-th experiment, i=1,...,11, is characterized by a training set that includes signals affected by noise levels from 0 to i-1 (level 0 refers to noise-free signals). The eleven experiments are then tested on the same test set consisting of, besides noise-free signals, signals affected by all levels of noise. Hereafter, we will refer to the noise free data as NF, and to the data affected by noise level i, i=1,...,10, as N_i (Table 7.14). In the experiments we compute the test accuracy separately on each of the levels of noise NF, N1, ..., N10.

After evaluating the performance of each accelerometer independently (we recall that the methodology steps previously described are

Table 7.14 *Training set and test set configurations for each experiment*

Training set	Test set
NF	
NF,N1	
NF,N1,N2	
NF,N1,N2,N3	
NF,N1,N2,N3,N4	NF;N1;N2;N3;N4;N5;
NF,N1,N2,N3,N4,N5	N6;N7;N8;N9;N10
NF,N1,N2,N3,N4,N5,N6	
NF,N1,N2,N3,N4,N5,N6,N7	
NF,N1,N2,N3,N4,N5,N6,N7,N8	
NF,N1,N2,N3,N4,N5,N6,N7,N8,N9	
NF,N1,N2,N3,N4,N5,N6,N7,N8,N9,N10	

performed separately for each accelerometer), we exploit classifier fusion and classifier selection, using the two accelerometers and the related trained MLPs to try to increase the results obtained by each accelerometer and the corresponding MLP.

First, we make use of the fusion of classifiers using the combiners *simple mean*, *maximum*, *minimum*, and *product*. Finally we adopt the classifier selection, in particular the method *Direct K-NN Estimate* (with $K = 1$) and compare its results with the ones obtained by the single classifiers and by the mixture of classifiers resulting from the fusion methods. The accuracy for each experiment is computed as the average accuracy on the test set over 100 trials. For each trial we randomly balance the classes of the original data set and then we randomly select 70% of the samples to create the training set and take the remaining 30% of the samples as test set.

7.3.3 Experiments and results

We aim to classify the signals into five classes C1, C2, C3, C4, and C5.

Considering the NF data as training set, using the FFS and the QDC classifier, we obtain four DFs, both for the first and the second

accelerometers independently. More precisely, the selected DFs are 296, 295, 277, and 278 for the first accelerometer, and 277, 296, 96, and 185 for the second accelerometer. Tables 7.15 and 7.16 show, respectively, for the first and second accelerometer, the four DFs with the accuracy that is achieved starting from the first selected feature and adding, each time, a new DF.

Table 7.15 *First accelerometer. List of the extracted DFs and accuracies reached by adding the corresponding feature*

DFs	Accuracy reached using a QDC classifier
296	55.21%
295	85.65%
277	88.15%
278	90.80%

Table 7.16 *Second accelerometer. List of the extracted DFs and accuracies reached by adding the corresponding feature*

DFs	Accuracy reached using a QDC classifier
277	75.43%
296	96.15%
96	97.20%
185	98.25%

From Table 7.15, we can see that using feature 296 we achieve an accuracy of 55.21%, using features 296 and 295 we obtain an accuracy of 85.65%, etc. Increasing further the number of features results in either a negligible improvement ($< 1\%$) or even a decrease in the accuracy. To let the user have an intuitive idea of how things are, Figs. 7.6 and 7.7 show the signals belonging to the five classes around the first and second extracted DFs for the second accelerometer.

With the four DFs we perform the classification trying different MLPs characterized by different parameters, such as the number of

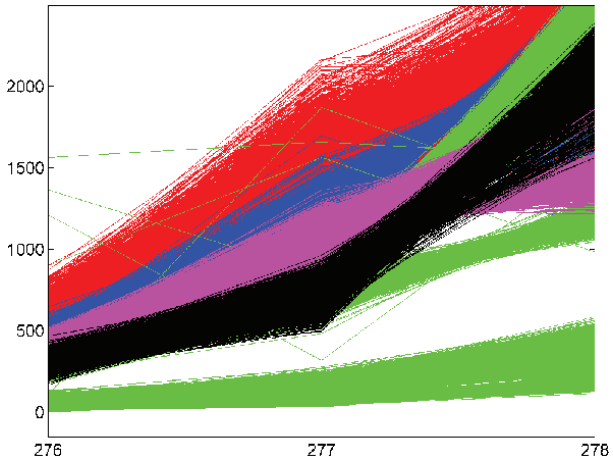


Figure 7.6 *Second accelerometer. Signals realted to the five classes C1 (red), C2 (blue), C3 (green), C4 (magenta), and C5 (black) around feature 277.*

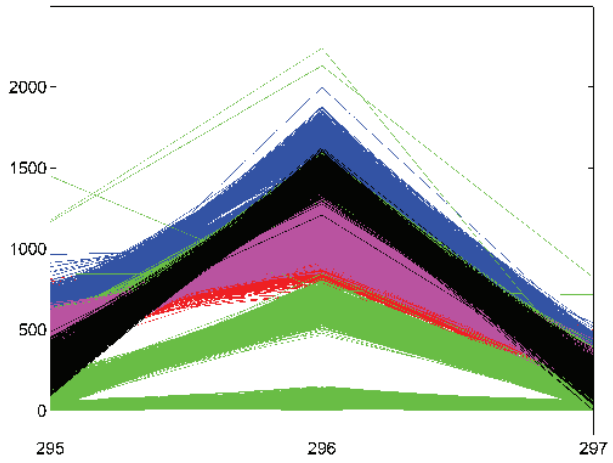


Figure 7.7 *Second accelerometer. Signals related to the five classes C1 (red), C2 (blue), C3 (green), C4 (magenta), and C5 (black) around feature 296.*

hidden neurons, the neuron transfer function, etc. The best results are obtained, for both accelerometers, by an MLP characterized by the parameters reported in Table 7.17.

We achieve an average accuracy of 92.06% and 98.39% over 100 trials, respectively, for the first and second accelerometers. Two typical confusion matrices for this classification problem are reported in Ta-

Table 7.17 *MLP's parameters*

Parameters	Values
Transfer function	Logarithmic sigmoid
Training algorithm	Back propagation
Type of learning rate	Dynamic
Initial learning rate	value 0.01
Momentum	0.95
Number of hidden layers	1
Number of hidden neurons	40
Stop criterion	Negligible error improvement, 0.1%

bles 7.18 and 7.19 for the first and second accelerometer, respectively.

Table 7.18 *Confusion matrix using the best MLP for the first accelerometer*

		Estimated Labels					Total
		C1	C2	C3	C4	C5	
True Labels	C1	439	3	0	13	0	455
	C2	1	414	10	27	3	455
	C3	0	32	423	0	0	455
	C4	40	11	0	399	5	455
	C5	2	0	0	17	436	455
Total		482	460	433	456	444	2275

To assess the robustness of the previous classification system (MLP with the four DFs), we test the system separately on data affected by the ten levels of noise. We perform this step considering each accelerometer independently. We recall that the data N1 are affected by the first level of noise, i.e., NL = 5, the data N2 are affected by the second level of noise, i.e., NL=10, and so on. Table 7.20 shows the noise analysis for the MLP classifier over 100 trials for both the accelerometers. Considering the first accelerometer, we never obtain a very high robustness (accuracy higher than, or equal to, 95.00%), but we achieve

Table 7.19 *Confusion matrix using the best MLP for the second accelerometer*

		Estimated Labels					Total
		C1	C2	C3	C4	C5	
True Labels	C1	453	2	0	13	0	455
	C2	2	441	5	5	2	455
	C3	1	5	442	4	3	455
	C4	1	2	4	448	0	455
	C5	0	0	0	0	455	455
Total		457	450	451	457	460	2275

a high/acceptable robustness (accuracy in the range 90.00%÷95.00%) up to SNR=21.47 db, and an acceptable/low robustness (accuracy in the range 85.00%÷90.00%) up to SNR=12.45 db. Regarding the second accelerometer, we obtain a very high robustness up to SNR=40.55 db, and a high/acceptable robustness up to SNR=21.47 db. In Table 7.20 we highlighted the different levels of robustness to noise with different colors, so that a very high robustness is represented with the color red, a high/acceptable robustness is represented with the color blue and an acceptable/low robustness is associated to the color green.

The two MLPs associated with the two accelerometers perform quite differently for different noise levels. More precisely, the second accelerometer outperforms the first one for the noise-free data and the first levels of noise, but the robustness to the noise decreases pretty fast with the increase of the level of noise. On the other hand, the robustness to the noise of the MLP applied to the first accelerometer decreases more slowly with the increasing of the level of noise, and remains pretty more accurate for higher levels of noise compared to the MLP applied to the second accelerometer.

As stated in the methodology section, in order to increase the robustness to noise, we train the MLP both with signals without noise and with signals affected by different levels of noise. The results (mean values) are shown in Tables 7.21 and 7.22, respectively, for the first and second accelerometers. In particular in Tables 7.21 and 7.22 the symbol ÷ indicated a range, thus, for example, if we consider the data

Table 7.20 *First and second accelerometers. Training set not affected by noise. Test sets affected by different levels of noise*

Noisy test data set	NL	SNR (db)	First Acc. Accuracy (Mean)	Second Acc. Accuracy (Mean)
N1	5	40.55	92.44%	97.99%
N2	10	28.62	91.17%	94.29%
N3	15	21.47	90.28%	90.84%
N4	20	16.52	89.03%	82.35%
N5	25	12.45	87.20%	74.97%
N6	30	9.59	84.78%	67.21%
N7	40	4.59	80.69%	54.77%
N8	60	-2.75	70.91%	40.55%
N9	80	-7.61	63.86%	34.16%
N10	100	-11.35	56.36%	30.22%

NF÷N5, then it means that we are considering all the noisy data from noise level 0 to noise level 5. From Table 7.21 we can notice that, for the first accelerometer, increasing the number of levels of noise used in the training process does not make the classification system more robust to noise. On the contrary, considering the second accelerometer (Table 7.22), we can see that the robustness to the noise is significantly increased. In particular, for the first noise levels in the test set, there is a performance improvement as the noise levels in the training set increase up to a point in which the performance slightly decreases though remaining noticeably high. For the last noise levels in the test set, on the contrary, there is a continuous performance improvement with the increase of the training noise levels, although the achieved accuracy remains pretty low.

Based on the complementary behavior of the two accelerometers, we then perform an analysis to see how a combination of classifiers can modify (possibly increase) the robustness to noise. More precisely we make use of the two strategies to combine classifiers: *fusion* and *selection*. We remind that the classifier fusion approach all classifiers know the whole feature space, whereas in the classifier selection one

each classifier is considered as an expert in a specific portion of the feature space. Regarding the fusion of classifiers, we try four different types of nontrainable combiners, i.e., *simple mean*, *maximum*, *minimum*, and *product*, whose results are shown in Tables 7.23-7.26. As far as the classifier selection approach is concerned, we apply the decision-independent estimates method Direct K-NN Estimate by dynamically and locally estimating the competence of each classifier. The results for each level of noise applying the classifier selection approach are reported in Table 7.27.

Table 7.21 *First Accelerometer. Training and test sets affected by different levels of noise. Average accuracy percentage on the test sets*

		Test set											
		NF	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Mean
Training set	NF	92.06	92.44	91.17	90.28	89.03	87.20	84.78	80.69	70.91	63.86	56.36	81.70
	NF÷N1	92.88	92.68	91.54	90.38	89.13	87.89	86.24	81.91	73.12	65.47	58.60	82.71
	NF÷N2	88.03	86.97	85.66	85.13	84.26	82.96	81.55	78.32	71.15	63.95	58.43	78.76
	NF÷N3	85.10	84.35	83.21	82.75	82.20	80.71	80.33	78.02	71.68	64.86	59.98	77.56
	NF÷N4	83.05	82.38	81.80	81.31	81.12	79.67	79.17	77.11	70.83	64.99	59.89	76.48
	NF÷N5	81.05	79.65	79.46	78.43	78.95	77.85	77.37	75.97	71.40	66.70	59.97	75.16
	NF÷N6	79.95	79.64	78.71	78.59	78.00	77.31	76.86	74.99	69.86	65.00	59.98	74.44
	NF÷N7	65.28	65.11	64.32	64.18	64.16	63.60	62.93	61.72	58.59	55.92	52.67	61.68
	NF÷N8	63.03	62.65	60.47	60.11	60.15	59.93	59.21	58.68	55.99	53.03	50.68	58.53
	NF÷N9	61.97	61.07	60.34	59.43	59.26	58.53	57.64	57.34	54.86	52.84	50.21	57.59
	NF÷N10	59.51	59.93	59.39	58.77	58.52	58.04	57.61	56.52	54.50	52.57	50.03	56.85
	Mean	77.45	76.99	76.01	75.40	74.98	73.97	73.06	71.02	65.72	60.83	56.07	

Table 7.22 *Second Accelerometer. Training and test sets affected by different levels of noise. Average accuracy percentage on the test sets*

		Test set											
		NF	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Mean
Training set	NF	98.39	97.99	94.29	90.84	82.35	74.97	67.21	54.77	40.55	34.16	30.22	69.61
	NF÷N1	98.41	98.13	96.09	93.14	88.35	82.89	76.72	66.02	51.98	44.00	39.00	75.88
	NF÷N2	98.94	98.80	97.40	95.82	92.77	88.96	83.86	74.86	60.81	50.21	43.01	80.49
	NF÷N3	99.06	98.68	97.41	95.19	92.94	88.74	85.95	75.14	60.98	50.81	42.60	80.68
	NF÷N4	98.80	98.78	97.42	95.51	94.14	91.89	87.67	78.25	64.24	52.41	45.10	82.20
	NF÷N5	98.12	98.25	96.69	94.92	93.08	90.44	86.30	78.08	60.01	50.00	40.00	80.53
	NF÷N6	98.01	97.97	96.37	95.11	92.43	89.75	85.39	77.00	60.53	50.03	43.48	80.55
	NF÷N7	97.33	95.72	94.22	91.96	90.37	86.64	85.11	77.14	61.49	51.80	44.75	79.68
	NF÷N8	96.14	95.00	92.78	91.46	90.07	86.97	84.19	78.43	67.15	56.92	48.99	80.73
	NF÷N9	95.81	93.23	92.36	90.91	88.99	86.56	83.81	78.78	66.59	57.74	51.61	80.58
	NF÷N10	95.02	92.96	91.50	90.28	88.32	86.34	83.90	79.61	67.14	58.15	52.40	80.51
	Mean	97.64	96.86	95.14	93.19	90.35	86.74	82.74	74.37	60.13	50.57	43.74	

Table 7.23 *Classifier fusion: simple mean combiner. Training and test sets affected by different levels of noise. Average accuracy percentage on the test sets*

		Test set											
		NF	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Mean
Training set	NF	97.55	97.84	94.71	94.52	90.83	85.52	81.02	73.17	62.70	54.28	48.60	80.07
	NF÷N1	95.73	95.90	95.26	93.59	89.08	84.91	80.18	71.47	59.84	50.89	46.11	78.45
	NF÷N2	95.13	96.36	96.16	94.09	89.10	84.80	82.63	71.76	58.03	50.82	46.07	78.63
	NF÷N3	94.13	94.29	93.84	91.74	88.83	84.83	80.96	70.20	59.69	51.60	46.72	77.89
	NF÷N4	93.34	93.16	92.41	91.57	88.73	86.12	83.06	75.02	62.24	53.47	47.12	78.75
	NF÷N5	93.94	93.32	93.92	92.93	90.71	87.79	85.23	77.84	65.29	56.00	49.84	80.62
	NF÷N6	91.16	90.92	90.06	88.60	87.04	86.05	86.46	77.19	64.03	56.41	49.13	78.82
	NF÷N7	90.30	89.88	87.20	87.33	86.61	84.35	81.18	76.97	64.70	58.20	51.12	77.98
	NF÷N8	86.80	86.71	86.47	86.14	85.44	83.39	80.35	76.15	66.56	57.88	51.55	77.04
	NF÷N9	85.94	86.23	86.53	84.84	84.99	84.66	80.35	77.96	67.54	58.88	52.11	77.27
	NF÷N10	85.80	86.13	86.23	83.99	83.04	80.92	81.02	72.81	65.14	58.68	51.23	75.91
	Mean	91.80	91.88	91.16	89.94	87.67	84.85	82.04	74.59	63.25	55.19	49.05	

Table 7.24 *Classifier fusion: maximum combiner. Training and test sets affected by different levels of noise. Average accuracy percentage on the test sets*

		Test set											
		NF	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Mean
Training set	NF	97.83	97.86	96.84	93.74	89.70	84.86	80.12	72.59	60.16	50.89	46.64	79.20
	NF÷N1	94.72	93.32	92.33	90.69	88.50	83.18	78.23	70.42	59.40	51.99	46.05	77.17
	NF÷N2	94.40	93.84	93.16	91.05	85.96	81.51	78.32	69.84	58.84	50.46	45.82	76.65
	NF÷N3	94.22	94.40	93.56	91.79	87.78	83.03	79.27	68.85	55.69	47.67	43.83	76.37
	NF÷N4	94.06	93.93	92.96	91.36	88.98	85.18	80.82	72.44	58.80	49.81	45.21	77.59
	NF÷N5	94.04	92.96	92.96	91.52	90.24	86.32	84.18	75.67	61.88	52.84	45.88	78.95
	NF÷N6	93.67	93.37	92.61	91.22	88.44	86.70	83.84	76.53	60.87	52.69	45.79	78.70
	NF÷N7	92.21	92.19	91.82	90.31	87.50	85.97	81.57	76.88	63.24	53.10	48.98	78.52
	NF÷N8	92.16	91.99	90.86	89.99	86.98	85.83	81.77	75.10	63.50	53.99	48.87	78.28
	NF÷N9	92.73	91.09	90.42	89.46	85.45	85.19	85.66	76.76	65.11	56.60	49.75	78.93
	NF÷N10	92.02	91.29	90.58	89.36	85.42	85.28	84.51	75.30	64.73	56.47	49.88	78.62
	Mean	93.82	93.29	92.55	90.95	87.72	84.82	81.66	73.67	61.11	52.41	46.97	

Table 7.25 *Classifier fusion: minimum combiner. Training and test sets affected by different levels of noise. Average accuracy percentage on the test sets*

		Test set											
		NF	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Mean
Training set	NF	91.91	91.87	91.71	91.05	89.99	88.3	86.43	81.80	75.51	73.16	70.03	84.71
	NF÷N1	89.25	89.25	88.08	88.67	85.01	83.91	82.59	80.61	77.74	74.60	71.58	82.84
	NF÷N2	88.47	89.46	88.42	87.21	82.92	80.46	80.92	78.92	77.01	74.72	72.35	81.90
	NF÷N3	85.63	86.10	86.19	86.04	85.59	84.88	83.75	82.07	79.44	76.59	73.85	82.74
	NF÷N4	85.41	86.41	84.35	84.14	84.73	84.10	83.07	81.85	79.02	76.50	73.77	82.12
	NF÷N5	86.90	87.73	84.55	84.27	84.92	83.40	82.65	81.54	81.29	80.54	77.61	83.22
	NF÷N6	85.77	85.73	84.58	84.39	84.20	83.87	82.53	79.85	79.88	78.65	75.65	82.28
	NF÷N7	84.29	84.50	84.51	84.15	83.83	83.42	82.93	82.02	79.22	77.91	75.48	82.02
	NF÷N8	84.98	84.70	84.45	84.26	84.00	84.66	83.23	82.59	80.06	79.05	77.26	82.66
	NF÷N9	83.72	83.69	83.40	83.12	82.81	82.36	81.91	81.13	79.83	78.13	76.19	81.48
	NF÷N10	81.76	81.72	80.69	80.46	80.22	80.96	80.60	79.05	78.12	76.92	75.54	79.64
	Mean	86.19	86.47	85.54	85.25	84.38	83.67	82.78	81.04	78.83	76.98	74.48	

Table 7.26 *Classifier fusion: product combiner. Training and test sets affected by different levels of noise. Average accuracy percentage on the test sets*

		Test set											
		NF	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Mean
Training set	NF	95.87	95.14	94.56	92.63	90.25	86.93	84.25	74.34	59.23	51.63	47.13	79.27
	NF÷N1	89.98	88.83	87.45	87.06	86.96	85.71	79.35	71.19	56.18	49.18	45.67	75.23
	NF÷N2	90.34	88.74	88.15	87.04	87.46	84.39	80.88	70.29	59.89	50.17	45.09	75.68
	NF÷N3	88.28	86.27	86.12	84.70	83.06	80.42	77.66	71.84	58.80	50.11	45.01	73.84
	NF÷N4	88.47	86.43	85.81	84.86	83.58	81.52	79.85	74.15	61.55	53.63	47.56	75.22
	NF÷N5	88.72	87.68	87.17	86.53	85.20	81.08	79.87	73.50	61.99	54.12	47.96	75.80
	NF÷N6	86.36	86.43	86.63	85.73	83.98	81.32	79.38	73.26	60.39	54.32	48.42	75.11
	NF÷N7	81.88	81.50	81.04	80.79	79.40	78.16	76.97	73.86	64.84	57.94	52.38	73.52
	NF÷N8	82.10	81.14	80.70	79.31	78.12	76.33	75.51	71.07	64.54	55.93	50.60	72.30
	NF÷N9	83.30	83.34	82.88	81.35	79.83	78.08	76.41	72.45	65.04	56.35	54.80	73.98
	NF÷N10	83.09	83.31	82.53	81.61	80.30	78.54	76.81	72.79	65.06	56.74	54.20	74.09
	Mean	87.12	86.25	85.73	84.69	83.47	81.13	78.81	72.61	61.59	53.65	48.98	

Table 7.27 *Classifier selection: Direct K-NN Estimate. Training and test sets affected by different levels of noise. Average accuracy percentage on the test sets*

		Test set											
		NF	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Mean
Training set	NF	98.21	98.02	97.39	95.92	93.61	90.86	88.80	82.19	70.86	62.05	55.92	84.89
	NF÷N1	98.37	98.26	97.77	95.92	93.25	90.53	86.48	80.08	71.08	63.03	57.99	84.80
	NF÷N2	97.98	97.72	96.91	95.49	92.90	90.04	87.18	80.14	71.50	61.63	56.93	84.40
	NF÷N3	98.69	98.39	98.14	96.75	94.30	91.52	88.12	81.68	71.12	62.15	56.65	85.23
	NF÷N4	97.99	97.91	96.91	95.88	92.79	90.40	86.73	80.54	70.30	60.92	55.64	84.12
	NF÷N5	97.68	97.16	96.34	95.30	92.80	90.59	87.66	81.19	70.05	60.50	55.98	84.18
	NF÷N6	97.17	97.17	96.26	95.06	93.00	90.16	86.80	80.78	71.60	61.81	55.82	84.11
	NF÷N7	97.27	96.66	95.56	94.20	92.07	90.02	87.37	80.44	70.30	61.27	55.47	84.15
	NF÷N8	96.18	95.59	94.35	92.53	90.72	88.03	85.03	80.00	68.96	60.45	54.97	83.69
	NF÷N9	96.14	95.64	93.94	92.46	89.87	86.97	84.46	78.74	66.98	60.05	54.12	82.43
	NF÷N10	95.03	94.11	93.21	90.57	88.46	85.79	82.92	76.78	66.39	59.21	53.88	81.76
	Mean	87.12	86.25	85.73	84.69	83.47	81.13	78.81	72.61	61.59	53.65	48.98	

To make things clearer, Figs. 7.8-7.18 show an alternative representation of the information contained in Tables 7.21-7.27. In particular, there are as many figures as there are test sets (columns) in each of these tables. Each figure represents the classification accuracy percentage of all the classifiers, namely, first accelerometer, second accelerometer, simple mean combiner, maximum combiner, minimum combiner, product combiner, and Direct K-NN Estimate, as a function of the test set pertinent to the figure. On the x-axis of each figure we represent the various training sets (rows in the tables), i.e., NF, NF÷N1, etc. More precisely, the numbers 1÷11 on the x-axis correspond to NF, NF÷N1, etc. Therefore Fig. 7.8 shows the classification accuracy of all the seven classifiers on the test set consisting of noise-free signals as the composition of the training set varies from NF, NF÷ N1, up to NF÷N10; Fig. 7.9 shows the classification accuracy of the seven classifiers on the test set consisting of the data N1 (i.e., signals affected by the first level of noise) as the composition of the training set varies from NF, NF÷N1, up to NF÷N10, etc. In all the Figs. 7.8-7.18 we use the following line styles and colors to represent the seven classifiers:

- (-. red): first accelerometer,
- (- red): second accelerometer,
- (-. blue): mean combiner,
- (- blue): maximum combiner,
- (-. green): minimum combiner,
- (- green): product combiner,
- (-. black): Direct K-NN Estimate.

For the sake of clarity, we also show two more figures (Figs. 7.19 and 7.20) which correspond directly to Tables 7.25 and 7.27. More precisely, Figs. 7.19 and 7.20 represent the classification accuracy of the minimum combiner and the Direct K-NN Estimate, respectively. In the figures, the axes x and y represent, respectively, the test set and the accuracy percentage achieved by the given classifier on the appropriate test set. In particular, the numbers from 1 to 11 on the x-axes correspond, respectively, to the test sets NF, N1, N2, etc. Each curve represents the behavior of the classifier on the specific test set using a given training set. In particular, each training set is described by a different line style and color as follows:

- (-. red): NF,

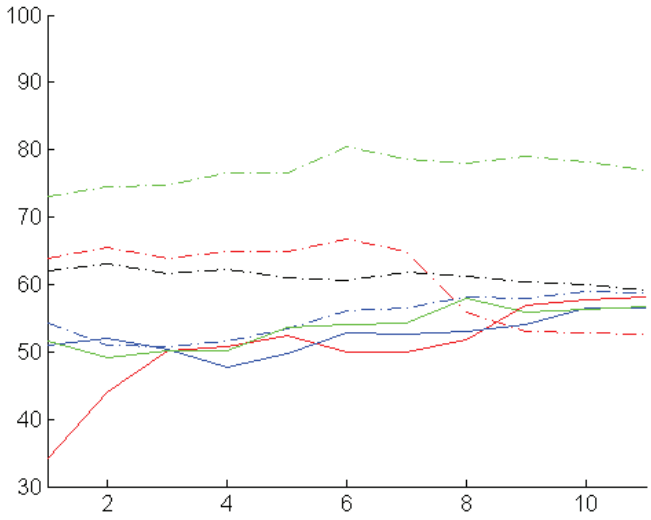


Figure 7.8 *Classification accuracy percentage of the seven classifiers on the test set NF as the composition of the training set varies.*

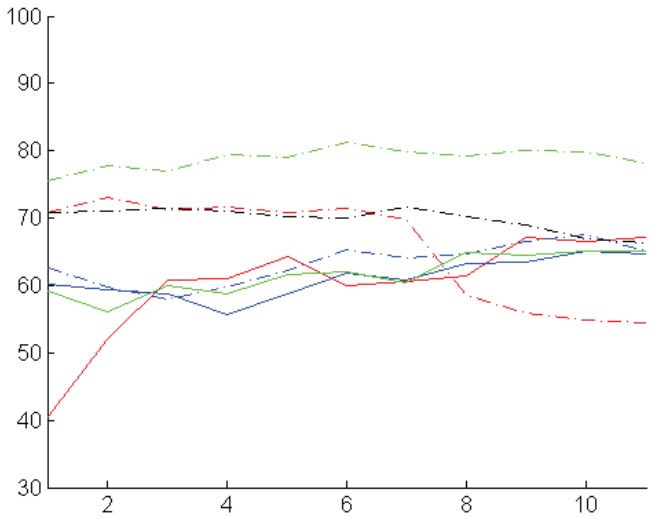


Figure 7.9 *Classification accuracy percentage of the seven classifiers on the test set N1 as the composition of the training set varies.*

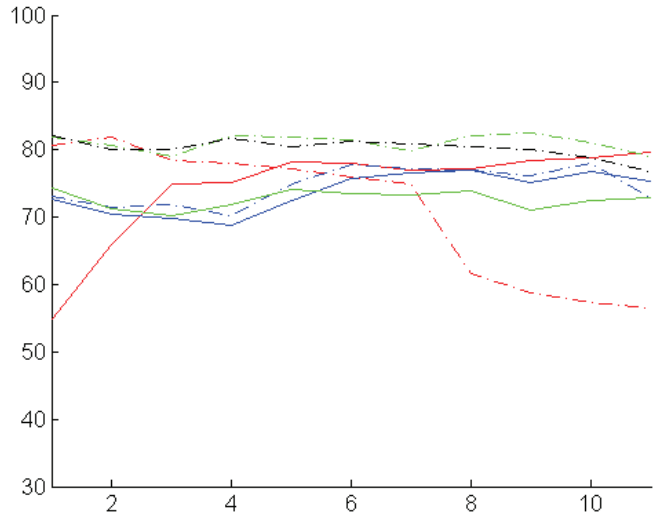


Figure 7.10 *Classification accuracy percentage of the seven classifiers on the test set N2 as the composition of the training set varies.*

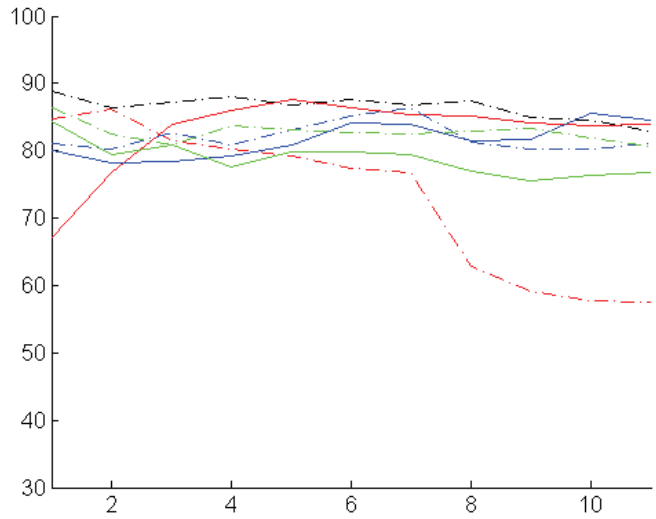


Figure 7.11 *Classification accuracy percentage of the seven classifiers on the test set N3 as the composition of the training set varies.*

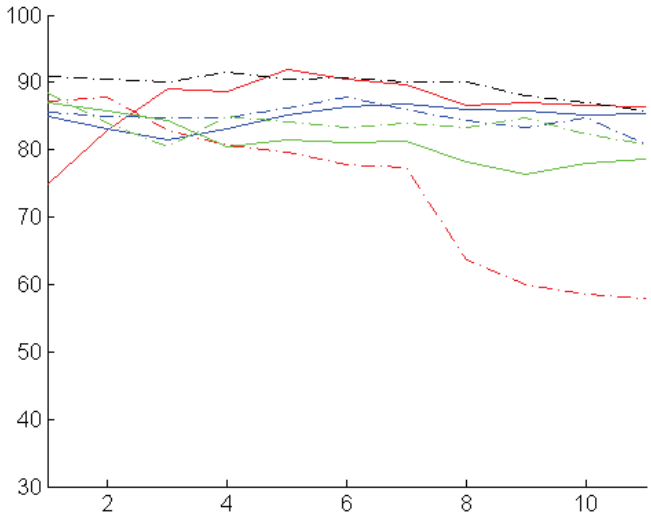


Figure 7.12 *Classification accuracy percentage of the seven classifiers on the test set N4 as the composition of the training set varies.*

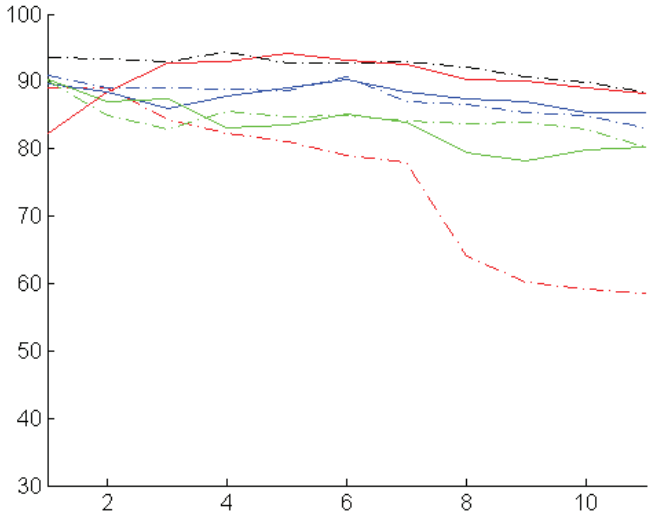


Figure 7.13 *Classification accuracy percentage of the seven classifiers on the test set N5 as the composition of the training set varies.*

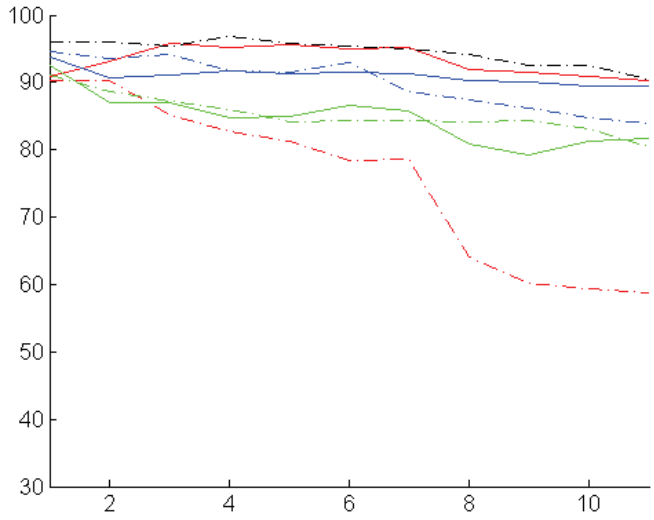


Figure 7.14 *Classification accuracy percentage of the seven classifiers on the test set N6 as the composition of the training set varies.*

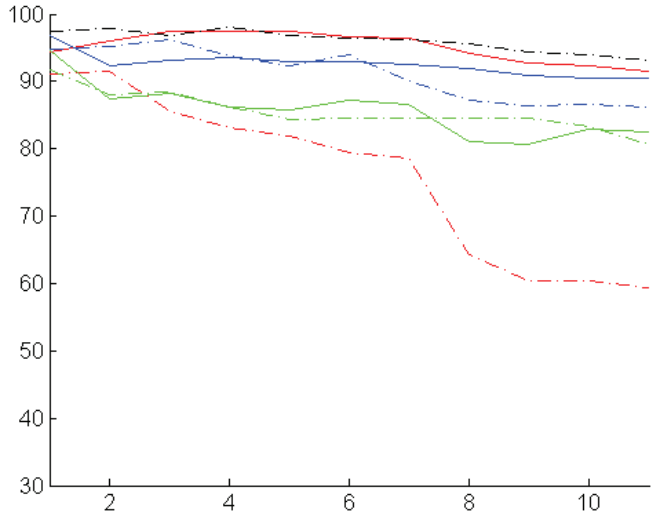


Figure 7.15 *Classification accuracy percentage of the seven classifiers on the test set N7 as the composition of the training set varies.*

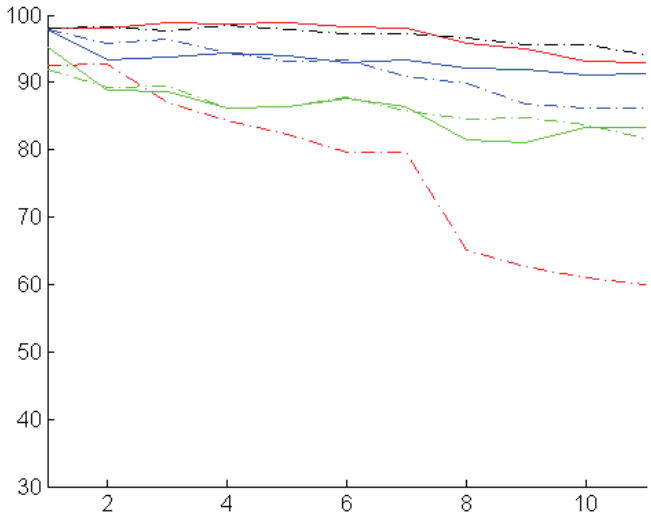


Figure 7.16 *Classification accuracy percentage of the seven classifiers on the test set N8 as the composition of the training set varies.*

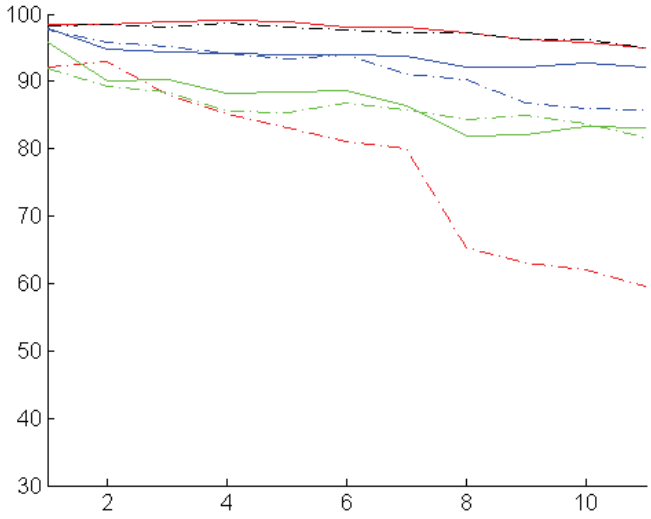


Figure 7.17 *Classification accuracy percentage of the seven classifiers on the test set N9 as the composition of the training set varies.*

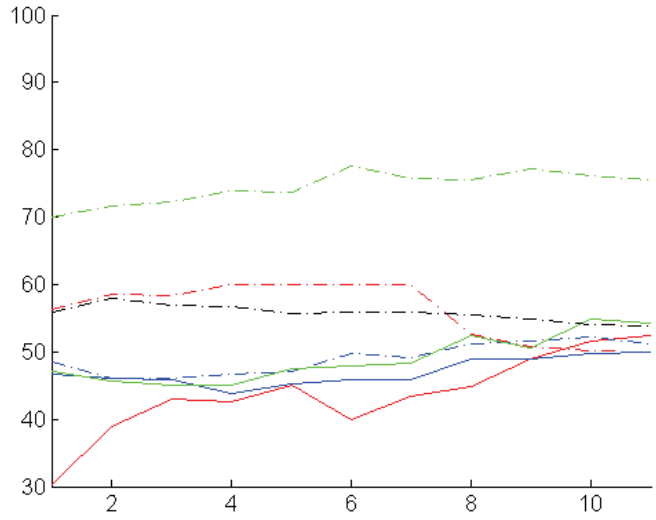


Figure 7.18 *Classification accuracy percentage of the seven classifiers on the test set N10 as the composition of the training set varies.*

- (- red): NF÷N1,
- (-. blue): NF÷N2,
- (- blue): NF÷N3,
- (-. green): NF÷N4,
- (- green): NF÷N5,
- (-. black): NF÷N6,
- (- black): NF÷N7,
- (-. cyan): NF÷N8,
- (- cyan): NF÷N9,
- (-. yellow): NF÷N10.

From Tables 7.21-7.27 and Figs. 7.8-7.20 we can notice that using a combination strategy we are able to significantly increase the classification accuracy. The choice of the specific combination strategy depends on the specific application domain and our knowledge of this domain. For example, we observe that, among the fusion approaches, the *minimum* combiner is the one that obtains the best results for high levels of noise, in particular for SNRs ranging from ï¿½2.75 db to ï¿½11.35 db. Thus, even though, for the lowest levels of noise, the *minimum* combiner

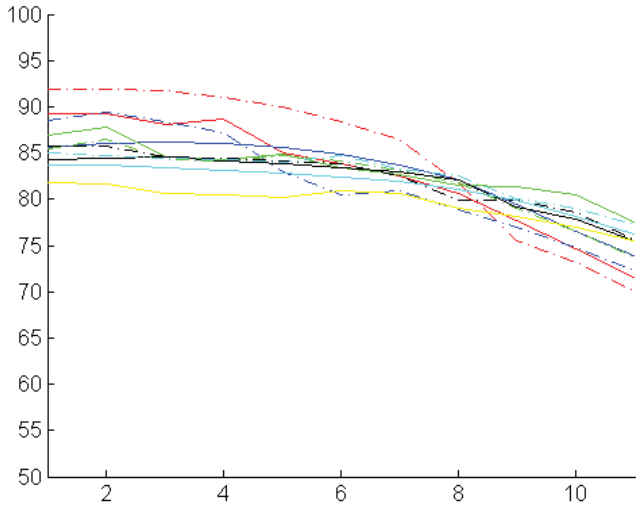


Figure 7.19 *A graphical representation of the information in Table 7.25 related to the minimum combiner.*

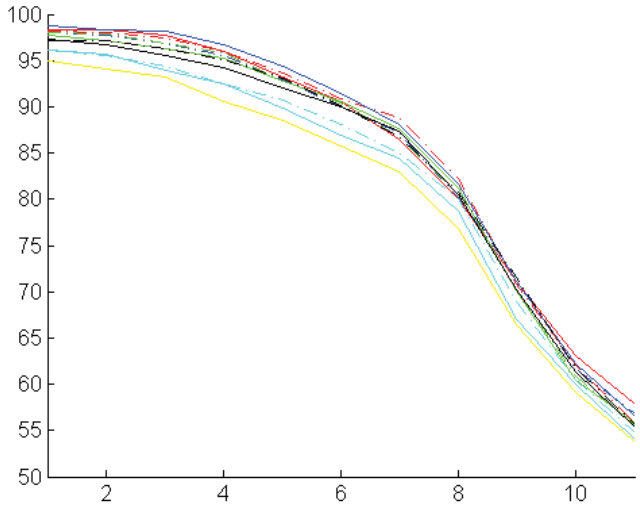


Figure 7.20 *A graphical representation of the information in Table 7.27 related to the Direct K-NN Estimate combiner.*

does not show very good results, it is surely the best classification system for very high levels of noise since its accuracy decreases gracefully

as the noise increases.

On the other hand, the best classification system for low and medium levels of noise is obtained by the classifier selection approach. Indeed, it generally outperforms the single MLP used for the best accelerometer (i.e., the second accelerometer) and all the other fusion strategies.

Finally, useful guidelines can be derived from the previous considerations. More precisely, depending on the specific real case, if we know the range within which the noise may vary in the real environment, we can perform an analysis aimed at identifying which is the best training configuration to be used to obtain the desired accuracy on the test set.

Consider, for example, an environment in which the SNR can vary from 16.52 db (NL=20) to 28.62 db (NL=10). Let us refer to Table 7.28, whose generic element is the average classification accuracy achieved by, respectively, the two accelerometers and the five classifier combiners, on the set consisting of the data N2, N3 and N4, i.e., the signals affected by the three levels of noise 10, 15 and 20. Analyzing these averages, we can notice that the highest average (96.40%) is achieved by the classificationsystem which makes use, as training set, the data affected by levels of noise up to the third level (NL=15, which corresponds to an SNR equal to 15.47 db) and, as classification strategy, the classifier selection, in particular the *Direct K-NN Estimate* algorithm.

Table 7.28 *Training and test sets affected by different levels of noise, with an SNR from 16.52 db (NL=20) to 28.62 db (NL=10). CF: Combiner Fusion. CS: Combiner Selection*

		Average Accuracy (%)						
		First Acc.	Second Acc.	CF Simple Mean	CF Maximum	CF Minimum	CF Product	CS Dir. K-NN Est.
Training set	NF	90.36	89.16	93.35	93.43	90.92	92.48	95.64
	NF÷N1	90.62	92.53	92.64	90.51	87.25	87.16	95.65
	NF÷N2	85.25	95.33	93.12	90.06	86.18	87.55	95.10
	NF÷N3	83.05	95.18	91.47	91.04	85.94	84.63	**96.40**
	NF÷N4	81.54	95.69	90.90	91.10	84.41	84.75	95.19
	NF÷N5	78.95	94.90	92.52	91.57	84.58	86.30	94.81
	NF÷N6	78.60	94.64	88.57	90.76	84.39	85.45	94.77
	NF÷N7	64.42	92.18	87.05	89.88	84.16	80.41	93.94
	NF÷N8	60.24	91.44	86.02	89.28	84.24	79.38	92.53
	NF÷N9	59.67	90.75	85.45	88.44	83.11	81.35	92.09
	NF÷N10	58.89	90.03	84.42	88.45	80.46	81.48	90.75

7.3.4 Conclusions to the developed methods to increase the robustness to noise

In this section we have performed a noise analysis to assess the robustness of a neural classification system, namely, an MLP, to different levels of noise. We have made use of the two main strategies in combining classifiers, fusion and selection. Considering the classifier fusion, we have applied four different combiners, namely, *simple mean*, *maximum*, *minimum*, and *product*, while for classifier selection we have exploited the *Direct K-NN Estimate* in order to dynamically evaluate the local competence of each classifier.

We have considered ten levels of noise, each of which characterized by a different signal-to-noise ratio, from 40.55 db to -11.35 db. In particular we have classified the noisy signals by means of a classifier trained on signals without noise, then we have used training data affected by increasing levels of noise.

We have shown that, using a combining strategy, we are able to significantly increase the classification accuracy and the robustness to noise. We have also shown how the presented analysis can be exploited to choose which classifier is the best and how it should be trained to achieve the best classification accuracy in the case of interest.

Chapter 8

The bearing diagnosis as a one-class classification problem

The important thing
is not to stop questioning.

- A. Einstein -

In this chapter we deal with diagnosis of rolling elements bearings within condition-based maintenance programs, considering the diagnosis issue as a one-class classification problem and classifying the signals into two classes, namely faultless and damaged classes.

In particular, we propose the use of the convex hull, usually adopted in application domains such as computer visualization, verification methods and computational geometry problems, and the snake operator, typically employed for image segmentation, as two one-class classifiers. Then we introduce two novel one-class classifiers, namely CSC_{CHC} and CSS, resulting from an appropriate integration of the convex hull and snake operator classifiers.

We compare CSC_{CHC} and CSS with traditional one-class classifiers, such as Gaussian, 1-NN and K-NN, in six experiments. We prove that our classifiers represent a valuable alternative to the traditional one-class classifiers since they achieve better results in all cases but one in which the difference between the proposed classifiers and the Gaussian classifier (the best) is practically negligible.

8.1 One-class classification to perform bearing diagnosis

As stated in Chap. 2, even though good results have been presented so far in the literature, little attention has been dedicated to the main challenging problem regarding the diagnosis of mechanical equipment: the lack of a significant set of damaged data.

In general, bearings can develop several types of faults (e.g., sandblasting, indentation on the inner raceway, unbalanced cage, and indentation on the roll) that can be further divided in classes according to the level of severity. It is difficult to collect all types of faults and severities to train a classification system, since it is not possible to identify all the possible types of faults and severity levels, for example an indentation on the roll can be positioned in different parts of a roll and thus it may produce different (vibration) signals. Furthermore, as shown in [141], (vibration) signals can change as time passes making thinner and fuzzier the division of faulty bearings into categories. Thus it is impossible to collect and "catalog" an infinite number of faults and severities. Moreover it is often difficult or even impossible to collect data from faulty bearings as it requires to put damaged bearings into the rotating machine causing unwanted consequences. For all these reasons the creation of a training set for the damaged samples can be either expensive or impractical and thus difficult to achieve. On the other hand, it is relatively cheap and simple to obtain measurements from faultless bearings and thus from a normally functioning machine.

However, most of the techniques presented in the literature deal with the bearing fault diagnosis as a two-class problem involving only data associated to pre-specified faults and severities regardless of all the "possible" types of fault and levels of severity they were not able to collect. Unfortunately, in real and practical cases, it is quite unlikely that a trained classification system will have to cope with only the *known faults and severities* (i.e., faults and severities used to train the classification system). On the contrary, it is very common that the classification system will have to cope with *unknown faults and severities* (i.e., faults and severities not used during the training process). Thus, even though some techniques can achieve very high accuracies on *known* faults and severities, they may perform very poorly on *unknown* faults and severities as shown in [142]. Actually, Figs. 8.1 and

8.2 show the discriminant functions obtained by some traditional classifiers, such as an LDC, a QDC and an MLP, which classify the samples into faultless class and damaged class. In Figs. 8.1 and 8.2 the red dots represent the faultless samples, while the blue and green dots represent the damaged samples, respectively, shown and not shown to the classification system during the training process. The achieved resubstitution (training) error of all the classifiers is practically 0%. However, when damaged samples (green dots) belonging to faults and severities different from the training ones (blue dots) are given to the classification system, the accuracy for the damaged class decreases significantly to an unacceptable level.

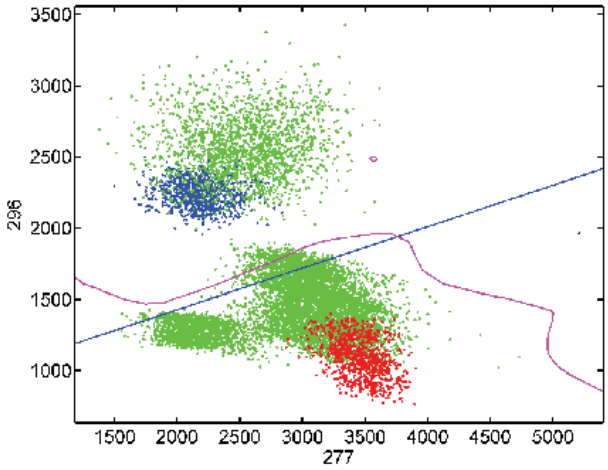


Figure 8.1 *Classification into faultless and damaged samples performed by an LDC (blue line) and an MLP (magenta line). The red dots and the blue dots represent, respectively, the faultless and damaged samples used during the training process. The green dots represent damaged samples belonging to faults and severity levels different from the ones (blue dots) used during the training process.*

In contrast with the approaches proposed in the literature, we would like to develop a classification system provided with high generalization capabilities, i.e., able to generalize the problem and thus correctly classify also damaged samples belonging to types of fault and levels of severity not used during the training process. For this reason we have decided to approach this classification problem as a one-class classification problem. This way, we achieve independence from the specific

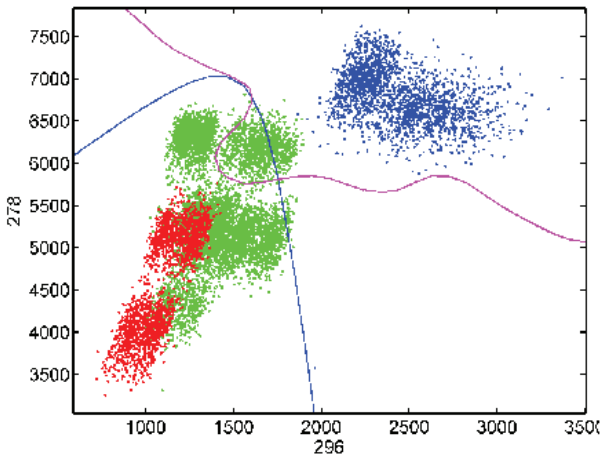


Figure 8.2 *Classification into faultless and damaged samples performed by a QDC (blue line) and an MLP (magenta line). The red dots and the blue dots represent, respectively, the faultless and damaged samples used during the training process. The green dots represent damaged samples belonging to faults and severity levels different from the ones (blue dots) used during the training process.*

damaged samples we were able to collect to train our classification system and, thus, we are able to develop a more general classifier that can reach a good accuracy not only for known but also for *unknown* faults and severities.

Finally, a further important aspect in the bearing diagnosis is the analysis of the accuracy of the classification system. Actually, in the literature, the classification results are typically expressed in terms of the average between the accuracies of the faultless class and the damaged class. However, except when a classification system reaches a 100% accuracy, looking at the average does not give any information about the single accuracy for the faultless and the damaged classes. However, we cannot consider a misclassification of a faultless sample for a damaged one with the same seriousness of a misclassification of a damaged sample for a faultless one. As a matter of fact it is usually preferable to misclassify a faultless element with a damaged one than vice versa. Indeed, in the former case we substitute a faultless bearing erroneously considered damaged, while in the latter case we do not perform any substitution of the damaged bearing letting the system continue to work in

a not appropriate way, perhaps reaching the breakdown which will be more expensive to repair than a simple, although useless, substitution. For this reason we will report the accuracies of both the faultless and the damaged classes independently.

8.2 Methodology

We deal with a classification problem aimed at classifying the data into two classes: faultless class and damaged class.

8.2.1 Signal representation

The first step of the proposed methodology aims to find out which type of signal it is better to sample. We decide to collect vibration signals. The reason for that, as explained in Chap. 2, is that vibration analysis may be advantageous in the bearing diagnostic field as the bearing area is the locus of the application of the basic dynamic loads and forces in a machine. Furthermore the vibration analysis provides the most information from the collected data [58].

8.2.2 Working domain and training and test sets creation

The second step of our methodologyconcerns the choice of the best domain to work with. We decide also in these experiments to work in the frequency domain by transforming the signals by the *Fast Fourier Transform* (FFT). The reason of our choice, as explained in Chap. 2, is that the frequency domain analysis makes it easier to identify and isolate certain frequency components of interest, such as the theoretical characteristic frequencies of the defects, compared to the time-domain analysis. Based on heuristic considerations and the results obtained in the experiments described in the previous chapters, we consider the frequency interval [1,300] Hz, sampled every 1 Hz. The frequency samples will be referred to as *features* in the following, so that we have a total of 300 features describing each signal sample.

Training and test sets are then created using the Hold-out method considering 70% of the original data as training set and the remaining data as test set. The split into training and test set is performed in a random manner.

Once the training set has been created, we balance the data in order to obtain classes with the same cardinality as the least numerous one.

We perform a random undersampling to balance the data. We chose this algorithm since it is very fast, very simple, it has empirically been shown to be one of the most effective resampling methods, and we can decide exactly how many elements should be removed [72, 141, 142].

8.2.3 Feature space dimensionality reduction

The third step of the methodology deals with the reduction of the feature space. We do not represent each signal with 300 features as the complexity of the classifiers would become overwhelming and even undesirable as a high number of features generally decreases the accuracy. Consequently, we perform a feature selection process in order to find the most significant and useful features, called discriminant frequencies (DFs). These features are the ones that provide the best accuracy when used to represent the signals to be classified. Furthermore, the reduction of the number of features brings to two further advantages that consist in the reduction of both the memory necessary for signal representation, and the computational time needed for classifier training and test. Both these aspects are crucial since we would like to develop a methodology that can be used in a real time environment.

However, in the classification process for fault diagnosis, when the number of objects in the training set is too small for the number of features used, most classification procedures cannot find good classification boundaries. For this reason, we adopt a feature selection algorithm to find out the DFs. This step is performed using the *Forward Feature Selection* (FFS). We chose to use FFS because it is a simple and often efficient algorithm which represents a reasonable compromise between exhaustive search and random search.

We adopt the statistical classifiers LDC and QDC to perform the FFS. Unfortunately we can perform a feature selection only if we know at least some elements of the damaged class. For this reason we fix as hypothesis that we always have at least either one type of fault (even though not exhaustively sampled) or one severity level of one type of fault (even though not exhaustively sampled) with which we can perform the feature selection, while, of course, the training of the one-class classifiers will be done using only one class, i.e., the faultless class.

8.3 Experiments' framework

We propose two main series of experiments, in the first we propose the comparison among some traditional multi-class classifiers, such as statistical classifiers, namely LDC and QDC, and neural networks, namely, MLP, with a one-class classifier, namely, a convex hull (CHC) to show that multi-class classifiers generally perform quite poorly with respect to the *unknown* faults and severities while the one-class CHC performs pretty better. In the second series of experiments we propose the comparison of several one-class classifiers, some well-known such as Gaussian, K-NN, 1-NN, and some introduced in this thesis such as the convex hull classifier (CHC), the snake operator classifier (SOC) and mixtures of convex and snake classifiers. In particular, we propose two one-class classifiers obtained by the combination of the CHC and SOC: the first (CSC) alternates the use of more CHCs and SOCs starting from a CHC, and the second (CSS) consists in a CHC stretched starting from an SOC applied on a CHC. All these classifiers are described in details in Chap. 3.

For each series of experiments, we perform six experiments, for each of which we perform the steps described in the Methodology section. Each of these experiments is characterized by different numbers of damaged classes used both in the training phase (which includes the feature selection process and the classifier training process) and in the test phase. The classification problem to be solved is still a 2-class problem, however the damaged class is composed each time by a different number of damaged classes. For the sake of clarity, let the sets of damaged classes used in the training phase and in the test phase referred to as classes C_{TR_D} and C_{TS_D}, respectively, independently on the specific classes involved.

For the first experiment of each series, during the feature selection process, we use the two classes, faultless class (C1) and damaged class (C_{TR_D}). Then to train the one-class classifiers we use only the faultless class (C1) while to train the multi-class classifiers we use both C1 and C_{TR_D}. On the contrary, in the test process we use the faultless class as well as the damaged class C_{TS_D} composed by all the damaged classes except the one used in C_{TR_D}. In this way we repeat the first experiment six times, since we have six damaged classes. Table 8.1 shows all the six different combinations of the classes used during the

training and test processes for the first experiment in each of the two series of experiments.

In the second experiment we use again C1 in the training phase, but this time to perform the feature selection process we use C_{TR_D} composed by two damaged classes instead of only one like in the first experiment. Then we repeat this experiment as many times as the number of combinations of two damaged classes, i.e., 15 times. Each time we use, as test set C1 and C_{TS_D}, composed by all the damaged classes but the two used in C_{TR_D} (Table 8.2).

Table 8.1 *Classes used in the first experiment of both the two series of experiments*

Classes used to perform the FFS	Classes used to train multi-class classifiers	Classes used to train one-class classifiers	Classes used for the test phase
C1, $C_{TR\_D} = \{C2\}$	C1, C_{TR_D}	C1	C1,$C_{TS\_D} = \{C3.1, C3.2, C3.3, C4, C5\}$
C1, $C_{TR\_D} = \{C3.1\}$	C1, C_{TR_D}	C1	C1,$C_{TS\_D} = \{C2, C3.2, C3.3, C4, C5\}$
C1, $C_{TR\_D} = \{C3.2\}$	C1, C_{TR_D}	C1	C1,$C_{TS\_D} = \{C2, C3.1, C3.3, C4, C5\}$
C1, $C_{TR\_D} = \{C3.3\}$	C1, C_{TR_D}	C1	C1,$C_{TS\_D} = \{C2, C3.1, C3.2, C4, C5\}$
C1, $C_{TR\_D} = \{C4\}$	C1, C_{TR_D}	C1	C1,$C_{TS\_D} = \{C2, C3.1, C3.2, C3.3, C5\}$
C1, $C_{TR\_D} = \{C5\}$	C1, C_{TR_D}	C1	C1,$C_{TS\_D} = \{C2, C3.1, C3.2, C3.3, C4\}$

Table 8.2 *Classes used in the second experiment of both the two series of experiments*

Classes used to perform the FFS	Classes used to train multi-class classifiers	Classes used to train one-class classifiers	Classes used for the test phase
C1,C_{TR_D} =$\{C2, C3.1\}$	C1, C_{TR_D}	C1	C1,C_{TS_D}=$\{C3.2, C3.3, C4, C5\}$
C1, C_{TR_D} =$\{C2, C3.2\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C3.1, C3.3, C4, C5\}$
C1,C_{TR_D} =$\{C2, C3.3\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C3.1, C3.2, C4, C5\}$
C1 C_{TR_D} =$\{C2, C4\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C3.1, C3.2, C3.3, C4, C5\}$
C1,C_{TR_D} =$\{C2, C5\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C3.1, C3.2, C3.3, C4\}$
C1,C_{TR_D} =$\{C3.1, C3.2\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.3, C4, C5\}$
C1,C_{TR_D} =$\{C3.1, C3.3\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.2, C4, C5\}$
C1,C_{TR_D} =$\{C3.1, C4\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.3, C3.3, C5\}$
C1,C_{TR_D} =$\{C3.1, C5\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.2, C3.3, C4\}$
C1,C_{TR_D} =$\{C3.2, C3.3\}$	C1, C_{TR_D}	C1	C1,C_{TS_D}=$\{C2, C3.1, C4, C5\}$
C1,C_{TR_D} =$\{C3.2, C4\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.1, C3.3, C5\}$
C1 C_{TR_D} =$\{C3.2, C5\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2C3.1, C3.3, C5\}$
C1,C_{TR_D} =$\{C3.3, C4\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.1, C3.2, C5\}$
C1,C_{TR_D} =$\{C3.3, C5\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.1, C3.2, C4\}$
C1, C_{TR_D} =$\{C4, C5\}$	C1, C_{TR_D}	C1	C1,C_{TS_D} =$\{C2, C3.1, C3.2, C3.3\}$

The remaining four experiments are similar to the first two, with the difference that the number of repetitions of the i-th experiment, with i=3...6, is equal to the number of combinations of i damaged classes. The number of combination is evaluated as follows:

$$\binom{6}{i}.$$

This approach, which performs each experiment once for each of all possible combinations of the damaged classes involved in the training process of that experiment, was chosen with the purpose of investigating how independent is the proposed classification methodology from the data available during the training phase. Actually, each combination of damaged classes can be considered as a different data set. Of course some combinations may result to be better and to give more useful information to the classifier so that the *unknown* faults and severities can be better classified.

For the sake of clarity, Figs. 8.4-8.5 show the distribution of the classes considering the two DFs extracted by an LDC classifier performing the FFS for the first combination of classes for the fourth, fifth and sixth experiments, respectively. From these figures we can notice that the damaged samples (represented by the gray stars and the light gray triangles) corresponding to the classes C2, C3.1, C3.2, C3.3, C4, and C5 are quite spread throughout the feature space, while the faultless class (C1), represented by the black dots, appears quite compact fitting perfectly a one-class classifier. In particular, going from Fig. 8.4 to Fig. 8.5 more and more damaged classes are used to perform the process of feature selection.

More precisely, in Fig. 8.4, which corresponds to the fourth experiment, we use four damaged classes (C2, C3.1, C3.2, and C3.3) besides the faultless class (C1) in the feature selection process and two damaged classes (C4 and C5) as well as the faultless class (C1) in the test process; in Fig. 8.3, which is related to the fifth experiment, we use five damaged classes (C2, C3.1, C3.2, C3.3, and C4) and the faultless class (C1) to perform the feature selection process and only one damaged class (C5) and the faultless class (C1) to perform the test process.

Finally in Fig. 8.5, which corresponds to the sixth experiment, we use all the damaged classes as well as the faultless class both in the feature selection and test processes.

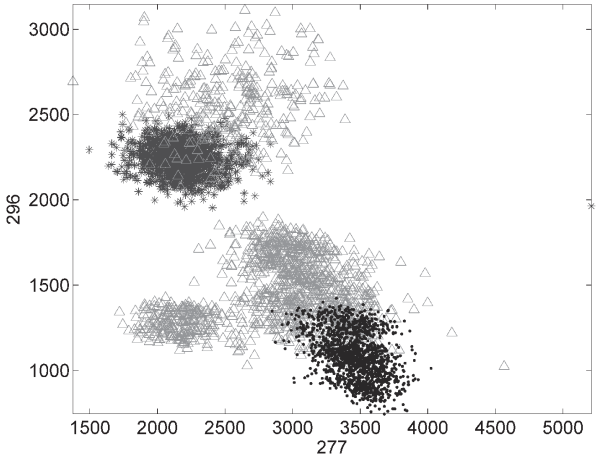


Figure 8.3 *Fourth experiment. Faultless class (C1): black dots; damaged classes used in the training set (C2, C3.1, C3.2, C3.3): gray stars; damaged classes used in the test set (C4, C5): light gray triangles.*

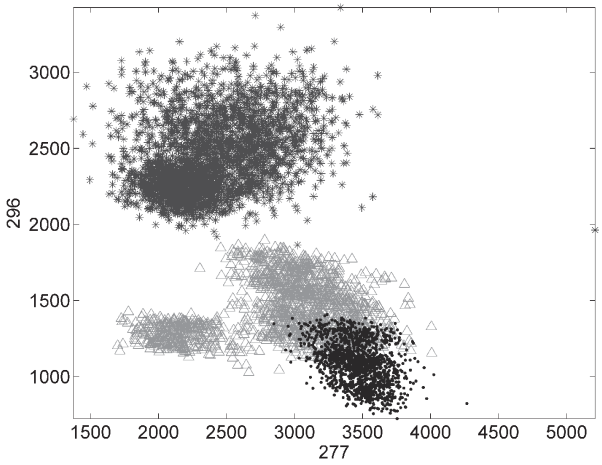


Figure 8.4 *Fifth experiment. Faultless class (C1): black dots; damaged classes used in the training set (C2, C3.1, C3.2, C3.3, C4): gray stars; damaged classes used in the test set (C5): light gray triangles.*

8.4 One-class classifiers vs multi-class classifiers

8.4.1 Introduction to the experiments to compare one-class with multi-class classifiers

In this section, we compare traditional classifiers, such as LDC, QDC and neural networks (in particular MLPs) with a one-class classifier,

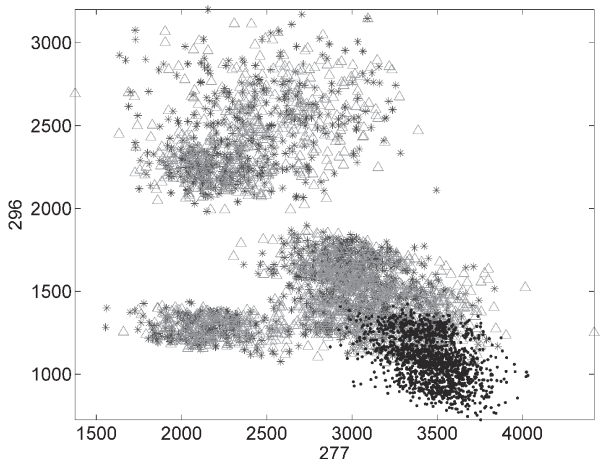


Figure 8.5 *Sixth experiment. Faultless class (C1): black dots; damaged classes used in the training set (C2, C3.1, C3.2, C3.3, C4, C5): gray stars; damaged classes used in the test set (C2, C3.1, C3.2, C3.3, C4, C5): light gray triangles.*

namely, convex hull. With reference to rolling element bearing diagnosis, we show that the convex hull classifier outperforms the traditional multi-class classifiers in the classification of *unknown* faults and severities.

8.4.2 Experiments and results

In all the experiments, for each classifier we compute its classification accuracy, for both faultless and damaged classes (the ones not used in the feature selection and training processes). We perform a comparison among the six classifiers, namely, LDC, MLP, CHC trained with the DFs selected by LDC, and QDC, MLP, CHC trained with the DFs selected by QDC.

Tables 8.3-8.8 show the average, the worst case and the standard deviation evaluated over all the different combinations for the experiment under consideration. Each row of these tables is associated with a different classifier.

From all these tables we can notice that the CHCs, besides presenting very high performance for the faultless class, always provide the best performance for the damaged class in all experiments.

In the following, we briefly comment each table singularly considering only the results related to the damaged class, comparing the be-

Table 8.3 *First experiment: comparison of the six classifier accuracies for faultless (F) and damaged (D) classes*

Classifiers	Average		Worst case		Std.Dev.	
	F	D	F	D	F	D
LDC	99.16%	43.55%	96.43%	0.01%	1.34%	28.96%
QDC	99.73%	42.05%	99.44%	15.63%	0.19%	24.91%
MLP1	99.42%	43.92%	97.37%	16.04%	1.01%	23.45%
MLP2	99.60%	54.02%	98.87%	16.56%	0.39%	24.17%
CHC1	97.16%	76.19%	89.27%	52.49%	3.92%	17.44%
CHC2	96.92%	75.20%	90.09%	55.88%	3.43%	17.92%

Table 8.4 *Second experiment: comparison of the six classifier accuracies for faultless (F) and damaged (D) classes*

Classifiers	Average		Worst case		Std.Dev.	
	F	D	F	D	F	D
LDC	99.22%	55.15%	96.31%	8.77%	1.22%	25.74%
QDC	98.72%	62.52%	92.63%	5.58%	1.85%	26.41%
MLP1	98.52%	63.73%	91.82%	25.25%	2.34%	23.97%
MLP2	98.95%	62.93%	95.28%	22.59%	1.49%	23.97%
CHC1	90.79%	84.77%	83.16%	53.78%	6.11%	14.26%
CHC2	93.1%	79.50%	82.81%	47.70%	5.41%	19.55%

havior of the CHCs with that of the traditional multi-class classifiers.

Please note that in Table 8.3, like in the following ones, with MLP1 and MLP2 we indicate the MLPs trained with the DFs extracted, respectively, by LDC and QDC, and with CHC1 and CHC2 we identify the CHCs trained with the DFs extracted, respectively, by LDC and QDC.

Table 8.3, which is related to the first experiment, shows that, considering the damaged class, LDC and QDC present pretty poor performance; on the other hand, also MLPs do not perform better than the two statistical classifiers, while the two CHCs, which show pretty similar behaviors, increase for more than 21% the best average accuracy

Table 8.5 *Third experiment: comparison of the six classifier accuracies for faultless (F) and damaged (D) classes*

Classifiers	Average		Worst case		Std.Dev.	
	F	D	F	D	F	D
LDC	99.31%	60.38%	97.93%	12.63%	0.54%	28.69%
QDC	99.12%	72.13%	96.31%	19.04%	0.97%	26.69%
MLP1	98.47%	74.28%	96.08%	23.84%	1.38%	25.91%
MLP2	99.14%	69.19%	96.31%	24.10%	1.27%	25.01%
CHC1	87.31%	88.39%	84.20%	49.17%	1.87%	13.72%
CHC2	91.59%	86.24%	84.89%	34.11%	4.74%	19.01%

of all the multi-class classifiers. Furthermore, the CHCs improve the worst case for even more than 35% and the standard deviation of the traditional classifiers for at least 5%. In particular the best CHC outperforms the best traditional classifier for more than 22%, 39%, and 6%, respectively, for the average accuracy, the worst case and the standard deviation.

The CHCs present a slight reduction of the faultless class accuracy compared to the traditional classifiers; this reduction, however, can be considered negligible if compared to the improvement obtained for the damaged class accuracy.

It should also be noticed that if each classifier is tested with the damaged class used during the training process the obtained accuracy is pretty high (always over 90.00%), so the decrease in accuracy is basically due to the inability of the used classifiers, and in particular of multi-class classifiers, to generalize when one of the classes is not exhaustively sampled. For these reasons the CHCs result to be more stable compared to the multi-class classifiers.

Considering the damaged class, Table 8.4, which is related to the second experiment, shows that the best CHC improves the average accuracy of the traditional classifiers for more than 21%, the worst case for even more than 28% and the standard deviation of the traditional classifiers for at least 9%.

Table 8.5, which shows the results obtained in the third experiment, shows that the best CHC improves the average accuracy of the tradi-

tional classifiers for at least 14% and the worst case for even more than 25%. Furthermore CHCs improve the standard deviation for at least 11%.

Table 8.6 *Fourth experiment: comparison of the six classifier accuracies for faultless (F) and damaged (D) classes*

Classifiers	Average		Worst case		Std.Dev.	
	F	D	F	D	F	D
LDC	99.02%	60.34%	97.92%	21.27%	0.60%	26.64%
QDC	98.44%	79.69%	89.19%	20.48%	2.62%	27.15%
MLP1	98.50%	77.63%	96.54%	17.47%	1.17%	25.87%
MLP2	98.71%	77.36%	87.89%	4.25%	3.05%	27.23%
CHC1	86.14%	93.32%	83.74%	72.82%	1.51%	8.81%
CHC2	88.07%	91.97%	83.97%	66.30%	4.03%	13.59%

Table 8.7 *Fifth experiment: comparison of the six classifier accuracies for faultless (F) and damaged (D) classes*

Classifiers	Average		Worst case		Std.Dev.	
	F	D	F	D	F	D
LDC	99.58%	48.37%	98.85%	5.59%	0.37%	43.32%
QDC	99.19%	77.00%	98.39%	9.77%	0.48%	37.80%
MLP1	98.19%	82.45%	95.16%	49.18%	1.62%	21.24%
MLP2	99.50%	83.45%	99.08%	45.85%	0.25%	24.39%
CHC1	87.67%	92.43%	84.66%	65.00%	1.01%	13.81%
CHC2	86.99%	98.31%	83.74%	89.89%	1.71%	4.13%

Table 8.6 shows the results for the fourth experiment. In this table we can see that the best CHC improves the average accuracy of the traditional classifiers for at least 13%, the worst case for even more than 51% and the standard deviation of the multi-class classifiers for at least 17%.

From Table 8.7, which is related to the fifth experiment, we can notice that the CHC improves the average accuracy for more than 14%,

the worst case for even more than 40% and the standard deviation for at least 17%.

Finally, Table 8.8, which refers to the sixth experiment, shows that, once again, the best CHC improves the average accuracy of the traditional classifiers, even though only of 0.4%. Please note that in the sixth experiment (Table 8.8) we did not report the worst case and the standard deviation since there is only one combination that includes all the damaged classes.

Table 8.8 *Sixth experiment: comparison of the six classifier accuracies for faultless (F) and damaged (D) classes*

Classifiers	Average	
	F	D
LDC	99.65%	85.14%
QDC	98.50%	99.41%
MLP1	99.19%	98.27%
MLP2	99.54%	99.31%
CHC1	86.66%	99.08%
CHC2	92.39%	99.81%

8.4.3 Conclusions to the experiments to compare one-class with multi-class classifiers

In this section we have compared multi-class classifiers, such as statistical classifiers, namely LDC and QDC, and neural networks, namely, MLP, with a one-class classifier, namely, CHC.

From the shown results we can affirm that multi-class classifiers generally perform quite poorly with respect to the *unknown* faults and severities while the one-class CHC performs pretty better. Furthermore the CHC presents for the damaged class a very low standard deviation and a pretty good worst case compared with all the other classifiers.

Considering the damaged class, the difference in the behavior between the one-class classifier and traditional classifiers is more relevant in the first experiments than the last ones. This is due to the fact that as we increase the number of damaged classes used in the feature selection process, the feature selection process itself becomes more ac-

curate and selects more significant features. Thus multi-class classifiers seem to be more dependent on the result of the feature selection process as they work better when the feature selection process is performed more accurately. However, in practical cases, it is very frequent that the classification system will have to deal with *unknown* defects and severities. Thus the choice of one-class classifiers, which show a higher level of independence from the damaged samples used in the feature selection process, seems to be a more proper choice for real industrial environments.

8.5 Traditional one-class classifiers vs proposed one-class classifiers

8.5.1 Introduction to the experiments to compare traditional with proposed one-class classifiers

In this section we propose to investigate and compare a set of one-class classifiers, some well-known such as Gaussian, K-NN and 1-NN, and others proposed in this chapter, namely, CSC and CSS, which are based on the convex hull classifier (CHC) and the snake operator classifier (SOC). All these classifiers are deeply described in Chap. 3.

8.5.2 Experiments and results

We have compared five one-class classifiers, namely, Gaussian, K-NN, 1-NN, CSC, and CSS. To compare these classifiers, we apply them to "solve" the six experiments introduced above.

All the results shown in this section are computed as the average of 100 trails performed for each combination of each experiment, in order to guarantee more stable and reliable results.

For each of the six experiments we have performed the feature selection process to select the best two DFS. In Fig. 8.6 we can see the histogram representing the two DFs selected as either first or second over the six experiments and the number of times each of them has been selected. The values reported in this histogram represent the average over the 100 trails of each combination.

In Figs. 8.7-8.12 we compare all these classifiers, showing the ROC curve associated to each classifier in the six different experiments. Each figure represents a different experiment, thus Fig. 8.7 refers to the first experiment, Fig. 8.8 refers to the second, and so on.

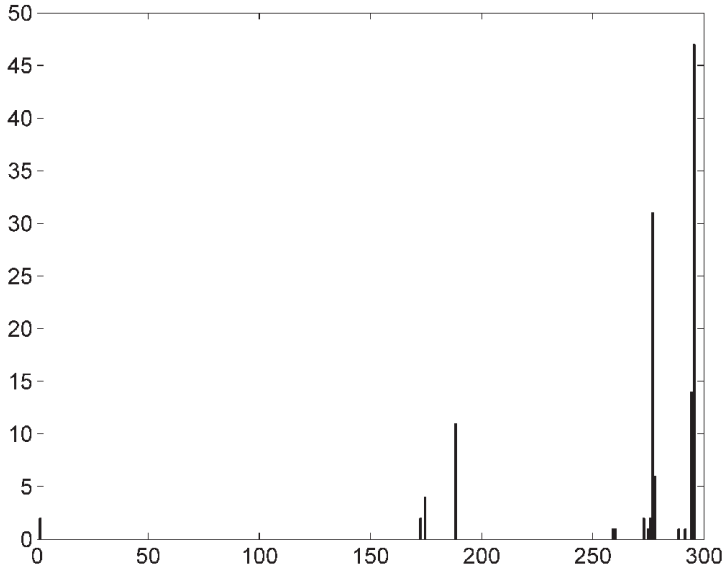


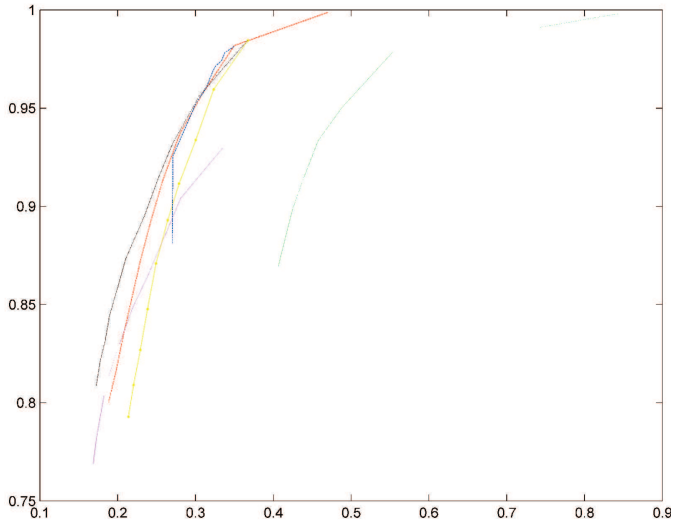
Figure 8.6 *Histogram representing the two DFs selected in the six experiments and the number of times each of them has been selected.*

In particular the x-axis represent the value 1-accuracy$_D$, where accuracy$_D$, which ranges in the interval [0,1], represents the accuracy on the damaged class, while the y-axis represent the accuracy on the faultless class in the range [0,1].

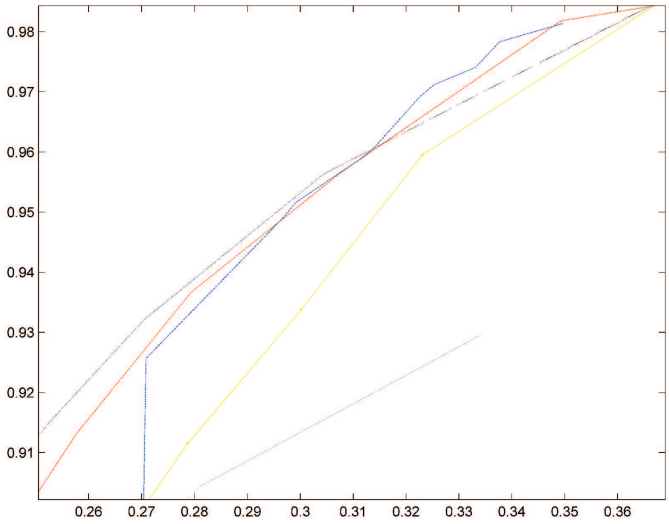
In the following, with reference to the CSC classifier, we will consider separately the sequence of convex hulls only, and the sequence of snake operators only. We will refer to the two sequences as CSC_{CHC} and CSC_{SOC}, respectively.

Please note that in the following figures not all the ROC curves representing the classifiers span all the ROC space owing to the limited range of classification accuracies achieved by the specific classifier.

As already stated, Fig. 8.7 refers to the first experiment, in which we can see that, while the K-NN, the 1-NN and the CSC_{SOC} classifiers perform pretty poorly, the Gaussian, the CSC_{CHC} and the CSS classifiers result to be the best. Actually, an appropriate selection between the CSC_{CHC} and the CSS classifiers allows us to achieve a performance higher than that obtained by the Gaussian classifier, as shown by the fact that the ROC curve representing the Gaussian classifier is always under the ROC curve of either of the two classifiers CSC_{CHC} and CSS.



(a)



(b)

Figure 8.7 *First experiment: (a) ROC curves: Gaussian (red), K-NN (yellow), 1-NN (green), CSS (blue), CSC_{CHC} (gray), CSC_{SOC} (magenta). (b) zoom of the most north-west part of the ROC space.*

Figure 8.8 refers to the second experiment, in which we can see that the 1-NN classifier performs pretty poorly, while the Gaussian, the CSC_{CHC} and the CSS classifiers achieve the best results. Actually, the differences among the three classifiers are negligible (the ROC curve of the Gaussian classifier is slightly higher) up to a point in which the proposed classifiers (CSC_{CHC} and CSS) outperform the Gaussian classifier.

Finally the K-NN and the CSC_{SOC} classifiers show intermediate performance between the worst and the best classifiers.

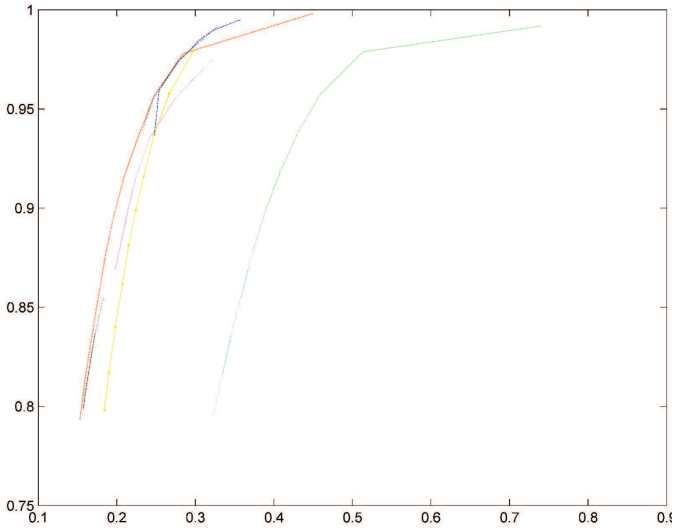
Figure 8.9 refers to the third experiment, where we can notice that once again the 1-NN classifier shows very bad results while the Gaussian, the CSC_{CHC} and the CSS classifiers obtain the best results. However, CSC_{CHC} is by far the best classifier. Finally the K-NN and the CSC_{SOC} classifiers show intermediate performance between the worst and the best classifiers.

As regards as the fourth experiment, Fig. 8.10 shows that the 1-NN classifier is again the worst. Besides the Gaussian classifier is no more among the best classifiers and its performance is comparable with those of K-NN and CSC_{SOC} classifiers. Finally, the CSC_{CHC} and CSS classifiers result to be by far the best.

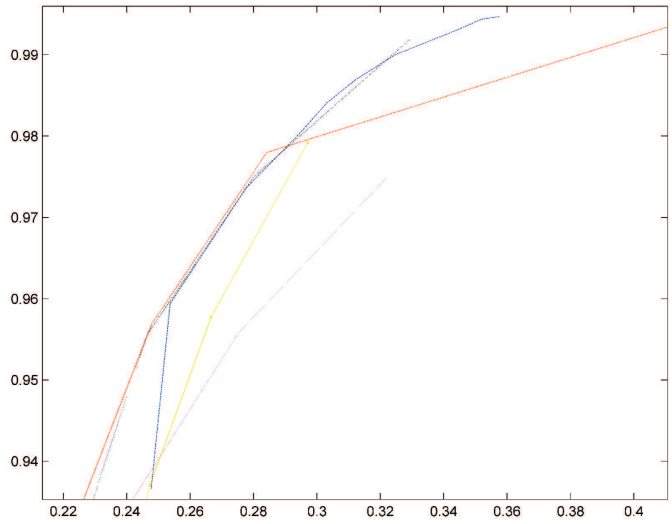
As far as the fifth experiment is concerned, in Fig. 8.11 the 1-NN classifier is still the worst classifier, the K-NN, the Gaussian, and the CSC_{SOC} classifiers obtain comparable, intermediate results (in particular, the Gaussian classifier achieves the worst results), and the CSC_{CHC} classifier achieves by far the best performance.

Finally Fig. 8.12 presents the results for the sixth and last experiment. The 1-NN classifier is still the worst classifier. The K-NN, the Gaussian, the CSC_{CHC} and CSC_{SOC} classifiers present quite similar, intermediate performance. On the other hand the CSS classifier significantly outperforms all the other classifiers, resulting to be by far the most accurate.

Summarizing the results thorough the six experiments, we can say that the 1-NN classifier is always the worst classifier. The K-NN and the CSC_{SOC} classifiers improve their performances presenting accuracy more similar to the Gaussian, CSC_{CHC}, and CSS classifiers as more damaged classes are used in the feature selection process. The Gaussian classifier is generally the best among the traditional one-class classifiers,

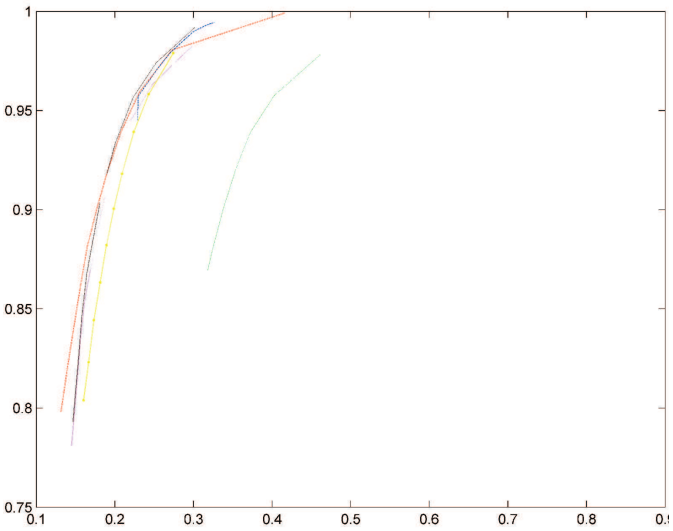


(a)

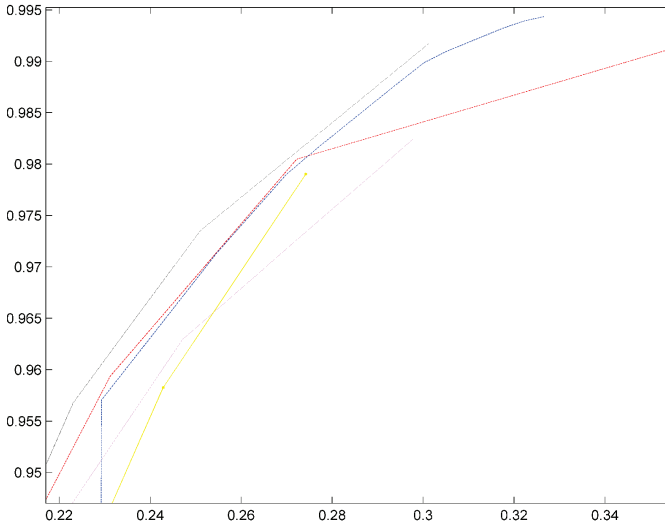


(b)

Figure 8.8 *Second experiment: (a) ROC curves: Gaussian (red), K-NN (yellow), 1-NN (green), CSS (blue), CSC_{CHC} (gray), CSC_{SOC} (magenta). (b) zoom of the most north-west part of the ROC space.*

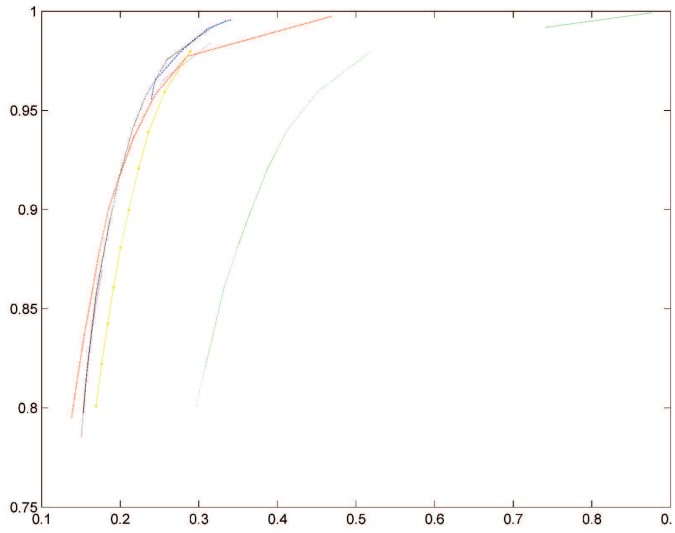


(a)

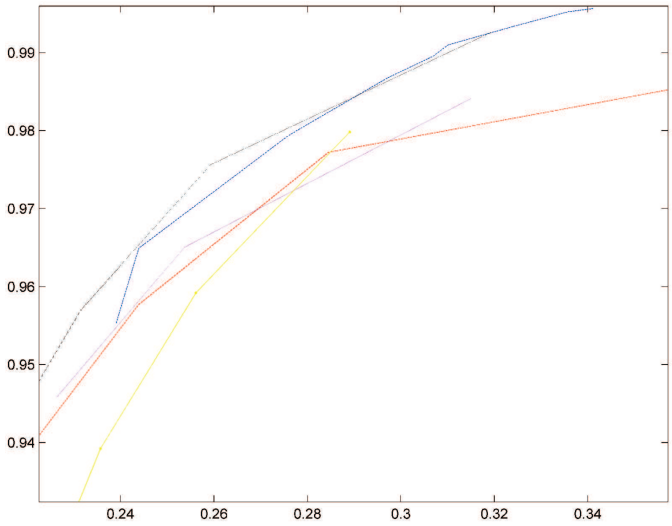


(b)

Figure 8.9 *Third experiment: (a) ROC curves: Gaussian (red), K-NN (yellow), 1-NN (green), CSS (blue), CSC_{CHC} (gray), CSC_{SOC} (magenta). (b) zoom of the most north-west part of the ROC space.*

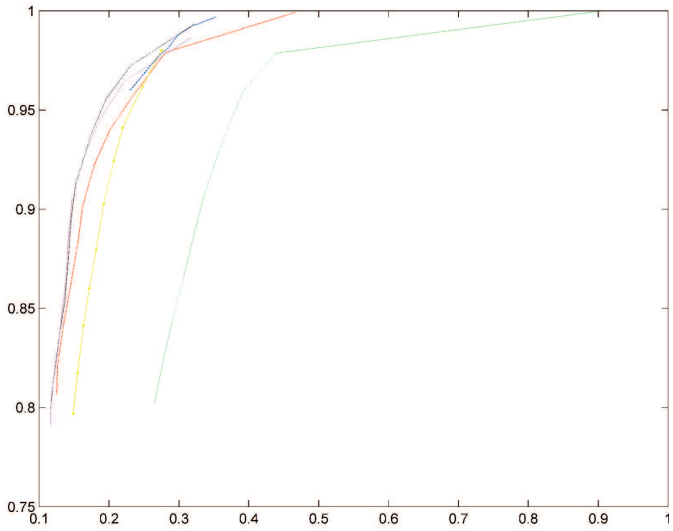


(a)

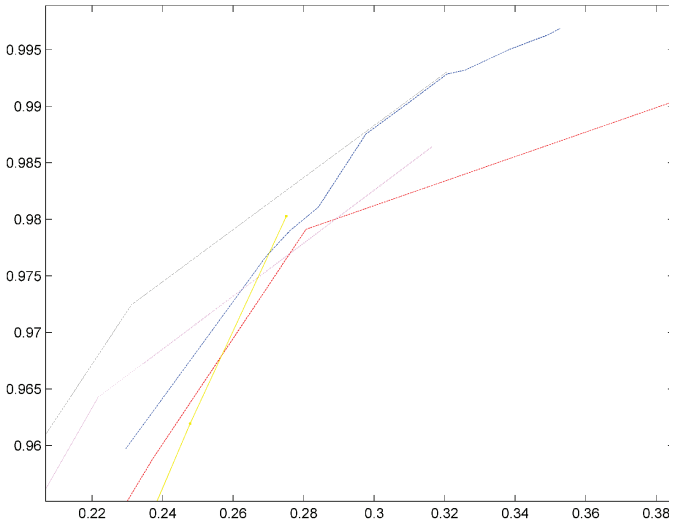


(b)

Figure 8.10 *Fourth experiment: (a) ROC curves: Gaussian (red), K-NN (yellow), 1-NN (green), CSS (blue), CSC_{CHC} (gray), CSC_{SOC} (magenta). (b) zoom of the most north-west part of the ROC space.*

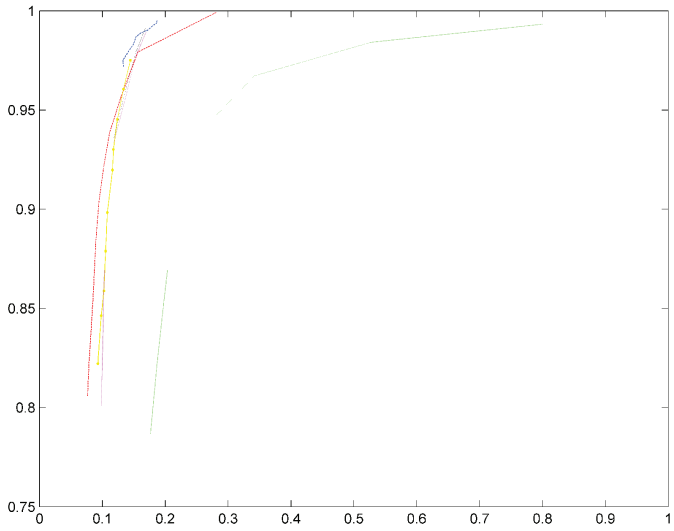


(a)

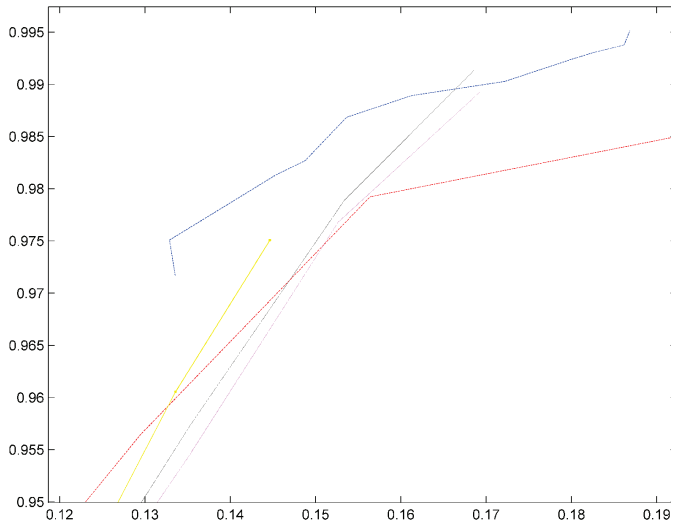


(b)

Figure 8.11 *Fifth experiment: (a) ROC curves: Gaussian (red), K-NN (yellow), 1-NN (green), CSS (blue), CSC_{CHC} (gray), CSC_{SOC} (magenta). (b) zoom of the most north-west part of the ROC space.*



(a)



(b)

Figure 8.12 *Sixth experiment: (a) ROC curves: Gaussian (red), K-NN (yellow), 1-NN (green), CSS (blue), CSC_{CHC} (gray), CSC_{SOC} (magenta). (b) zoom of the most north-west part of the ROC space.*

however, increasing the number of faulty classes for feature selection, it is outperformed by the two proposed one-class classifiers CSC_{CHC} and CSS. On the other hand, the CSC_{CHC} and CSS classifiers are always among the best classifiers throughout all the six experiments.

We can therefore claim that the two proposed one-class classifiers have proved to be a valuable alternative to the traditional one-class classifiers.

8.5.3 Conclusions to the experiments to compare traditional with proposed one-class classifiers

In this chapter we have dealt with diagnosis of rolling elements bearings based on vibration signals represented in the frequency domain by means of the FFT, registered by one accelerometer. We have coped with this problem as a classification problem.

We have proposed the use of the convex hull, usually adopted in application domains such as computer visualization, verification methods and computational geometry problems, and the snake operator, typically employed for image segmentation, as two one-class classifiers for rolling bearing fault diagnosis within CBM programs.

We have introduced two novel one-class classifiers, namely CSC_{CHC} and CSS, resulting from an appropriate integration of the convex hull and snake operator classifiers. CSC_{CHC} and CSS have been compared with traditional one-class classifiers, such as Gaussian, 1-NN and K-NN, in six experiments concerning the diagnosis of rolling bearing defects. The proposed classifiers have proved to be very accurate being able to achieve better results than traditional one-class classifiers in all cases but one in which the difference between the proposed classifiers and the Gaussian classifier (the best) is practically negligible.

We can therefore claim that the two proposed one-class classifiers represent a valuable alternative to the traditional one-class classifiers.

Chapter 9

Conclusions

The purpose of this work was to design and develop new methodologies, with high levels of accuracy and robustness to noise, to perform the detection, diagnosis and prognosis of defects on rolling elements bearings.

We used vibration signals recorded by four accelerometers on a mechanical device including rolling element bearings: the signals were collected both with all faultless bearings and after substituting one faultless bearing with an artificially damaged one. Four defects, namely, indentation on the inner raceway, indentation on the roll, sandblasting of the inner raceway, and unbalanced cage, and three levels of severity, namely, light, medium, and high, were considered.

This research has carried out a complete analysis of advanced soft computing techniques ranging from the multi-class and one-class classification to the combination strategies based on fusion and selection of classifiers.

This research has started from the present state of the art and has brought to the design and development of new maintenance methodologies to perform the prognosis and diagnosis of rotating machinery components. The high levels of accuracy and robustness to noise, shown by the results obtained in the performed experiments, prove the effectiveness of such methodologies, which can be thus profitably used within real-time condition-based maintenance programs.

Besides, we have also introduced two novel one-class classifiers resulting from an appropriate integration of the convex hull and snake operator classifiers. These new classifiers have proved to be very ac-

curate and thus to represent a valuable alternative to the traditional classifiers.

9.1 Future work

There are several interesting lines of research which arise from this Ph.D work and that should be more deeply analyzed.

It would be of interest to apply the proposed methodologies to other bearing fault data as well to extend the proposed study to the diagnosis and prognosis of more types of faults and levels of severity.

Furthermore, a very interesting problem is the one related to the balance of the data. Since more faultless samples than damaged ones are likely to be collected, the use of different class-balancing algorithms should be considered to increase the classification accuracy and the robustness to noise.

Finally, the introduction of cost-sensitive classifiers instead of balancing algorithms should be taken into account. This way we could further increase the performance of the presented classification systems/methodologies in terms of accuracy and robustness to noise.

Bibliography

[1] Ahlmann H., *The economic significance of maintenance in industrial enterprises*, Lund University, Lund Institute of Technology, Sweden, 1998.

[2] Allen D.M., *The relationship between variable selection and data augmentation and a method for prediction*, Technometrics, vol. 16, pp. 125–127, 1974.

[3] Al-Najjar B., *Impact of real-time measurements of operating conditions on effectiveness and accuracy of vibration-based maintenance policy: a case study in paper mill*, Journal of Quality in Maintenance Engineering, vol. 6, n. 4, pp. 275–287, 2000.

[4] Al-Najjar B., Alsyouf I., *Enhancing a company's profitability and competitiveness using integrated vibration-based maintenance: a case study*, Journal of European Operation Research, vol. 157, pp. 643–657, 2004.

[5] Al-Najjar B., *The lack of maintenance and not maintenance which costs: A model to describe and quantify the impact of vibration-based maintenance on company's business*, International Journal of Production Economics, vol. 107, pp. 260–273, 2007.

[6] Anderson J.A., Rosenfeld E., *Neurocomputing*, Foundations of Research, The MIT Press, Cambridge, Massachusetts, 1988.

[7] Arlot S., *A survey of cross-validation procedures for model selection*, Statistics Surveys, vol. 4, pp. 40–79, 2010.

[8] Bartlett P.L., Boucheron S., Lugosi G., *Model selection and error estimation*, Machine Learning, vol. 48, pp. 85–113, 2002.

[9] Barzilay O., Brailovsky V.L., *On domain knowledge and feature selection using a support vector machine*, Pattern Recognition Letters, vol. 20, n. 5, pp. 475–484, 1999.

[10] Barzilay O., Davidson I., Wagstaff K.L., *Constrained clustering. Advances in algorithms, theory, and applications*, CRC Press, 2009.

[11] Ben-David S., Lindenbaum M., *Learning distributions by their density levels: A paradigm for learning without a teacher*, Journal of Computer and System Sciences, vol. 55, n. 1, pp. 171-182, 1997.

[12] Bishop C.M., *Neural Networks for pattern recognition*, Clarendon Press, Oxford, 1995.

[13] Bishop C.M., *Pattern recognition and machine learning*, Springer, 2006.

[14] Bloch H.P., Geitner F.K., *Machinery failure analysis and troubleshooting*, Gulf Professional Publishing, 1997.

[15] Bonissone P., Goebel K., *When will it break? A hybrid soft computing model to predict time-to break margins in paper machines*, Proceedings of SPIE 47th Annual Meeting, International Symposium on Optical Science and Technology, vol. 4787, pp. 53-64, 2002.

[16] Brown K.Q., *Voronoi diagrams from convex hulls*, Information Processing Letters, vol. 9, n. 5, pp. 223–228, 1979.

[17] Chawla N.V., Bowyer K.W., Hall L.O., Kegelmeyer W.P., *SMOTE: Synthetic Minority Over-sampling Technique*, vol. 6, pp. 321–357, 2002.

[18] Chow C., *On optimum recognition error and reject tradeoff*, IEEE Transactions on Information Theory, IT-16,1, pp. 4146, 1970.

[19] Cococcioni M., D'Andrea E., Lazzerini B., Volpi S.L., *Short-time forecasting of renewable production energy in solar photovoltaic installations*, International Conference on Competitive and Sustainable Manufacturing, Products and Services (APMS'10), Como, Italy, 2010.

[20] Cococcioni M., Forte P., Menconi S., Sacchi C., *Rolling bearing monitoring using classification techniques*, 8th International Conference on Vibrations in Rotating Machines (SIRM'09), Vienna, 2009.

[21] Cococcioni M., Lazzerini B., Volpi S.L., *Rolling element bearing fault classification using soft computing techniques*, IEEE International Conference on Systems, Man, and Cybernetics (SMC'09), vol. 1, pp. 4926–4931, Hyatt Regency Riverwalk, San Antonio, TX, U.S., 2009.

[22] Cococcioni M., Lazzerini B., Volpi S.L., *Bearing condition monitoring using classifier fusion*, IASTED International Conference on Artificial Intelligence and Soft Computing (ASC'09), vol.1, pp. 131–137, Palma de Mallorca, Spain, 2009.

[23] Cococcioni M., Lazzerini B., Volpi S.L., *Automatic diagnosis of defects of rolling element bearings based on computational intelligence techniques*, IEEE 9th International Conference on Intelligent Systems Design and Applications (ISDA'09), vol. 1, pp. 970–975, Pisa, Italy, 2009.

[24] Cooley J.W., Tukey O.W., *An algorithm for the machine calculation of complex fourier series*, Mathematics of Computation, vol. 19, pp. 297–301, 1965.

[25] Cormen T.H., Leiserson C.E., Rivest R.L., Stein C., *Introduction to algorithms*, 2nd ed., The MIT Press and McGraw-Hill, 2001.

[26] Dasarathy B.V., Sheela B.V., *A composite classifier system design: concepts and methodology*, Proceedings of IEEE, vol. 67, n. 5, pp. 708–713, 1979.

[27] Dash M., Liu H., *Intelligent data analysis 1*, Feature Selection for Classification, 1997.

[28] Dekker R., Wildeman R E., Van Der Duyn Schouten F.A., *A review of multi-component maintenance models with economic dependence*, Mathematical Methods of Operational Research, vol. 45, n. 3, pp. 411–435, 1997.

[29] Devroye L., Wagner T.J., *The L1 convergence of kernel density estimates*, Annals of Statistics, vol. 7, pp. 1136–1139, 1979.

[30] Dietterich T.G., *Ensemble methods in machine learning*, in J. Kittler and F. Roli editors, Multiple Classifier Systems, vol. 1857 of Lecture Notes in Computer Science, Cagliari, Italy, Springer.

[31] Duda R., Hart P., *Pattern classification and scene analysis*, John Wiley & Sons, New York, 1973.

[32] Duda R.O., Hart P.E., Stork D.G., *Pattern classification*, 2nd ed., John Wiley & Sons, New York, 2001.

[33] Duin R.P.W., Juszczak P., Paclik P., Pekalska E., de Ridder D., Tax D.M.J., Verzakov S., *PRTools4, A Matlab Toolbox for Pattern Recognition*, Version 4.1, August 2007.

[34] Dunna R.A., *A Statistical Approach to neural networks for pattern recognition*, Wiley-Interscience, New Jersey, 2007.

[35] Engel S.J., Gilmartin B.J., Bongort K., Hess A., *Prognostics, the real issues involved with predicting life remaining*, IEEE Aerospace Conference, pp. 457-469, 2000.

[36] Fausett L., *Fundamentals of neural networks*, Prentice Hall Inc., Englewood Cliffs, N.J., 1994.

[37] Fawcett T., Niculescu-Mizil A., *PAV and the ROC convex hull*, Machine Learning, vol. 68, n. 1, pp. 97-106, 2007.

[38] Fawcett T., *An introduction to ROC analysis*, Pattern Recognition Letters, vol. 27, n. 8, pp. 861–874, 2006.

[39] Fisher R.A., *The use of multiple measurements in taxonomic problems*, Annals of Eugenics, vol. 7, pp. 179-188, 1936.

[40] Frank A., Asuncion A., *UCI Machine Learning Repository*, http://archive.ics.uci.edu/ml, Irvine, CA: University of California, School of Information and Computer Science, 2010.

[41] Fu K.-S., *Syntactic pattern recognition and applications*, Englewood Cliffs, N.J, 1982.

[42] Garcìa V., Mollineda R.A., Sànchez J.S., *On the k-NN performance in a challenging scenario of imbalance and overlapping*, Pattern Analysis & Applications, vol. 11, pp. 269-280, 2008.

[43] Geisser S., *The predictive sample reuse method with applications*, Journal of the American Statistical Association, vol. 70, pp. 320–328, 1975.

[44] Gelfand I.M., Kapranov M.M., Zelevinsky A.V., *Discriminants, resultants, and multidimensional determinants (Mathematics: Theory & Applications)*, Birkauser, Boston, 1994.

[45] Giacinto G., Roli F., *Design of effective neural network ensembles for image classification processes*, Image Vision and Computing Journal, vol. 19, n. 9–10, pp. 699–707, 2001.

[46] Goebel K.F., Bonissone P.P., *Prognostic information fusion for constant load systems*, Proceedings of the 7th Annual Conference on Information Fusion, 2005.

[47] Hassoun M.H., *Fundamentals of artificial neural networks*, The MIT Press, Cambridge, Massachusetts, 1995.

[48] Hakan A., *Optimal resampling and classifier prototype selection in classifier ensembles using genetic algorithms*, Pattern Analysis & Applications, vol. 7, n. 3, pp. 285–295, 2004.

[49] Hastie T., Tibshirani R., Friedman J., *The elements of statistical learning*, Springer Series in Statistics, Springer-Verlag, New York, Data mining, inference, and prediction, 2nd ed., 2009.

[50] Haykin S., *Neural networks. A comprehensive foundation*, Macmillan College Publishing Company, New York, 1994.

[51] He H., Garcia E.A., *Learning from imbalanced data*, IEEE Transactions on Knowledge and data Engineering, vol. 21, n. 9, 2009.

[52] Hebb D.O., *The organization of behavior*, Wiley, New York, 1949.

[53] Heng A., Zhang S., Tan A.C.C, Mathew J., *Rotating machinery prognostics: State of the art, challenges and opportunities*, Mechanical Systems and Signal Processing, vol. 23, pp.724-739, 2009.

[54] Ho T.K., Hull J.J., Srihari S.N., *Decision combination in multiple classifier systems*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 16, n. 1, pp. 66-75, 1994.

[55] Hochbaum D., Shmoys D., *A best possible heuristic for the k-center problem*, Mathematics of Operations Research, vol. 10, n. 2, pp. 180-184, 1985.

[56] Huang D.-S., Heutte L., Loog M., Lee H.-H., Nguyen N.-T., Kwon J.-M., *Bearing diagnosis using time-domain features and decision tree advanced intelligent computing theories and applications. With aspects of artificial intelligence*, Lecture Notes in Computer Science, Springer Berlin/Heidelberg, vol. 4682, pp. 952–960, 2007.

[57] Jack L.B., Nandi A.K., *Fault detection using support vector machines and artificial neural network, augmented by genetic algorithms*, Mechanical System and Signal Processing, vol. 16, n. 2–3, pp. 373–390, 2002.

[58] Janjarasjitt S., Ocak H., Loparo K.A., *Bearing condition diagnosis and prognosis using applied nonlinear dynamical analysis of machine vibration signal*, Journal of Sound and Vibration, vol. 317, n. 1–2, pp. 112–126, 2008.

[59] Japkowicz N., *Concept-learning in the absence of counter-examples: an autoassociation-based approach to classification*, Ph.D thesis, New Brunswick Rutgers, The State University of New Jersey, 1999.

[60] Japkowicz N., Stephen S., *The class imbalance problem: a systematic study*, Intelligent Data Analysis, vol. 6, n. 5, pp. 429-449, 2002.

[61] Jardine A.K.S., Lin D., Banjevic D., *A review on machinery diagnostics and prognostics implementing condition based maintenance*, Mechanical Systems and Signal Processing, vol. 20, n. 7, pp. 1483–1510, 2006.

[62] Jiang M.F., Tseng S.S., Su C.M., *Two-phase clustering process for outliers detection*, Pattern Recognition Letters, vol. 22, n. 6-7, pp. 691-700, 2001.

[63] Jolliffe I.T., *Principal component analysis*, Series: Springer Series in Statistics, 2nd ed., Springer, New York, 2002.

[64] Juszczak P., *Learning to recognise. A study on one-class classification and active learning*, Ph.D thesis, Delft University of Technology, 2006.

[65] Kass M., Witkin A. Terzopoulos D., *Snakes: active contour models*, International Journal of Computer Vision, vol. 1, n. 4, pp. 321-331, 1988.

[66] Kim H-E., Tan A.C.C., Mathew J., Kim E.Y.H., Choi B-K., *Prognosis of bearing failure on health state estimation*, 4th World Congress on Engineering Asset Management, Athens, Greece, 2009.

[67] Kittler J., *Feature selection and extraction*, in Handbook of Pattern Recognition and Image Processing, Academic Press, New York, Chap. 3, 59, 1986.

[68] Klein R., Ingman D., Braun S., *Non-stationary signals: phase-energy approach-theory and simulations*, Mechanical Systems and Signal Processing, vol. 15, n. 6, pp. 1061-1089, 2001.

[69] Knorr E., Ng R., Tucakov V., *Distance-based outliers: algorithms and applications*, VLDB Journal, Very Large Data Bases, vol. 8, n. 3-4, pp. 237-253, 2000.

[70] Kohavi R., Provost F., *Glossary of terms*, Editorial for the Special Issue on Applications of Machine Learning and the Knowledge Discovery Process, vol. 30, pp. 2–3, 1998.

[71] Koppel M., Schler J., *Authorship verification as a one-class classification problem*, International Conference on Machine Learning, pp. 489-495, 2004.

[72] Kotsiantis S., Pintelas P.E., *Mixture of expert agents for handling imbalanced data sets*, Annals of Mathematics, Computing & Tele-Informatics, vol. 1, n. 1, pp. 46–55, 2003.

[73] Kuncheva L.I., *Combining pattern classifiers: methods and algorithms*, Wiley Interscience, New Jersey, USA, 2004.

[74] Kuncheva L.I., Bezdek J.C., Duin R.P.W., *Decision templates for multiple classifier fusion: an experimental comparison*, Pattern Recognition, vol. 34, n. 2, pp. 299–314, 2001.

[75] Larson S.C., *The shrinkage of the coefficient of multiple correlation*, Journal of Educational Psychology, vol. 22, pp. 45–55, 1931.

[76] Lazzerini B., Volpi S.L., *Noise assessment in the diagnosis of rolling element bearings*, International Conference on Intelligent Computing and Cognitive Informatics (ICICCI'10), vol. 1, pp. 227–230, Kuala Lumpur, Malaysia, 2010.

[77] Li B., Chow M.-Y, Tipsuwan Y., Hung J.C., *Neural-network-based motor rolling bearing fault diagnosis*, IEEE Transactions Industrial Electronics, vol. 47, n. 5, pp. 1060–1069, 2000.

[78] Li C.J., Wu S.M., *On-line detection of localized defects in bearings by pattern recognition analysis*, IASME Journal of Engineering for Industry III, pp. 331–336, 1989.

[79] Li H., Zheng H., Tang L., *Wigner-Ville Distribution based on EMD for faults diagnosis of bearing*, Lecture Notes in Computer Science, Springer Berlin/Heidelberg, vol. 4223, pp. 803–812, 2006.

[80] Liu T.I., Singonahalli J.H., Iyer N.R., *Detection of roller bearing defects using expert system and fuzzy logic*, Mechanical Systems and Signals Processing, vol. 10, n. 5, pp. 595–614, 1996.

[81] Liu C., Wechsler H., *Robust coding schemes for indexing and retrieval from large face databases*, IEEE Transactions on Image Processing, vol. 9, no. 1, 2000, pp. 132–136.

[82] Liu H., Motoda H., *Computational methods of feature selection*, Chapman & Hall-CRC, 2008.

[83] Looney C.G., *Pattern recognition using neural networks. Theory and algorithms for engineers and scientists*, Oxford University Press, Oxford, 1997.

[84] Manevitz L., Yousef M., *One-class SVMs for document classification*, Journal of Machine Learning Research, vol. 2, pp. 139-154, 2001.

[85] McCulloch W.S., Pitts W., *A logical calculus of ideas immanent in nervous activity*, vol. 5, pp. 115–133, 1943, reprinted in Anderson and Rosenfeld, 1988.

[86] McInerney T., Terzopoulos D., *Deformable models in medical image analysis: a survey*, Medical Image Analysis, vol. 1, n. 2, pp. 91-108, 1996.

[87] Micheli-Tzanakou E., *Supervised and unsupervised pattern recognition - feature extraction and computational*, CRC Press, New York, 2000.

[88] Mosteller F., Tukey J.W., *Data analysis, including statistics*, in The Handbook of Social Psychology, vol. 2, 2nd ed., eds. G. Lindzey and E. Aronson, Reading, Mass.: Addison-Wesley, 1968.

[89] Moya M.R., Koch M.W., Hostetler L.D., *One-class classifier networks for target recognition applications*, World congress on neural networks, International Neural Network Society, INNS, pp. 797–801, Portland, OR, 1993.

[90] Muller A., Crespo Marquez A., Iung B., *On the concept of e-maintenance: Review and current research*, Reliability Engineering and System Safety, vol. 93, pp. 1165–1187, 2008.

[91] Nguyen N., Lee H., *Bearing fault diagnosis using adaptive network based fuzzy inference system*, International Symposium on Electrical & Electronics Engineering, HCM City, Vietnam, 2007.

[92] Ocak H., Loparo K.A., Discenzo F.M., *Online tracking of bearing wear using wavelet packet decomposition and probabilistic modeling: A method for bearing prognostics*, Journal of Sound and Vibration, vol. 302, n. 4–5, pp. 951–961, 2007.

[93] Pan Y., Chen J., Guo L., *Robust bearing performance degradation assessment method based on improved wavelet packet-support vector data description*, Mechanical Systems and Signal Processing, vol. 23, n. 3, pp. 669–681, 2009.

[94] Pao Y.-H., *Adaptive pattern recognition and neural networks*, Addison-Wesley, Reading, Massachusetts, 1989.

[95] Parra L., Deco G., Miesbach S., *Statistical independence and novelty detection with information preserving nonlinear maps*, Neural Computation, vol. 8, pp. 260-269, 1996.

[96] Parzen E., *On the estimation of a probability density function and mode*, Annals of Mathematical Statistics, vol. 33, pp. 1065-1076, 1962.

[97] Patterson D.W., *Artificial neural networks. Theory and applications*, Prentice Hall, Simon & Schuster, Singapore, 1996.

[98] Pearson K., *On lines and planes of closest fit to systems of points in space*, Philosophical Magazine, vol. 2, n. 6, pp. 559–572, 1901.

[99] Pekalska E., Tax D.M.J., Duin R.P.W., *One-class LP classifier for dissimilarity representations*, Neural Information Processing Systems, pp. 761-768, 2003.

[100] Prabhu R., *Rolling bearing diagnostics*, IndoUS Symposium on Emerging Trends in Vibration and Noise Engineering, New Delhi, pp. 311-320, 1996.

[101] Pusey H.C., Roemer M.J., *An assessment of turbo machinery condition monitoring and failure prognosis technology*, The Shock and Vibration Digest, vol. 31, pp. 365371, 1999.

[102] Rahnamayan S., Tizhoosh H.R., Salama M.M.A., *Automated snake initialization for the segmentation of the prostate in ultrasound images*, Image Analysis and Recognition, Lecture Notes in Computer Science, Springer Berlin/Heidelberg, vol. 3656, pp. 930–937, 2005.

[103] Rao B.V.A., Swarnamani S., Varghese G.V., *Studies on a test rig to check defective and spurious ball and roller bearings*, National Conference on Industrial Tribology, Bombay, India, 1986.

[104] Rastrigin L.A., Erenstein R.H., *Method of Collective Recognition*, Energoizdat, Moscow (in Russian), 1991.

[105] Reif Z., Lai M.S., *Detection of developing bearing failures by means of vibration*, ASME Design Eng Div (Publ) DE, vol. 18, n. 1, pp. 231-236, 1989.

[106] Raynor W.J.J., *The International dictionary of artificial intelligence*, Glenlake Publishing Company, Ltd., Chicago, London, New Delhi, Amacom, American Management Association.

[107] Ripley B.D., *Pattern Recognition and neural networks*, Cambridge University Press, 2007.

[108] Rojas R., *Neural networks. A systematic introduction*, Springer, Berlin, 1995.

[109] Rojas A., Nandi A.K., *Detection and classification of rolling-element bearing faults using support vector machines*, IEEE Workshop on Machine Learning for Signal Processing, vol. 12, pp. 153–158, 2005.

[110] Rojas A., Nandi A.K., *Practical scheme for fast detection and classification of rolling-element bearing faults using support vector machines*, Mechanical Systems and Signal Processing, vol. 20, n. 7, pp. 1523–1536, 2006.

[111] Rosenblatt F., *Principles of neurodynamics: perceptrons and the theory of brain mechanisms*, Spartan Books, Washington D.C., 1962.

[112] Rudemo M., *Empirical choice of histograms and kernel density estimators*, Scandinavian Journal of Statistics, vol. 9, pp. 65–78, 1982.

[113] Sain S.R., Gray H.L., Woodward W.A., Fisk M.D., *Outlier detection from a mixture distribution when training data are unlabeled*, Bulletin of the Seismological Society of America, vol. 89, pp. 294-304, 1999.

[114] Samanta B., Al-Balushi K.R., Al-Araimi S.A., *Bearing fault detection using artificial neural networks and genetic algorithm*, Journal on Applied Signal Processing, vol. 4, n. 3, pp. 366–377, 2004.

[115] Samanta B., Al-Balushi K.R., Al-Araimi S.A., *Artificial neural networks and genetic algorithm for bearing fault detection*, Soft Computing, vol. 10, n. 3, pp. 264–271, 2006.

[116] Shao Y., Nezu K., *Prognosis of remaining bearing life using neural networks*, Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, vol. 214, n. 3, pp. 217–230, 2000.

[117] Srivastava M.S., Carter E.M., *Applied multivariate statistics*, North Holland Amsterdam, 1983.

[118] Stone M., *Asymptotics for and against cross-validation*, Biometrika, vol. 64, n. 1, pp. 29–35, 1977.

[119] Sugumaran V., Ramachandran K.I., *Automatic rule learning using decision tree for fuzzy classifier in fault diagnosis of roller bearing*, Mechanical Systems and Signal Processing, vol. 21, n. 5, pp. 2237–2247, 2007.

[120] Sugumaran V., Muralidharan V., Ramachandran K.I., *Feature selection using decision tree and classification proximal support vector machine for fault diagnostic of roller bearing*, Mechanical System and Signal Processing, vol. 2, n. 2, pp. 930–942, 2007.

[121] Sun W., Chen J., Li J., *Decision tree and PCA-based fault diagnosis of rotating machinery*, Mechanical Systems and Signal Processing, vol. 21, n. 3, pp. 1300–1317, 2007.

[122] Sun A., Wong A.K.C., Kamwl M.S., *Classification of imbalanced data: a review*, International Journal of Pattern Recognition and Artificial Intelligence, vol. 23, n. 4, pp. 687–719, 2009.

[123] Sutherland H., Repoff T., House M., Flickinger G., *Prognostics, a new look at statistical life prediction for condition-based maintenance*, GE Global Research, New York, Tech. Rep. 2002GRC347, February 2003.

[124] Tarassenko L., Hayton P., Brady M., *Novelty detection for the identification of masses in mammograms*, International Conference on Artificial Neural Networks, pp. 442-447, 1995.

[125] Tandon N., Choudhury A., *A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings*, Tribology International, vol. 32, n. 8, pp. 469–480, 1999.

[126] Tax D.M.J., Duin R.P.W., *Support vector domain description*, Pattern Recognition Letters, vol. 20, n. 11–13, pp. 1191–1199, 1999.

[127] Tax D.M.J., *One-class classification*, Ph.D thesis, Delft University of Technology, 2001.

[128] Tran V.T., Yang B.-S., Oh M.-S., Tan A.C.C., *Machine condition prognosis based on regression trees and one-step-ahead prediction*, Mechanical Systems and Signal Processing, vol. 22, pp.1179-1193, 2008.

[129] Tsang A.H.C., *Condition-based maintenance: tools and decision making*, Journal of Quality Maintenance Engineering, vol. 1, n. 3, pp. 3–17, 1995.

[130] Tse P.W., Peng Y.H., Yam R., *Wavelet analysis and envelope detection for rolling element bearing fault diagnosis - Their effectiveness and flexibilities*, Journal of Vibration and Acoustics, vol. 123, n. 3, pp. 303–310, 2001.

[131] Toussaint G.T., *Bibliography on estimation of misclassification*, IEEE Transactions on Information Theory, vol.20, pp.472–479, 1974.

[132] Tubbs J.D., Alltop W.O., *Measures of confidence associated with combining classification rules*, IEEE Transactions on Systems, Man, and Cybernetics, vol. 21, n. 28, pp. 690-692, 1991.

[133] Ullman N., *Elementary statistics, an applied approach.*, Wiley and Sons, 1978.

[134] Vachtsevanos G., Lewis F., Roemer M., Hess A., Wu B., *Intelligent fault diagnosis and prognosis for engineering systems*, Wile, Hoboken, NJ, 2006.

[135] Valdez-Flores C., Feldman R.M., *A survey of preventive maintenance models for stochastically deteriorating single-unit systems*, Naval Research Logistics, vol. 36, n. 4, pp. 419–446, 1989.

[136] Van Der Duyn Schouten F., *Maintenance policies for multicomponent systems*, in Ozekici, S. (Ed.), Reliability and maintenance

of complex systems, NATO ASI series, 154, Springer, Berlin, Proceedings of the NATO Advanced Study Institute on Current Issues and Challenges in the Reliability and Maintenance of Complex Systems, Kemer-Antalya, Turkey, pp. 117–136, 1995.

[137] Vapnik V.N., *Statistical Learning Theory*, Wiley, 1998.

[138] Volpi S.L., *Introduction to Matlab and PRTools*, ed. S.E.U., Pisa 2010.

[139] Volpi S.L., *Design and realization of the module for the extraction of the mediastinum and the lung area from DICOM images in a lung CAD system for CT exams*, Master degree thesis (in Italian), University of Pisa, 2007.

[140] Volpi S.L., Antonelli M., Lazzerini B., Marcelloni F., Stefanescu D.C., *Segmentation and reconstruction of the lung and the mediastinum volumes in CT images*, IEEE 2nd International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL), vol. 1, pp. 1–6, Bratislava, Slovak Republic, 2009.

[141] Volpi S.L., Lazzerini B., Stefanescu D., *Time evolution analysis of bearing faults*, IASTED International Conference on Intelligent Systems and Control (ISC'09), Cambridge, MA, U.S., 2009.

[142] Volpi S.L., Cococcioni M., Lazzerini B., Stefanescu D., *Rolling element bearing diagnosis using convex hull*, IEEE World Congress on Computational Intelligence (WCCI'10), vol. 1, pp. 1–8, Barcelona, Spain, 2010.

[143] Wang W.J., Chen J., Wu X.K., Wu Z.T., *The application of some non-linear methods in rotating machinery fault diagnosis*, Mechanical Systems and Signal Processing, vol. 15, n. 4, pp. 697–705, 2001.

[144] Wasserman P.D., *Advanced Methods in Neural Computing*, Van Nostrand Reinhold, USA, 1993.

[145] Webb A., *Statistical Pattern Recognition*, John Wiley & Sons, New York, 2002.

[146] Widodo A., Yang B., *Support vector machine in machine condition monitoring and fault diagnosis*, Mechanical Systems and Signal Processing, vol. 21, n. 6, pp. 2560–2574, 2007.

[147] Williams J.H., Davies A., Drake P.R., *Condition-based maintenance and machine diagnostics*, Chapman & Hall, London, 1994.

[148] Withey D.J., Pedrycz W., Koles Z.J., *Dynamic edge tracing: Boundary identification in medical images*, Computer Vision and Image Understanding, vol. 113, n. 10, pp. 1039–1052, 2009.

[149] Woods K., Kegelmeyer W.P., Bowyer K., *Combination of multiple classifiers using local accuracy estimates*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, n. 4, pp. 405-410, 1997.

[150] Xu Z., Xuan J., Shi T., Wu B., Hu Y., *A novel diagnosis method of bearing on improved fuzzy ARTMAP and modified distance discriminant technique*, Expert Systems with Applications, vol. 36, pp. 11801–11807, 2009.

[151] Xu L., Krzysak A., Suen C.Y., *Methods of combining multiple classifiers and their application to handwriting recognition*, IEEE Transactions on Systems, Man, and Cybernetics, vol. 22, n. 3, pp. 418–435, 1992.

[152] Xu R., Wunsch II D.C., *Clustering*, John Wiley & Sons, Inc., Hoboken, New Jersey, 2009.

[153] Xue F., Bonissone P., Varma A., Yan W., Eklund N., Goebel K., *An instance-based method for remaining useful life estimation for aircraft engines*, Journal of Failure Analysis and Prevention, vol. 8, pp. 199-206, 2008.

[154] Yao X., Xie X., Fu M., Marcus S.I., *Optimal joint preventative maintenance and production policies*, Naval Research Logistics, 2005.

[155] Ypma A., *Learning methods for machine vibration analysis and health monitoring*, Ph.D thesis, Delft University of Technology, 2001.

[156] Ypma A., Duin R., *Support objects for domain approximation*, International Conference on Artificial Neural Networks, pp. 719-724, Springer, Berlin, 1998.

[157] Yang J., Zhang Y., Zhu Y., *Intelligent fault diagnosis of rolling element bearing based on SVMs and fractal dimension*, Mechanical Systems and Signal Processing, vol. 21, n. 5, pp. 2012-2024, 2007.

[158] Zarei J., Poshtan J., *Bearing fault detection using wavelet packet transform of induction motor stator current*, Tribology International, vol. 40, n. 5, pp. 763–769, 2007.

[159] Zhang Z., Wenzhi L., Shen M., *Active learning of support vector machine for fault diagnosis of bearings*, Lecture Notes in Computer Science, vol. 3973, pp. 390–395, 2006.

List of Figures

List of Tables