POLITECNICO DI TORINO

Electronic Engineering Master Degree
Sistemi Elettronici Course

Master Degree Thesis

Creation of a Machine Learning model for the Predictive Maintenance of an engine equipped with a rotating shaft

Supervisor:
Prof.ssa Tania Cerquitelli

Company Tutor:
Dott.ssa Paola Dal Zovo

Candidate:
Manfredi Manfrè

March 2020

Summary

One of the most promising applications of Industry 4.0 enabling technologies concerns the creation of systems capable of providing condition-based and predictive maintenance services. This thesis work deals with the introduction of the objectives of these services, their difficulties and known problems, and the solutions offered by the literature. It also describes the design and implementation of a system capable of detecting vibrations on a rotating shaft of an electric motor. This solution is based on a Data-driven approach, using an accelerometer and combining machine learning models to determine the operating status of the machine and report any anomaly. Particular attention is paid to the preprocessing of data to limit the calculation costs and increase the speed of execution while maintaining high reliability.

Table of contents

Chapter 1

Introduction

Regarding the total costs for the production of goods, those related to maintenance constitute from 15% up to 40% [1].
In particular, the main source of loss lies in all those non-fixed maintenance activities, namely, those interventions that aim to solve problems related to sudden breakdowns of machines and blocking in relation to production.
Analysis of the maintenance costs indicate that the same intervention entails a cost approximately tripled when it is performed unexpectedly compared to when it is programmed in advance [2].

It is therefore not surprising that with the advent of *Industry 4.0*, which is expected to be the fourth industrial revolution, one of the sectors where many investments are being made and in which the research is very active is precisely that of maintenance.
The goal is to be able to deduce the state of health of the machines and their critical components and use this information to plan when to intervene with replacements or repairs.

Expected earnings are manifold:

- Sudden failures due to worn components and the costs deriving from the consequent corrective maintenance are avoided;
- The times in which the machines are operating are maximized, increasing production and efficiency;
- You save on the components, using each of them as much as possible, replacing them only when its residual useful life is now close to zero;
- Warehouse occupation with spare parts is reduced, which can be ordered when the level of degradation of machines or components exceeds a certain threshold.

Enabling technologies of Industry 4.0 promise to lend themselves very well to the implementation of solutions for such a problem.
The general resolution scheme, which seems consolidated, is: use sensors positioned on the machines to collect relevant information, to communicate this information in real-time to an aggregation and analysis center, and to apply machine learning models to extrapolate useful information regarding the state of health of the machines, which will be used to support maintenance decisions and plans.

However, some problems and difficulties that are encountered when moving to the actual implementation of these systems in real scenarios are equally well known. Among them are the following:

- The high percentage of noise present in the data collected by the sensors in a context such as that of industrial production;
- The computational cost required for machine learning applications on a large amount of data;
- The need for historical data with the association to the relative state of health from which the model manages to perform learning and the strong dependence on the human factor that must certify the correctness of this labeling.

In this thesis, the problem is discussed and a possible solution is presented.

Goal and Contribution

The project aims to demonstrate the possibilities and advantages of the application of the Internet of Things and machine learning technologies in the industrial sector and to present possible implementations for a real case.

Specifically, the contribution of this thesis is the design and implementation of an anomaly recognition and classification system. The system is applied to a functioning machine, on which it is possible to intervene easily in order to bring it from conditions of normal activity to states of anomaly and vice versa.
The solution is based on the collection of acceleration data on the 3 axes from a sensor positioned on the machine to deduce the state of health. The sensor used is composed of low cost and easy to find components.
Great attention has been paid to the processing of raw data and to the way of creating simple functions for the machine learning algorithms used.
The proposed learning model is able to detect an anomaly state by requesting only examples of data collected over normal operation during training and also allows a

maintenance technician to associate a particular type of anomaly a posterior with input signals, so as to make it possible to classify that type of specific anomaly if it occurs in the future.

The thesis project was carried out in collaboration with Santer Reply SpA, which is a provider of end-to-end solutions and consultancy in the IT sector, with special focus on IoT.

Composition Structure

The rest of the thesis is structured as follows.

The second chapter provides a description of the context in which the thesis project is located, thus presenting the concepts of Industry 4.0 and maintenance.

The third chapter reports the related works, highlighting the possible approaches, the common problems and the solutions proposed by the literature to create systems for detecting, classifying and predicting machine failures. In this chapter, in fact, the concepts of data analytics and machine learning will be explained, with a particular focus on existing classification techniques.

The fourth chapter, the overall architecture, the individual components of the proposed solution, the data provided, their structure and the technological tools used to construct the classification models are analyzed.

The fifth chapter describes in detail the project carried out in detail. It is divided into three parts, which respectively represent the different phases of the work: in the first one, we analyze different methodologies with which to carry out the pre-processing of raw data, that is preliminary treatment useful to better prepare the received dataset, the second and the third part instead deal with the treatment of different types of models developed, with the common purpose of creating a system of help in predictive maintenance.

The sixth chapter describes future developments, where a possible solution to the problems encountered during the work carried out in the fifth chapter is explained. Furthermore, a program for the validation of the system on operating machines through the use of an industrial sensor is presented.

Lastly the seventh chapter presents the overall conclusions.

Chapter 2

Background

2.1 Industry 4.0

Industry is the economy branch that deals with material assets, in a mechanical and automatic way. Over the years, technology innovation has led to huge and quick changes in the industrial sector, leading to the occurrence of what we define as revolutions. The term "Industry 4.0" is used for the upcoming industrial revolution, which is about to take place right now. This industrial revolution has been preceded by three other industrial revolutions in the history of mankind. The first industrial revolution was the introduction of mechanical production facilities, starting in the second half of the 18th century and being intensified throughout the entire 19th century. From the 1870s on, electrification and the division of labour (i.e. Taylorism) led to the second industrial revolution. The third industrial revolution, also called "the digital revolution", started in the 1970s approximately, when advanced electronics and information technology developed further the automation of production processes[3].
The term "Industry 4.0" became publicly known in 2011; an association of representatives from business, politics, and academies supported the idea as an approach to strengthening the competitiveness of the German manufacturing industry [4]. Promoters of this idea expect Industry 4.0 to deliver "fundamental improvements to the industrial processes involved in manufacturing, engineering, material usage and supply chain and life cycle management"[5].
Enabled through the communication between people, machines, and resources, the fourth industrial revolution is characterized by a paradigm shift from centrally controlled to decentralized production processes. Smart products know their production history, their current and target state, and actively steer themselves through the production process by instructing machines to perform the required manufacturing tasks and ordering conveyors for transportation to the next production stage [6].

Fig.2.1 briefly schematizes industrial revolutions history.

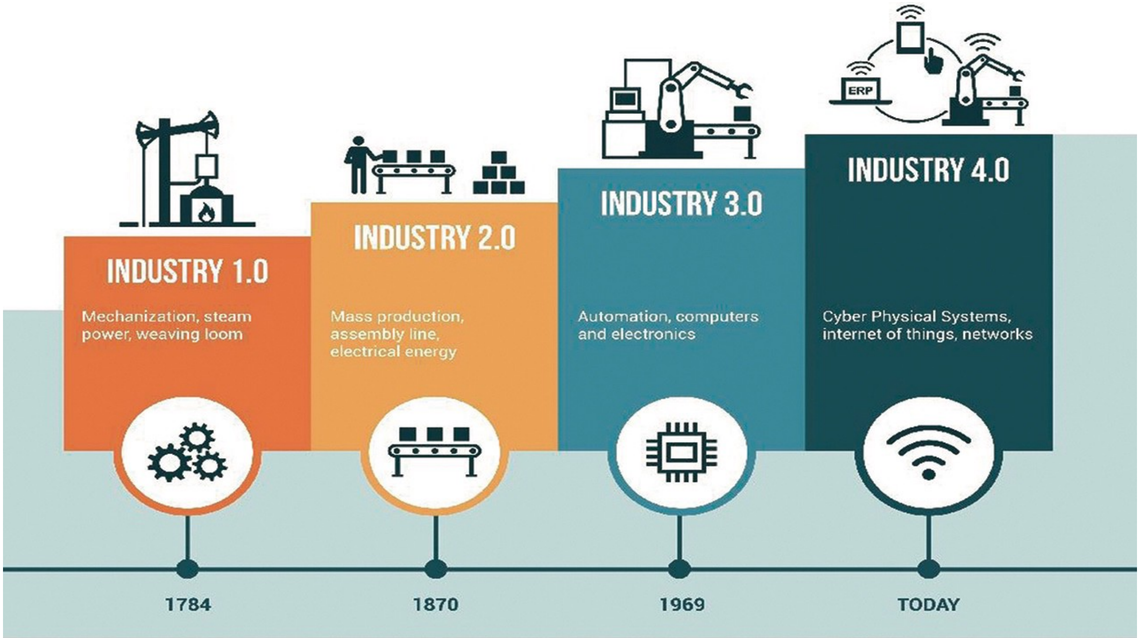


Figure 2.1. Industrial revolutions overview

In order to better describe what constitutes Industry 4.0,both the enabling technologies and the main benefits are described hereafter.

2.1.1 Enabling Technologies

Cyber-Physical System (CPS)

Cyber-Physical Systems (CPS) are integration of computation, networking, and physical processes. Embedded computers and networks monitor and control the physical processes, with feedback loops where physical processes affect computations and vice versa [7]. The human component is integrated into the system through specific advanced man-machine communication interfaces, which allow vocal, visual or haptic iterations.

Internet of Things (IoT)

The internet of Things (IoT) is about extending the power of the internet beyond computers and smartphones to a whole range of other things, processes, and environments. Those "connected" things are used to gather information, send information back, or both. When something is connected to the internet, that means that it can send information or receive information, or both. This ability to send and/or receive information makes things smart. To be smart, a thing doesn't need to have super storage or a supercomputer inside of it. Instead, it must only be connected to super storage or a supercomputer.
In the Internet of Things, all the things that are being connected to the internet can be put into three categories:

- Things that collect information and then send it;
- Things that receive information and then act on it;
- Things that do both.

And all three of these have enormous benefits that feed on each other.
IoT provides businesses and people better insight into and control over the 99 percent of objects and environments that remain beyond the reach of the internet. And by doing so, IoT allows businesses and people to be more connected to the world around them and to do more meaningful, higher-level work [8].

Industrial Internet of Things (IIoT)

IIoT is a subset of IoT and focuses specifically on industrial applications such as transportation, manufacturing, energy and agriculture. Notably, IIoT has different technical requirements given its increased level of complexity, interoperability and security needs. It is possible to implement an autonomous process of collection and exchange of information about the physical world, jointly with the usage of sensors, that makes several innovative services possible, especially if they are integrated with artificial intelligence and CPS functions.
Sensing and connectivity on industrial equipment mean that we are always able to monitor equipment conditions even remotely.

Cloud Computing

Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user. The term is generally used to describe data centers available to many users over the Internet. It allows companies to achieve necessary services and resources to enable Industry 4.0 applications, avoiding or minimizing up-front IT infrastructure costs. Also, it allows enterprises to get their applications up and running faster, with improved manageability and less maintenance, and that it enables IT teams to more rapidly adjust resources to meet fluctuating and unpredictable demand.

Big Data

With Big Data we mean the collection of data flow such that in terms of volume, speed, and variety, they are not manageable by a traditional relational database system. This data could be about activity, events, sensor values and machine state. The true advantage comes out during the data analysis, where we are looking for unknown correlations between monitored events and variables. The analysis is, therefore, the moment in which data are translated in information, which supports decision stages.
The presence and analysis of this big data require appropriate solutions to be implemented, so they typically rely on cloud services.

Machine Learning (ML)

Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. The main advantage in using a learning technique is that it is possible to solve problems that a traditional sequence of instructions could hardly manage, even with the contribution of domain experts.
In an Industry 4.0 context where there is a high amount of potentially useful data, machine learning became a crucial tool.

Usually, ML algorithms don't directly use raw data, but values derived from them instead. They are called *feature* and they are used for the first data elaboration step. Feature extraction is required to make model work easier, it can be done by reducing input variables number but preserving their information and characteristics, also to facilitate interpretability.

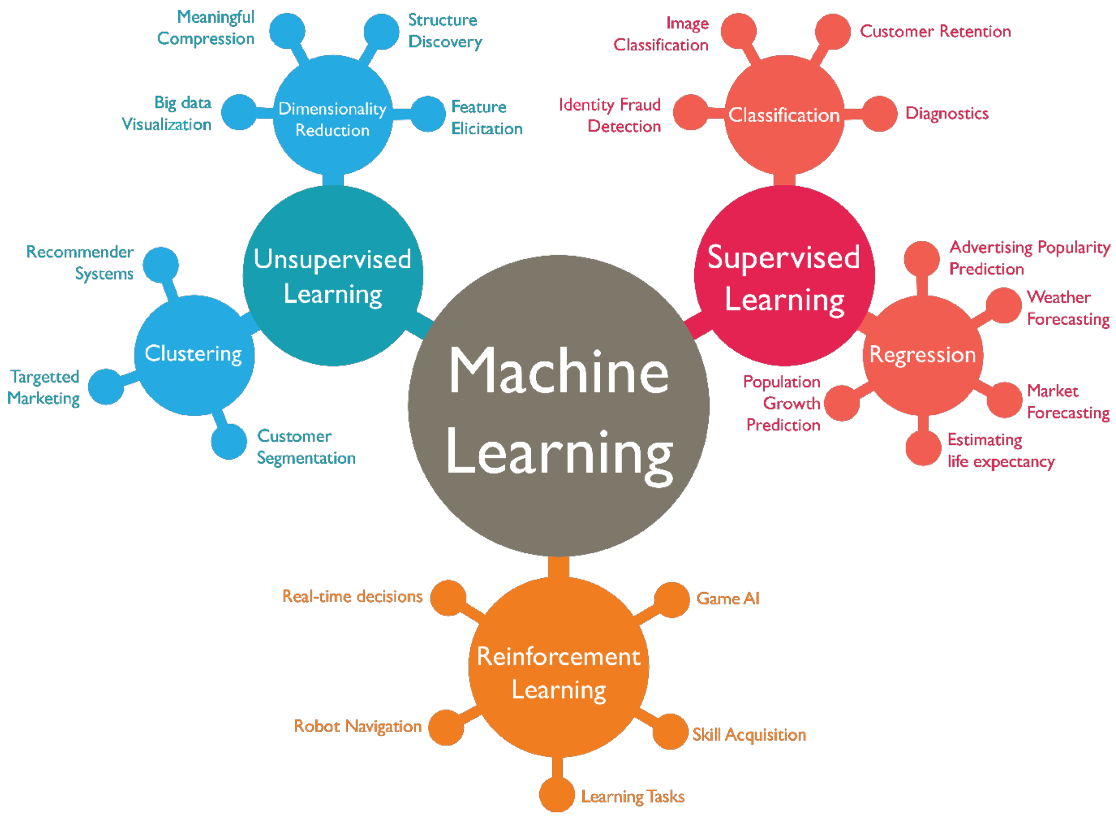


Figure 2.2. Most used Machine Learning approaches [9].

Machine learning approaches are divided in:

- *Supervised*: learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs[10]. The data is known as training data and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs[11]. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data;

- *Unsupervised*: learning algorithms take a set of data that contains only inputs, and looking for structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data;

- *Reinforcement*: learning algorithms based on rewards. After performing different actions, through different attempts and mistakes, the algorithm chooses which of them leads to more rewards. Obviously, its objective is to maximize the rewards, therefore it will be brought to choose actions which are taken for right, in order to reach the target as soon as possible;

- *Semi-supervised*: learning algorithms that fall between unsupervised (without any labeled training data) and supervised learning (with completely labeled training data). It is used either when we don't have enough labeled data or when acquiring this data is tricky or expensive.

2.1.2 Prospective and advantages

The application of illustrated technologies to the industrial field promises to lead the following benefits.

- *Integration* between the processes taking place during all the production chain; this allows digitization and optimization of activities that spread from internal logistic to the sale;

- *Interoperability* between systems or even between different companies, which allows the creation of a network that connects and simplifies the transfer of information between all partners who collaborate in the production of a certain asset;

- The ability to collect and store data about every machine and production aspect and the possibility it for operators and managers to access in every moment provide big support to *decision-making* activity. Analysis and artificial intelligence methods further simplify the task, providing additional information extracted from the data;

- Strong capacity of reconfiguration and modularity is achieved thanks to automation and digitization. These features simplify the necessary effort to satisfy individual customer needs. Rapid and optimized *individualized production* is made possible.

- Better collaboration between man and machine is made possible thanks to cyber-physical systems and innovative interaction interfaces, which make it easier and support the operators in the various tasks they must perform.

2.2 Maintenance

The Maintenance is defined as *combination of all technical, administrative and management actions, during the life cycle of an entity, intended to maintain or restore it to a state where it can perform the required function*[12].

The maintenance concept and process have undergone a strong evolution over the years, going from simple technique tasks performed individually when machines or instruments had blocking failures, to a complex system strongly integrated with other production processes and with an important strategic role.

As reported in [13], maintenance costs can currently represent from 15% up to 40% of total production costs. This is, therefore, strongly pushing companies to develop special strategies for maintenance management, finalized to reach the ideal object of *zero downtime*, namely succeed to keep the production lines always active, without having to stop them for the repair of malfunctions.

Among the most significant activities that allow moving in this direction there are failure systematic analysis and their reasons, careful management and planning of the warehouse for the immediate availability of spare parts, the use of CMMS (computerized maintenance management system) to support maintenance workflow and the usage of special maintenance policies.

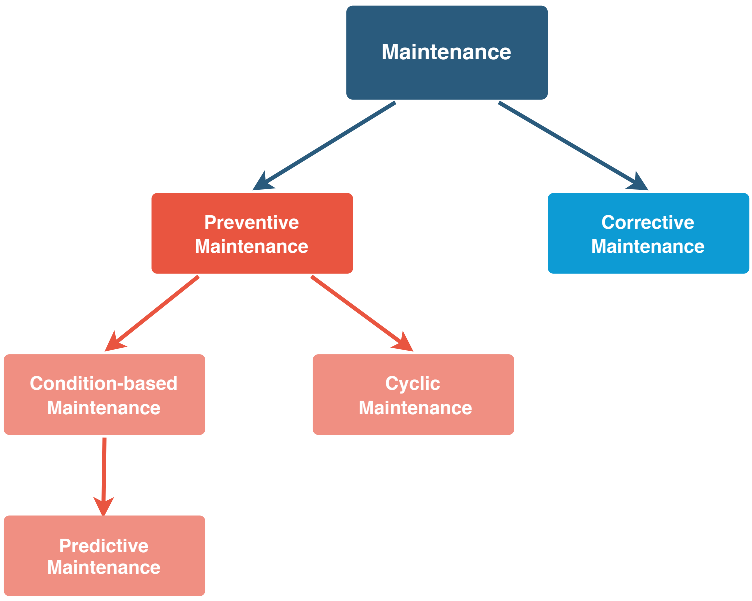


Figure 2.3. Maintenance approaches scheme

2.2.1 Corrective Maintenance

The corrective maintenance *is performed after the detection of a fault*[12], namely, an intervention is performed only when a machine failure occurs. For this reason it is also defined as *reactive maintenance*.
It is the simplest strategy to use and the one that involves the lowest costs to be implemented, since there are no expenses related to spare parts except when it is strictly necessary.
However, the above-mentioned costs are those related to the downtime of the machines, which are often unavoidable when failure occurs. Therefore, this leads to a halt of production and an overall drop in efficiency. Furthermore, the impossibility of scheduling interventions is added, since a malfunction can occur at any time of activity. Finally, the corrective approach requires that all the spare parts of the main components, or at least those critical for production, are always available in stock.

2.2.2 Preventive Maintenance

It includes all those maintenance activities *performed either at predetermined intervals or according to criteria, designed to reduce the probability of failure or degradation of the entity*[12].
This approach is driven by the significant money saving that a planned intervention allows, compared to the reactive case. Analysis of maintenance costs report that the cost is about 1/3 of the corrective intervention [14].
Inspections and replacements are therefore made based on time criteria, with the aim of intervening before the malfunction occurs. The time intervals are deduced from historical data, identifying the average duration of the useful life of all components subjected to degradation.
However, preventive maintenance is also subject to critical issues, due to the average life of the components being purely a statistical indicator, not providing any guarantee. Indeed, scenarios in which components are replaced in good condition and still functioning are possible, causing a waste of resources, and cases in which corrective maintenance occurs because components fail long before their expected duration.

2.2.3 Condition-based Maintenance

It is a specialization in preventive maintenance that *includes a combination of automatic or manual activities for monitoring conditions and their analysis*[12].
Therefore, this means that before acting effectively on the components, a survey of their state of health has been carried out and action is taken only if a certain

probability of risk is highlighted.
An anomaly condition is detected when some physical parameter of the machine is not in compliance with normal operation. Typical examples are the increase in noise, vibrations or temperature.
It can be implemented either by humans, for example by maintenance personal through special inspections or by expert operators who notice changes during use, or by means of special sensors, which are used to continuously monitor the parameters deemed significant.

2.2.4 Predictive Maintenance

Predictive maintenance is a further specialization of condition-based maintenance, *it is carried out following the identification and measurement of one or more parameters and extrapolation according to the appropriate models of the time remaining before the fault*[12].
According to this approach, the data relating to the functioning and conditions of the various components are recorded and saved in a history, therefore they can be used to build a trend of the overall behaviour. The information thus obtained is used to predict the evolution of the degradation level of a component and therefore plan a related maintenance activity.
The main advantage respect to condition-based maintenance lies precisely in the analysis of the trend and the construction of a model for the evolution of the state based on the experience gained from past analysis, which allows to estimate the residual useful lifetime of the component after detecting a deviation from normal operation when it is still in its first phase.
An effective predictive system allows to considerably improve and optimize the availability of the machinery and the time spent in production, reducing the number of maintenance interventions and their cost. It also has positive effects on product quality.

2.3 Maintenance in Industry 4.0 context

The maintenance is one of the most important fields of Industry 4.0.
As described in the previous section, condition- (and also predictive-)based maintenance approaches are more easily implemented through specific hardware and software systems, although possible also relying on manual inspections performed by maintenance technicians.

Analyzing the components and functionalities necessary to implement a **Condition-Based Maintenance (CBM)** system, the following elements are highlighted:

- *Sensors*, installed on the machines to perform measurements regarding the physical parameters of interest;

- *Communication*, in order to transmit the data collected to an aggregation center;

- *Storage*, to keep the history of the sensor values, and possibly integrate it with a list of events and activities;

- *Analysis*, to extrapolate correlations between variables and machine state, so consequently recognize and predict failure.

Therefore, there is a strong correspondence between the enabling services and technologies of Industry 4.0.

The main advantages and benefit coming from the application of these systems are reported below.

- Through the visualization of the data collected by the sensors in real time, it is possible to carry out *continuous monitoring activities* of the production machinery, even during normal operation and without having periodic inspections by maintenance workers;

- The storage of historical data supports the usage of statistical or machine learning models, in order to progressively increase the knowledge and capabilities of the system, until it is possible to identify the presence of anomalies from the initial stage and estimating a trend of deterioration, combining state and history;

- Since it is not possible to identify and prevent 100% of the blocking faults, when they occur a CBM system is useful for identifying the component that caused the stoppage and the type of problem that affected it, simplifying the task of the maintenance technicians and reducing the time needed to restore production activities.

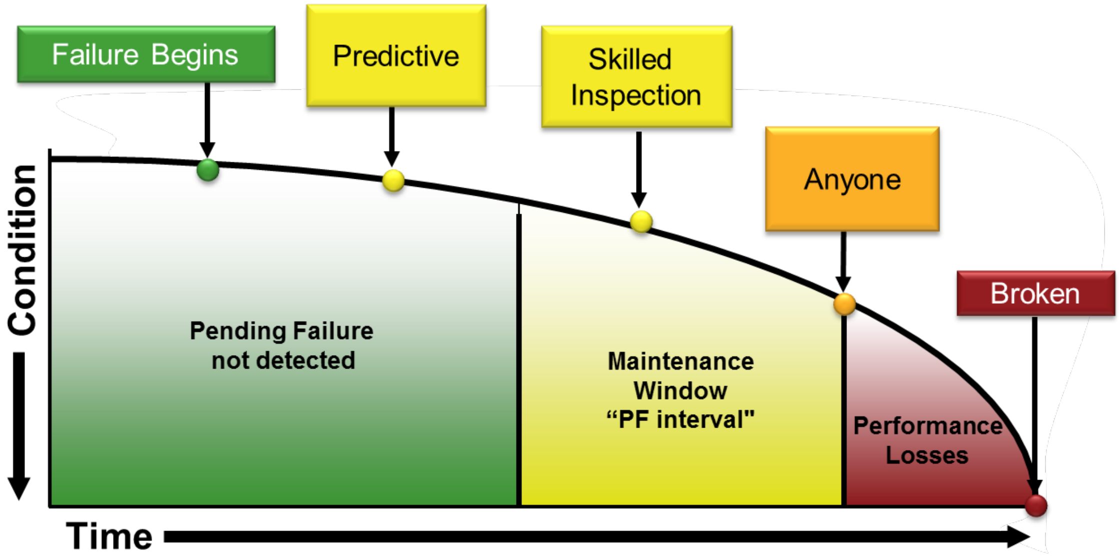


Figure 2.4. Predictive temporal scheme

In addition to these listed advantages (characteristic to the CBM system) there are also all the gains in terms of productivity, efficiency, planning and quality that result from the application of the predictive maintenance policies that these systems enable.

Chapter 3

Related Works

The contributions and results of research in recent years in the context of Condition-Based Maintenance (CBM) systems have been numerous.
As already reported in the previous chapter, they are mainly due to the strong interest of the industrial sector, which sees maintenance on condition and predictive as one of the most profitable applications of Industry 4.0.

Therefore, since the vastness of the topic in question, in this chapter, the main definitions, the different problems to be solved and the most relevant methodologies used in the solutions proposed in the literature are presented.

3.1 Problem Definition

Several problems fall under the CBM family, but they can be grouped together as they are aimed to achieve the same business objectives, namely, reducing costs related to machine maintenance and increasing the time spent in production.

Trying to summarize and generalize the structure of a CBM application, we have that it is composed of three phases [15], common to all the different specific implementations. The stages are:

- **Data Acquisition**: the process of gathering all the information that is considered relevant to be able to deduce the state of the machine or its components;

- **Data Processing**: the management and analysis of the data collected, in order to provide an interpretation and their transformation into knowledge about the machine;

- **Maintenance decision-making**: the definition of a decision policy regarding the maintenance actions to be performed which also depends on the additional information obtained through the processing step.

The main distinction within CBM applications is between diagnostics and prognostics.
The purpose of a **diagnostic** system is to detect and identify a fault when it occurs. In the ideal case, this therefore means monitoring a system, indicating when something is not working in an expected way, indicating which component is affected by the anomaly and specifying the type of anomaly.
On the other hand, the **prognostic** aims to determine whether a failure is close to occurring or to deduce its probability of occurrence. Obviously, since the prognostic is a prior analysis, it can provide a greater contribution as regards the reduction of the costs of the interventions, but it is more complex objects to be achieved.

Another option is to simultaneously use diagnostic and prognostic solutions applied to the same system. Their combination provides two valuable advantages:

- Diagnostics allows to intervene to support decisions in cases where the prognostic fails; this scenario is in fact inevitable, as there is failure that does not follow a predictable model, and also the failures that are foreseeable with good precision cannot be identified in all their events;

- The information obtained through diagnostic applications can be used as an additional input to the forecasting systems, thus allowing the creation of more sophisticated and precise models.

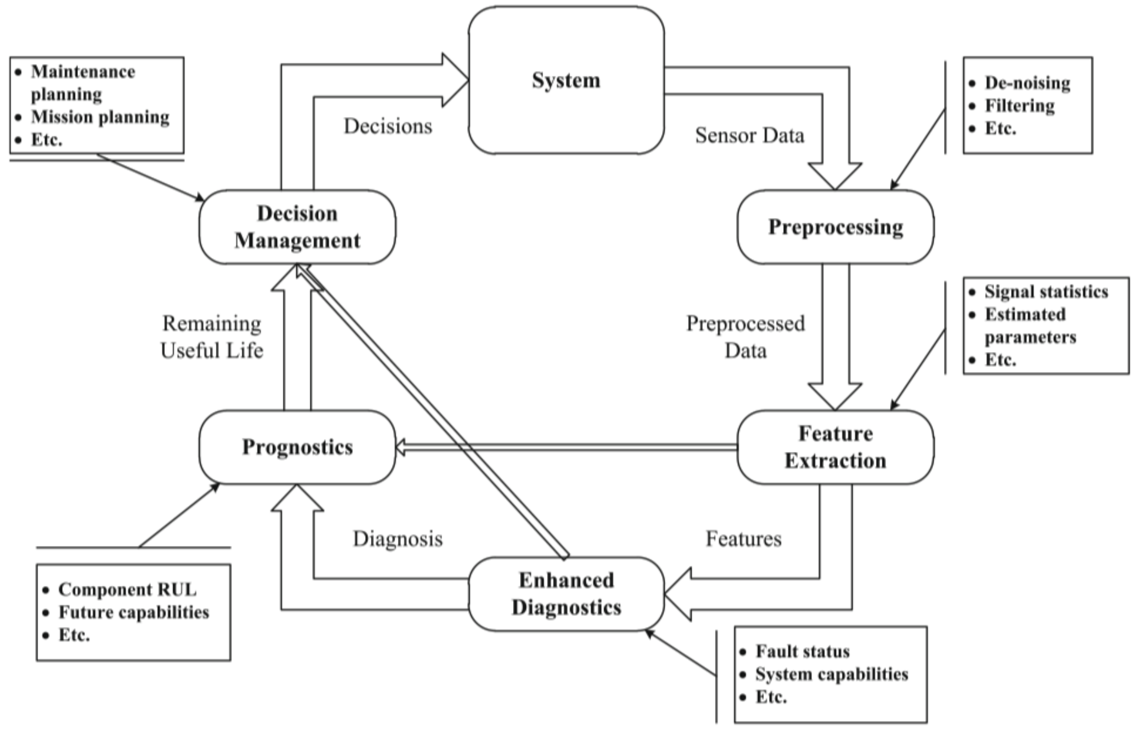


Figure 3.1. CBM system that combine diagnostic and prognostic [16]

3.2 Approach Methodologies

One of the most used categorizations in the literature regarding CBM systems is based on the used approach. It is a high-level distinction, as each of the reported classes is in turn composed of different specific models.
They differ from each other for characteristics such as the cost of application, the complexity, the generalizability, the expected accuracy and the type of input that they need to function.
It is considered useful to illustrate them, to provide a complete view of the paths that have been followed by research to solve diagnostic and prognostic problems.

3.2.1 Physical Models

This first possible approach is the physical models, which are based on the description of the actual process of degradation of the components of the machine under control. This means modeling in terms of the laws of physics on how operating conditions affect the efficiency and longevity of assets.

The most relevant variables include various thermal, mechanical, chemical and electrical quantities. Being able to represent how they impact the health of machinery is a very complicated task. Therefore the figure that deals with realizing this type of solution requires a high knowledge of domain and modeling skills.
Once the model has been created, it is necessary to have sensors available that allow obtaining values corresponding to the quantities considered relevant during the analysis and modeling phase, in order to use them as inputs.

The main advantage of this type of approach is that it is descriptive, therefore, it allows you to analyze the reasons for each output it provides, precisely because it is based on a physical description of the process. Consequently, it allows for validation and certification [16]. Regarding precision, it is strongly linked to the quality of analysis and modeling by domain experts.
The negative aspects, on the other hand, are the complexity and the high cost of construction, together with the high specificity for the system, which entails a limited possibility of reuse and extension.

3.2.2 Knowledge-Based Models

Even for the creation of knowledge-based models, domain experts are used, since what you want to model with this type of approach is directly the skills and behavior of the experts themselves.
The goal is to obtain a formalization of the knowledge they possess, in order to allow it to be reproduced and applied automatically.
Expert systems are in fact programs that use knowledge bases collected from people who are competent in a given field and then apply inference and reasoning mechanisms to them to emulate thinking and provide support and solutions to practical problems.

Among the most common approaches for implementing this type of model are rule-based mechanisms and fuzzy logic [17].
The former has the simplicity of construction and interpretability, but it may not be sufficient to express complicated conditions and may result in a combinatorial explosion when the number of rules is very high.
The use of fuzzy logic allows describing the state of the system through more vague and inaccurate inputs, making the process of model formalization and description simpler and more intuitive.
Even for expert systems, as for physical methods, the results are strongly guaranteed by the quality and level of detail obtained from the model and are highly specific.

3.2.3 Data-Driven Models

The data-driven models apply statistical or learning techniques to the collected data relating to the machines, intending to be able to recognize the status of the components. The idea is to be able to obtain the greatest amount of information regarding the state of the machinery in real-time, typically through sensors and from the production and maintenance activity logs, and to correlate them with the level of degradation of the individual components or with the entire performance of the system.
An analysis of the literature available on CBM shows that this type of approach is currently the most detailed by researchers and the most used in practical cases. The reasons are the following [16, 18]:

- Data-driven approaches, as the name itself suggests, require large amounts of data to be effective, but with the advent of Industry 4.0 and in particular of the Industrial Internet of Things this need is not difficult to satisfy;
- Compared to other approaches, they have the great advantage of not requiring in-depth knowledge specific to the application domain, thus making the contribution of experts on the final performance of the model less decisive; the contribution of the experts may still be useful to speed up the process of selecting the quantities to be used as input, but it has a much lesser weight if compared with the knowledge-based or physics-based methods; besides we have that learning and data mining techniques may be able to detect relationships between the input parameters and the state of the system that even to the experts themselves are not known in advance;
- Numerous tools are available[1] that implement machine learning algorithms that can be used for these CBM scenarios that require few configuration and optimization operations for the specific case.

The choice of a specific data-driven type model is highly dependent on the objective to be achieved by the system. In fact, based on the objective, the problem is modeled differently. The main options are shown below [19]:

Binary Classification

The simplest way is to represent CBM as a binary classification problem, in which every single input representing the state of the system must be labeled with one of

[1]Most cited and used Tools in literature are TensorFlow, Scikit-Learn, Keras, PyTorch, Theano and SciPy.

two possible mutually exclusive values.
In the event of a diagnostic problem, this means *deciding whether the machine is operating correctly or not correctly*, making all possible states fall into these two classes.
To apply the binary classification also to the prognostic case, the interpretation becomes that of *deciding whether the machine can fail within a fixed time interval.*
The difference between the two meanings is simply given by the different interpretation of the labels. This means that the same model can solve both problems. What will be differentiated is the labeling of the dataset used to carry out the training phase of the model.

Multiclass Classification

The multiclass version is a generalization of binary classification, in which the number of possible labels to choose from is increased. However, only one label must be associated with each input.
The diagnostic case extends the previous case in a very intuitive way, namely deciding whether the machine is working correctly or incorrectly, and in the second case in which of the possible states of the anomaly.
While in the prognostic applications you can find in which time interval before the failure the machine is located, where therefore the possible labels represent different intervals of proximity to failure.

Regression

Regression can be used to model prognostic problems. This means allowing to estimate the remaining useful life of a component in terms of a continuous number (provided by the regression model) of pre-fixed time units.
In this specific case, the training dataset must only contain data relating to components that have been subject to failure, in order to allow the labeling of the inputs backward starting from the instant of failure.

Anomaly Detection

Another possible representation of diagnostic problems is to consider it as an anomaly detection problem.
This means that the model must be able to establish whether the operation of the machine returns to a normal state or if it deviates from it, that is, coming in a case of anomaly.

The interpretation of the problem is therefore very similar to the binary classification.
However, this methodology differs from the classification which is part of the cases of semi-supervised learning (a difference of the previous cases, which are all supervised), because the model only needs to learn from input which represents correct and MUST functioning states, following the training phase, recognize anomalous states which are not known, or whose characteristics are unknown.

3.3 Data and Dataset

As it is easy to understand, data play a fundamental role in CBM applications, especially in the case of a data-driven approach. In this section we want to describe the relevant characteristics of the data that are present in systems of this type and also report the main datasets encountered in the literature, describing their composition.

3.3.1 Type and Sources of Data

The data can be divided into the following types [17, 19]:

Sensors Data

They are the measurements of all those physical quantities that describe in some way the state of the machine during its operation; they are obtained through special sensors that convert the physical value into an electrical value. Examples of these parameters used are noise, vibrations, pressure, temperature and humidity, where the relevance of each of them strongly depends on the system being monitored.
To be more specific, it is possible to distinguish sensor data, according to the type of values [15]:

- *Simple Values:* a single value, typically numerical, collected at a precise instant of time, such as temperature, pressure and humidity;
- *Signals:* namely the trend os a single quantity for a period of time, such as a sound wave or vibrational signal;
- *Multidimensional Values:* namely, a multiplicity of values collected at the same time referring to the same concept, such as a photograph or infrared thermography.

Statistic Data

Metadata, as also defined in [19], are the data that correlate the static operating conditions of the machine or plant at each instant of time, such as the type of piece produced, the code of the materials used, the speed of machine production, identification and characteristics of the operator who is using the machine.
The sources of this information can be the PLCs of the machines or the ERP systems of the production plant, or, if they are not available, the manual declarations of the operators, which must be digitized and integrated later.

Log Data

They are the historian of the events and relevant actions that concern a machine and its components. In particular, the lists of repair and replacement interventions or the history of the faults found are useful.
Also, in this case, they can be obtained thanks to ERP or CMMS systems, or by specific operator declarations.

3.4 Proposed Solutions

3.4.1 Architecture

Concerning the overall architecture of a CBM system in industry 4.0, most of the consulted literature uses a common approach, both as regards the components and their responsibilities, and for the interactions that take place between them.
An overall scheme is shown in Fig. 3.2, which generalizes the solutions proposed by the literature and provides a description of the roles of each component.

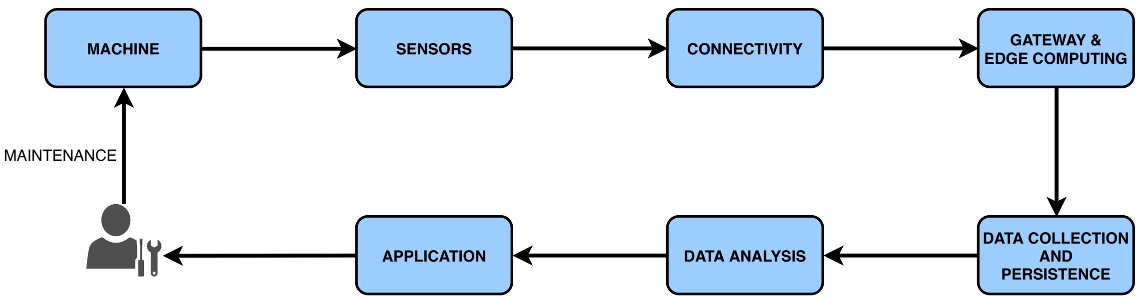


Figure 3.2. Architecture of generic CBM system

- *Sensors:* as already described in 3.3.1, these are the devices that deal with detecting the physical quantities of interest from the machine;
- *Connectivity:* devices that interface directly with the sensors to carry out the data collected by them and then transmit them through some communication technology, which can be via cable or wireless depending on the characteristics of the specific scenario;
- *Gateway and edge computing:* it is a first point of collection of raw data from multiple sensors; these data can be filtered or aggregated according to a well-defined logic, to reduce the traffic of data on the network and to detect and discard any anomalous or not-significant data as soon as possible;
- *Data collection and persistence:* it deals with the collection of information from the gateways and is the level that knows which data must be kept and which instead can be discarded; the data saved on the database will then be used later for analysis;
- *Data analysis:* it is the component that implements the statistical or learning model and therefore transforms the data into meaningful information;
- *Application:* namely, where the information derived from the previous component is presented to the end-user, possibly also intervening in the decision-making phase, suggesting corrective actions that the user can then carry out.

3.4.2 Classification Models

As highlighted in the previous paragraph, the biggest differences between the proposed solutions are in the realization of the actual model.
Among the three types of approach to the problem reported, we focus on the data-driven one since, although many of them are designed and tested in specific scenarios, the idea and the principle of operation are easily generalizable and can also be used in different contexts.
Classification algorithms are part of data mining and use supervised machine learning methods to make predictions about data. In particular, a set of data already divided ("labeled") into two or more classes of belonging is provided as input thanks to which a classification model is created, which will than be used on new ("unlabeled") data to assign them to the appropriate class.
The starting dataset is usually divided into two groups, namely the training dataset, which is used to create the model, and the test dataset, which has the purpose of testing the model. The validation of the model takes place by particular partitioning techniques, such as *Cross-Validation*. The latter works by dividing the dataset into

a certain number (k, chosen a priori) of groups: in rotation on all the groups, one of them will act as a test set and all the others as a training set.
Cross-Validation generally works well on many types of the dataset, but if necessary other methods perform the same function (such as fixed partitioning, used for very large dataset).
Many classification techniques have significant differences between them.
The best known are the following:

- *Decision Tree:* algorithms with a tree structure where each node represents a specific test on the data attributes and each branch is a "road" that is traveled based on the test result. The final nodes are the label with which each data can be associated. Its strengths are its interpretability, efficiency and good accuracy while the main weakness is its sensitivity to missing data;

- *Random Forest:* these are classifiers that combine the results of multiple decision trees for greater accuracy. Their weakness, however, concerns lower scalability compared to the size of the training sets;

- *Bayesian Classification:* it is based on the calculation of the probability that data belongs to a certain class. It is an accurate classifier with fair interpretability but the generation of the model is very slow in case of full-bodied datasets. To deal with this problem it is often necessary to introduce the hypothesis of statistical independence among the attributes of the dataset (called the Naive hypothesis, which however risks simplifying the model too much, reducing its accuracy);

- *K-Nearest Neighbors:* algorithm based on the calculation of the distance (the Euclidean one is often used) between the elements of the dataset. For example, data is assigned to a certain class if close enough to the other data of the same class. Parameter K represents the number of neighboring data taken into account when assigning classes. The K-NN risks becoming computationally expensive due to the calculation of the distances between the data, especially in cases where there are many attributes;

- *Neural Network[20]:* these are very accurate and robust techniques in case of missing data or outliers which, however, have poor interpretability and a slow learning process. Their functioning resembles the human brain: each node, which represents the neuron, receives the data, processes it and transmits the data and its analyzes to the subsequent nodes: in this way the nodes of the subsequent levels obtain more and more detailed information.

The outputs of the various classification algorithms can be evaluated by calculating some metrics that verify their quality, to understand if the created model is working well or needs some adjustments:

- The *accuracy* of the model, which is calculated as the ratio between the number of correctly classified data and the total number of data present in the test dataset;
- The *recall* and *precision*, calculated for each different class. The first is the ratio between the data correctly classified in a certain class and the total data belonging to the same class, while the second is the ratio between the data correctly classified in a certain class and the number of data assigned to that class.

Recall and precision must be calculated because accuracy alone is not enough to describe the model's output, especially in the case of unbalanced data sets in the distribution of classes.

3.5 Challenges and Known Problems

Although the possible approaches for the realization of a solution are numerous and different, there are some problems that are common and that all systems have to face in some way.
They are listed and described below.

- The data used, especially when dealing with values collected from sensors positioned on industrial machinery, usually contain a significant level of noise. Furthermore, they may also show variations due to different environmental conditions or other external factors. Therefore, the algorithms must be sufficiently robust to tolerate these oscillations not related to the health of the components;
- Learning models on a very high quantity and variety of data require very intense computational cost processing. This aspect is more delicate when you want to create systems for real-time diagnostics or prognostics, in which reactivity plays a fundamental role;
- The process performed by data-driven approaches to provide output is completely independent of the actual physical process. Except for a few algorithms (including decision trees, as indicated in the previous section), it is not possible to intuitively interpret why the model believes it is in a certain state;
- Uncertainty management must be taken into account. There are several sources of uncertainty: the first is introduced by noise in data and external conditions, the second is the recognition of the current state, and finally the predicting future states;

- As regards the prognostic, there is also an additional difficulty necessary for the fact that the same failure can occur as a result of different degradation paths. Furthermore, it often has a strong dependence on the components of the machine, and the failure of one can ignite the health of others. We must therefore also consider the case of occurrence of multiple simultaneous failures;

- Many of the approaches described in the related works assume that a dataset is available containing examples of sensor values classified by specific fault classes. In a real scenario, it is very difficult for it to be present, and if it is missing it means that defects should be specifically induced on the machine, and this is often very complicated to implement;

- Finally, there is a strong dependence on the human factor. They are the experts in the sector who must carry out the classification on the dataset that will be used. It is a very delicate phase, as errors during labeling can cause incorrect training and consequently the system risks being useless if not harmful.

3.6 State of Art

In this last section of the chapter some examples of real applications of predictive maintenance will be presented, with the aim of making it clear in which areas its use is already widespread today.

A typical use case of predictive maintenance is the detection of abnormal vibrations from an engine [21]. Moments before a fault occurs, an engine can emit small vibrations that become more and more evident with time: they are the first signs of some problem that is starting to manifest. If the engine is not repaired in time, it may then present other symptoms of malfunction such as wear, a decrease in performance or unusual noises and temperatures. The more time passes, the higher the probability of a more serious failure and the more difficult and expensive the intervention of a mechanic (see Fig.3.3). Immediate intervention, namely, as soon as the first anomalous vibrations are detected, can allow cost savings and ensure that after a small adjustment, the engine will continue to run in the usual way.

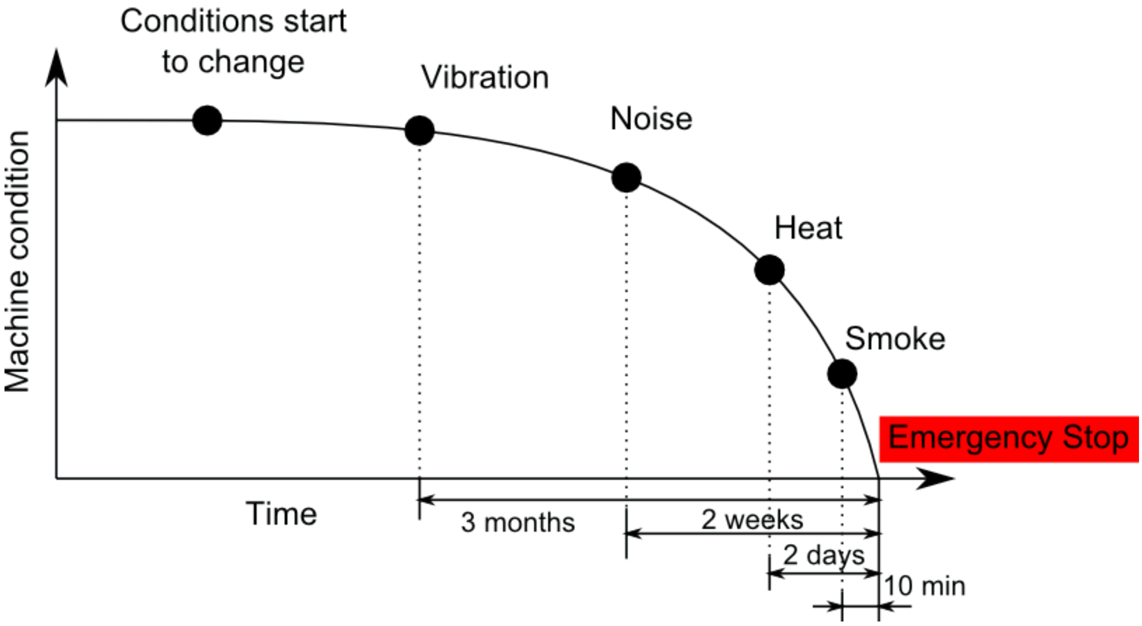


Figure 3.3. Machine condition relation to time and moments of indicators occurrence.

Through some specific sensors, however, it is possible to perceive the small initial vibrations and allow targeted intervention quite before the problem becomes serious.
In fact, these sensors transmit the data collected on vibrations to the algorithm which is adequately interpreted by an adequate staff.

This kind of approach is being considered, in addition to industrial production, also in other sectors: for example, many airlines are investing their capital in this technology in order to be able to apply different maintenance techniques on their fleets in order to avoid aircraft breakdowns and achieve significant cost savings. A Honeywell survey [22] said that among airlines, about 69% of them will increase the budget for these operations and that, for example, EasyJet has implemented and is already using over 50 predictive algorithms on its planes. The low-cost airline, in fact, began to carry out predictive maintenance operations already during the flight of the plane, monitoring the main signals emitted by the aircraft. In this way, the data is sent to the ground and analyzed by technicians and engineers, who in case of expected failure and malfunction, can report problems and schedule the replacement of defective parts even before the plane lands [23]. Thanks to this strategy it is possible for EasyJet to save time (scheduling component replacements in advance can avoid delays in subsequent flights) and, equally important, to monitor the planes during their effective operation. In fact, if the maintenance were carried out only on the ground, it would be impossible to find all the problems due to the completely different conditions: temperature, pressure and vibrations analyzed during the flight phase have a significant influence on the monitoring of the aircraft condition.

Another case of application of predictive maintenance is the monitoring of the state of health of wind turbine gearboxes. For this purpose, a special project has been launched, called SIMAP [24] (abbreviation which stands for "intelligent system for predictive maintenance") aimed at designing a dynamic maintenance calendar, optimized for the needs and operating life of wind turbines. In addition to identifying anomalies in operation, it also allows advance planning of repairs. There are several parameters that can be studied and recorded during the operation of a wind turbine. Also in this case, very often it is necessary to pay attention to the vibrations emitted, paying attention to the fact that they often also depend on the rotation speed of the turbines themselves: at low speeds, the sensors may not pick up the signals of a possible failure. In addition to the vibrations, the acoustic emission is also monitored, which can be very useful for identifying problems still in the initial phase, even at low rotation speeds, the temperature and the current and power signals during the operating phase.

Finally, another example of predictive maintenance application concerns railway networks [25]. In particular, systems have been developed to detect important information through sensors placed on the network, such as data on temperature, deformation of mechanical parts, impact and weight of various components. The ultimate aim is to avoid service interruptions as much as possible and therefore increase the average speed on the route of the railway network to meet the growing demand for services and load on the network itself. To develop such a system, thousands of historical data have been used regarding past faults, the types of trains that run the network and even weather data: by crossing all this information, the generated model is able to warn in case of failure risks and plan a maintenance intervention.

Chapter 4

Architecture

This chapter describes the various parts, both hardware and software, that make up the system and the interactions between them.
The general structure is based on what appears to be the most used in literature, shown in Fig. 3.2, where storage, analysis and application responsibility are condensed into a single node.
The diagram representing the system architecture and the description of the individual components are shown below.

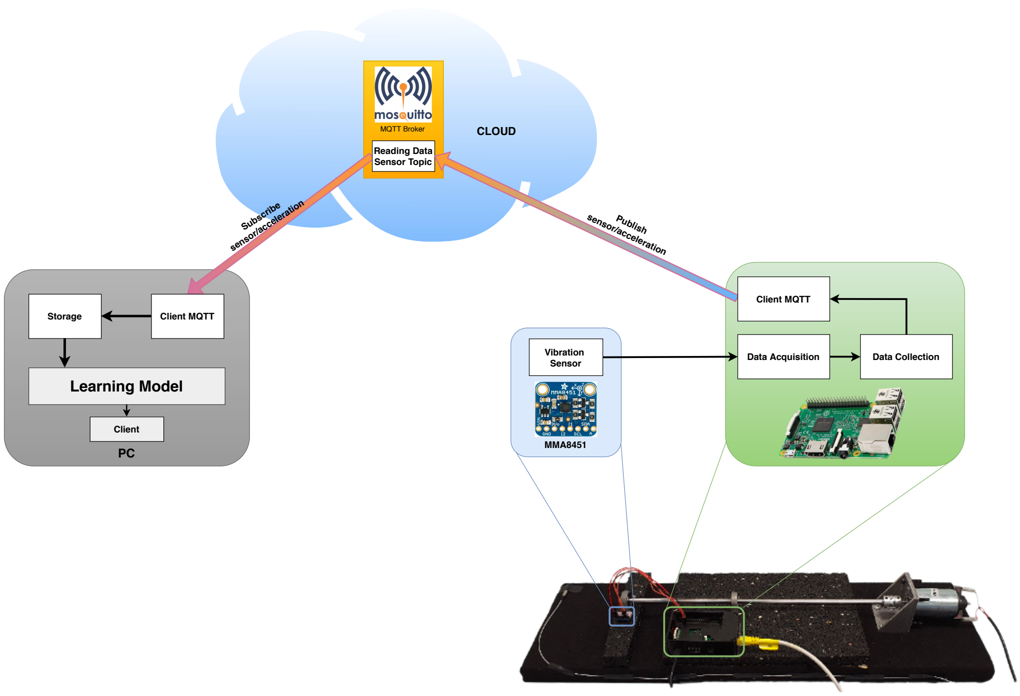


Figure 4.1. Proposed System Architecture.

4.1 Machine

The machine in which the anomaly detection system is applied was designed to measure the vibrations of a bearing. Through appropriate modifications, two different operating conditions have been created, making possible to apply certain algorithms to evaluate its status.

During the analysis phase of the project, the requirements that the machine should have satisfied were identified, which are the follows:

- The possibility of applying sensors on it to detect significant parameters on its state, without interfering with the normal operation;
- The full-time availability of the machine to be used for exploratory testing and analysis;
- The possibility of intervening on it in a simple way to bring the state into one or more situations of anomalies and from them to return to the optimal operating state;
- The ease of ignition and use, so as not to require specific skills to train and test the CBM system;
- The low cost and easy availability of its components, so as to operate on it with greater freedom;
- Similarity or correspondence with industrial machinery, in function and components.

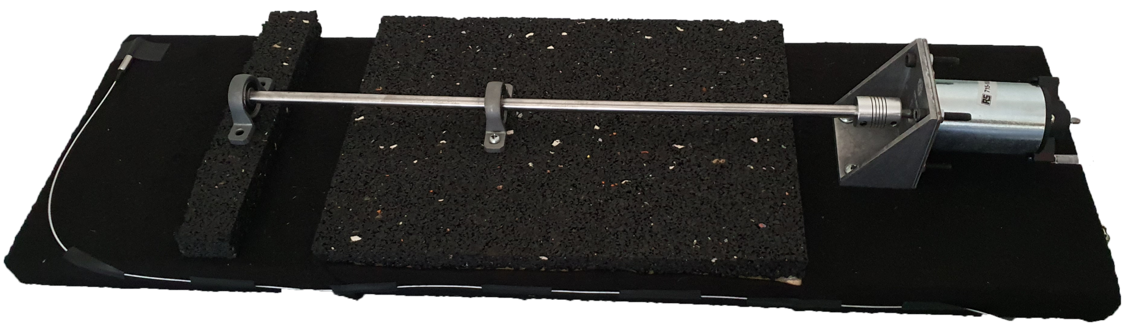


Figure 4.2. Machine used for the project realization.

The machine was built and made available by Reply, in order to create a model for studying the vibrations on the bearings.

It is driven by a Crouzet Motor, that is a 16W brushed DC motor.
Through a rotating shaft, $75Ncm$ of torque are transmitted to the bearings, reaching an output speed of $3370rpm$.
The whole is run with a $12V$ and $1.5A$ power supply.

4.1.1 Fault

The aim of the project is to create a model capable of recognizing fault states when they occur on the machine.
Therefore two possible states of the machine have been created by means of the clamps that secure the bearings. In fact, while the central bearing is fixed to the structure and has the task of stabilizing the rotating shaft, the outermost bearing has the possibility of loosening the fastening that fix it to the structure, simulating its malfunction.

4.2 Sensor

In order to determine the operating status of the rotating machines, the best way is to analyze the vibrations, as highlighted in the literature [26].
The vibration analysis of electrical rotating machines lies on the fact that all rotating machines in good condition have fairly stable vibration pattern. Under any abnormal condition in working of machines, the vibration pattern gets changed.
Based on the type of defect and its slope of progression, predictive maintenance schedule can be proposed. As a general rule, machines do not break down or fail without some form of warning, which is indicated by an increased vibration level.
The vibrations caused by the defects occur at specific vibration frequencies. The vibration amplitudes at particular frequencies are indicative of severity of the defects.
The vibration analysis technique is the mainstay of the predictive maintenance technique. Vibration analysis is most effective techniques for monitoring the health of machinery.

Accelerometer
Among the characteristics of the sensors used in the consulted articles, it was noted that the sampling frequency of the acceleration values varies from $20kHz$ up to $40kHz$. Hardware capable of ensuring this level of performance requires very high costs, outside the budget foreseen for the project.
We opted for the MMA845, a low consumption MEMS[2] triaxial accelerometer with an I^2C interface, which has been specially programmed, to obtain the maximum

[2]Micro Electro-Mechanical Systems

sampling frequency of $2.2kHz$, sufficient to analyze the vibrations of our machine since its moderate rotation speed.

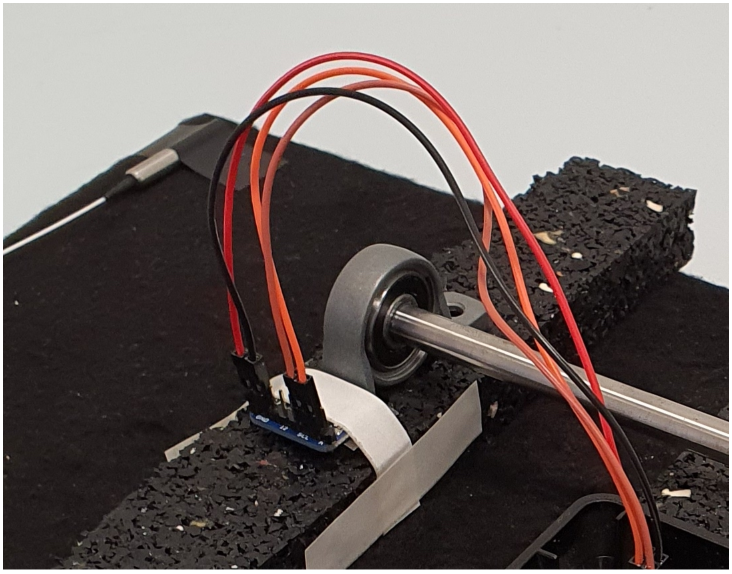


Figure 4.3. Accelerometer positioning on bearing.

As shown in Fig.4.3 the accelerometer has been fixed on the bearing support, in order to analyze its vibrations by direct contact.

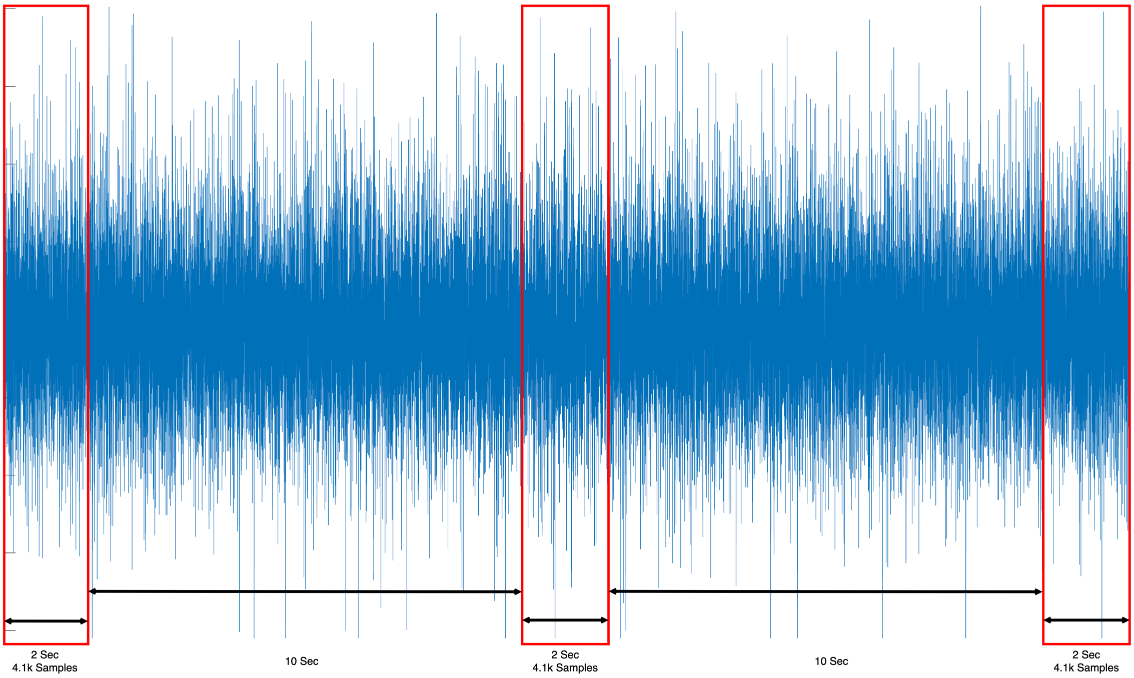


Figure 4.4. Scheme of the acceleration sampling process.

The sampling technique partially mirrors that used in the NASA Bearing [27], namely, it samples 2 seconds of vibrations at a frequency of $2.2kHz$ every 10 seconds. After each sampling interval, the Raspberry Pi convert and send the data through MQTT protocol.

4.3 Data Collection

Data collection function is performed by a Raspberry Pi, which acquire the raw data, processes it by creating a JSON type file that will be saved locally and send via the MQTT protocol to a online Broker.
The data acquisition algorithm was created following the structure suggested by the NASA and FEMTO Bearing datasets [27], namely, a file is created for each data sampling interval. In the specific case, each file that is created contains approximately 4000 acceleration values for each of the x, y and z axes, which represent the entire vibration signal corresponding to the two second sampling window.

```
{
    "signals": {
        "x": [ ARRAY (FLOAT)],
        "y": [ ARRAY (FLOAT)],
        "z": [ ARRAY (FLOAT)]
    }
}
```

Listing 4.1. Data file structure for a single range

Above it is a JSON (JavaScript Object Notation) file example. It is very simple to read and interpret both for people and for the main programming languages. JSON is a data transmission format actually independent of these languages, which however uses some conventions typical of programming, in the sense that it groups its data as sets of key - value pairs (similar to Python dictionaries or maps of Java, for example) and as ordered lists, very similar to the classic arrays. The syntax of the JSON files therefore follows a fixed scheme where each key (which represents a specific field, or attribute) corresponds to an element (which can be a string, a number, a list or a Boolean value that contains the information of interest). The data relating to the project of this thesis were therefore saved as JSON because in addition to the ease of use, this format allows to save information that is not completely structured (for example, the lists do not necessarily have to be all the same length) and can be read from non-relational databases.
Therefore, the elements contained in the file provided have the following structure; the names of the attributes are written in quotation marks and the type of data they contain is indicated for each one.

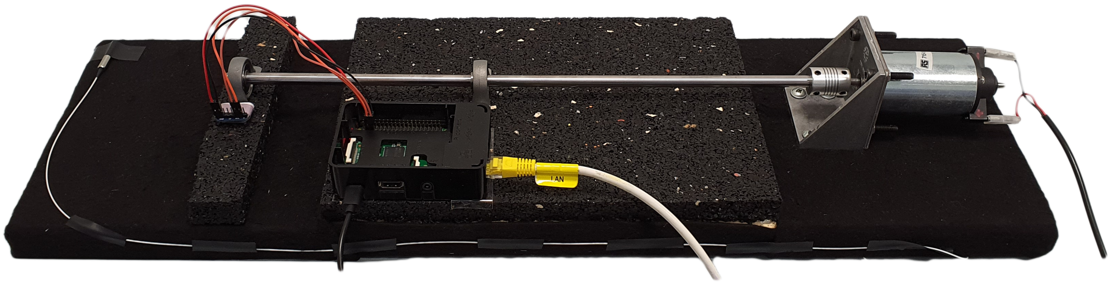


Figure 4.5. Acquisition system.

4.4 Transmission Protocol

For data transmission, the MQTT protocol [28] is used.
MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium. For example, it has been used in sensors communicating to a broker via satellite link, over occasional dial-up connections with healthcare providers, and in a range of home automation and small device scenarios. It is also ideal for mobile applications because of its small size, low power usage, minimised data packets, and efficient distribution of information to one or many receivers.
In our scenario, after a vibration sampling, a script that runs inside the Raspberry Pi converts the raw data into JSON and registering on a "topic" sends them to an MQTT broker which will forward the message to all those clients inscribed on the same topic, in our case the client will be another python script that runs on the PC which, being inscribed on the "topic", will receive every file produced by the Raspberry Pi and save it in a specific folder that will be then used to carry out the necessary analyzes using machine learning algorithms. The Fig.4.1 shows the data flow.

4.5 Used Tools

The tools used for the project related to this thesis are essentially some software libraries suitable for the management and manipulation of big data.
Therefore, the real fulcrum of the data analytics project was carried out with Python (and through Pycharm, an integrated development environment used mainly for this language) because it has several optimized libraries for data analysis that have proven to be fundamental not only for the formulation of prediction models but also precisely for the simple management of data thanks to adequate internal structures. The main libraries used are shown in the following list, together with a brief description of the operations that have allowed them to be carried out:

- Json [29]: it is a library that allows the encoding and decoding of a JSON type file in a Python dictionary and vice versa;

- Numpy [30]: it is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including: a powerful N-dimensional array object, sophisticated (broadcasting) functions and useful linear algebra, Fourier transform, and random number capabilities.

- Pandas [31]: it is a library that allows the management of data in tabular form (through the so-called DataFrames, data structures very similar to the tables of a classic database, which allow to index the data and manipulate them effectively) or sequential form (through one-dimensional indexable vectors called Series, less used in this thesis compared to DataFrames). Among its most important features, there is also the ability to perform numerical, statistical operations and display of results in a very quick and intuitive way. In addition to these reasons, Pandas was then used for its methods of reading and writing external files in different formats, such as CSV;

- Matplotlib [32] and Seaborn [31]: these are useful libraries for creating graphics. During this work, extensive use was made of these libraries, as the graphic display of the results obtained was a fundamental component of the project. It proved to be fundamental during the exploration of the dataset to direct the analyzes towards a certain direction, and also during the tests of the prediction models to deny or confirm hypotheses and choices made. Both Matplotlib and Seaborn, in a few lines of code, allow to view 2D or 3D graphics;

- Scikit-learn [33] and Keras [34]: they are data analysis and machine learning oriented libraries, specially designed to be used together with other libraries such as Numpy (another Python library that supports large vectors and multidimensional matrices and adds different mathematical functions) or Pandas (many features of this libraries are in fact designed to receive a DataFrame as input, for example). Scikit-learn contains various classes and methods to carry out any type of analysis, thanks to the possibility of implementing all the most important data mining algorithms. As part of this thesis, it was mainly used for the preprocessing phase and for the construction, train and test of the various classifiers designed. Keras instead is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. It supports both convolutional networks and recurrent networks, as well as combinations of the two end runs seamlessly on CPU and GPU.

Chapter 5

Models Realization

This chapter of the paper describes how the data collected by the sensors are used for the detection of anomalies on the machine and therefore how the functions exposed by the system described in the previous chapter are actually implemented. Therefore all the steps carried out in the project and all the analysis carried out to reach the final purpose will be explained in detail.

For each step, the methodology used and the reasons for any choices made, as well as the results obtained, even if partial or intermediate, will be explained.

There has been a lot of focus on the preprocessing of raw data, in the specific case of vibrations of rotating mechanical parts and in the evaluation and selection of features, because it is considered the most important and most impact part in the creation of a machine learning model, as it is greater the quality of the features as easier and faster it will be from the point of view of the algorithm to train and get better results. For this reason, two parallel paths have been taken, in order to show advantages and disadvantages in the two cases examined which will lead to the choice of a definitive architecture considering requirements in a project from which this thesis is born.

In Fig.5.1, the yellow blocks represents the common part of all the analyzed algorithms, that is the data collection and the creation of a Data Set. The blue ones represent the two parallels path: on the left the creation of FFT while on the right the creation of DWT, for both it continues with the creation of the features and finally with the selection of the same. Lastly, the green blocks instead represent the steps for creating the general predictive models, they are used both after the creation of the features and after the selection of the features to looking for a minimum amount of data to make correct predictions.

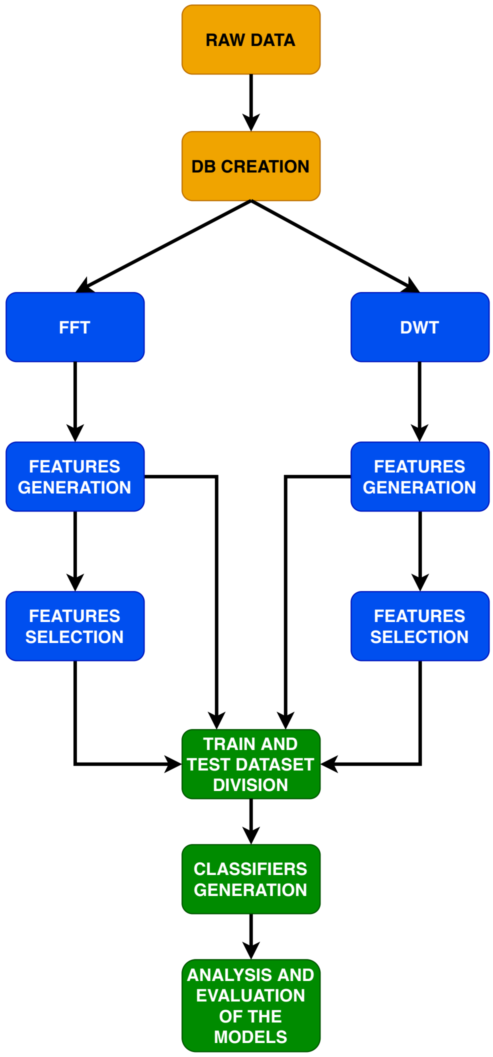


Figure 5.1. Fundamental Blocks of the Project.

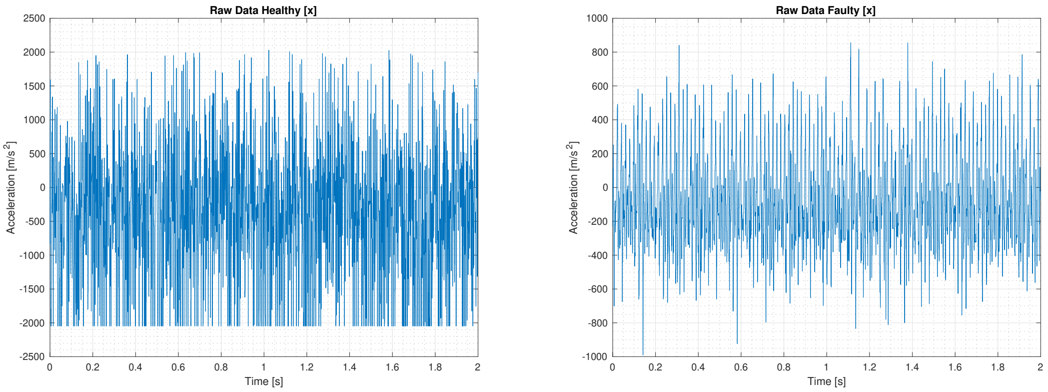
5.1 Data Preprocessing

Preprocessing refers to the transformations applied to our data before feeding it to the algorithm. Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not suitable for the analysis. For achieving better results from the applied model in Machine Learning projects the format of the data has to be transformed in a proper manner. Some specified Machine Learning model needs information in a specified format, for example, Random Forest algorithm does not support null values, therefore to execute random forest algorithm null values have to be managed from the original raw data set. Another aspect is that data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in one data set, and best out of them is chosen.

In the project specific scenario, each input element coming from the machine to be observed through the models is composed of a vibration signal of a two-second interval, calculated from approximately 4000 acceleration values on each axis. Since features from the frequency domain make the algorithm more accurate than the features from the time domain when evaluating the condition of the bearing [35], the next step is to convert the raw data from time domain to frequency domain. To operate this domain transformation two possible ways are proposed.

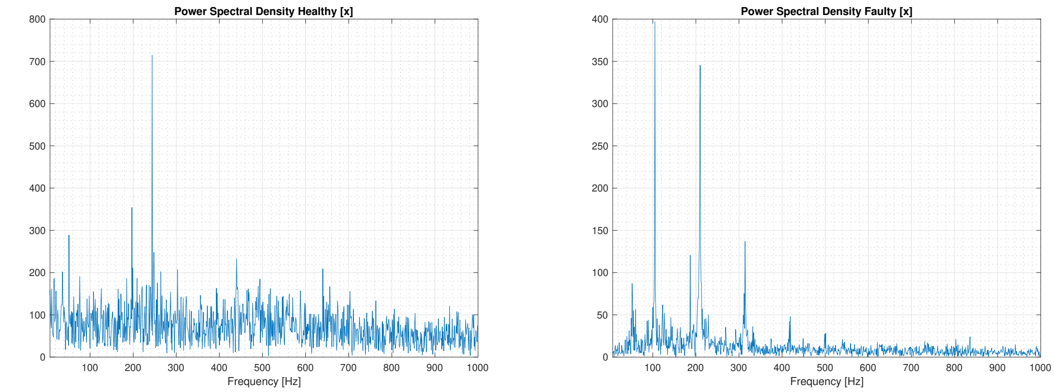
5.1.1 Fast Fourier Transform (FFT)

Fourier analysis is a field of study used to analyze the periodicity in (periodic) signals. If a signal contains components which are periodic in nature, Fourier analysis can be used to decompose this signal in its periodic components, telling us the frequency of these periodical component.
Two (or more) different signals (with different frequencies, amplitudes, etc) can be mixed together to form a new composite signal. The new signal then consists of all of its component signals.
The reverse is also true, every signal no matter how complex it looks can be decomposed into a sum of its simpler signals. These simpler signals are trigonometric functions (sine and cosine waves). This was discovered (in 1822) by Joseph Fourier and it is what Fourier analysis is about. The mathematical function which transform a signal from the time-domain to the frequency-domain is called the Fourier Transform, and the function which does the opposite is called the Inverse Fourier Transform.
The Fast Fourier Transform (FFT) is an efficient algorithm for calculating the Discrete Fourier Transform (DFT) and is the de facto standard to calculate a Fourier Transform. It is present in almost any scientific computing libraries and packages, in every programming language.
Nowadays the Fourier transform is an indispensable mathematical tool used in almost every aspect of our daily lives.
In our specific case, we can see that the distinction between health and faulty data is more marked and easy to understand by going to analyze the FFT in Fig.5.3 rather than the Raw data in Fig.5.2.



(a) Healthy Raw Data (b) Faulty Raw Data

Figure 5.2. Time domain data representation.



(a) Healthy FFT (b) Faulty FFT

Figure 5.3. Frequency domain data representation.

This first manipulation of raw data brings a considerable advantage from the point of view of machine learning algorithms, making their training easier.
It is not possible to directly use the coefficients coming from the FFT calculation since there would be about 4000 features, thus making the training of the machine learning algorithms inefficient and unnecessarily expensive from the computational point of view.

5.1.2 Discrete Wavelet Transform (DWT)

The general rule is that this approach of using the Fourier Transform will work very well when the frequency spectrum is stationary. That is, the frequencies present in the signal are not time-dependent; if a signal contains a frequency of xHz this frequency should be present equally anywhere in the signal, making impossible to know the precise instant when particular event took place in FFT based approach. The more non-stationary/dynamic a signal is, the worse the results will be. This is not good, since most of the signals we see in real life are non-stationary in nature.
A much better approach for analyzing dynamic signals is to use the Wavelet Transform instead of the Fourier Transform.

How Wavelet Transform works

The Fourier Transform uses a series of sine-waves with different frequencies to analyze a signal. Therefore, a signal is represented through a linear combination of sine-waves.
The Wavelet Transform uses a series of functions called wavelets, each with a different scale. The word wavelet means a small wave, and this is exactly what a wavelet is.

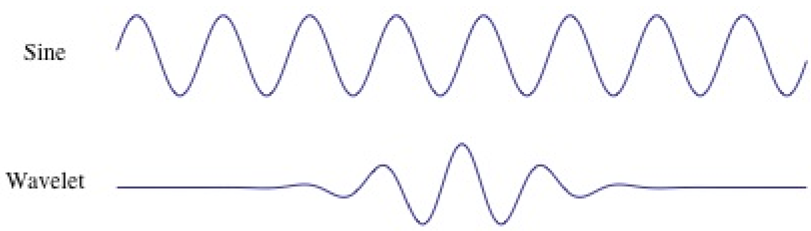
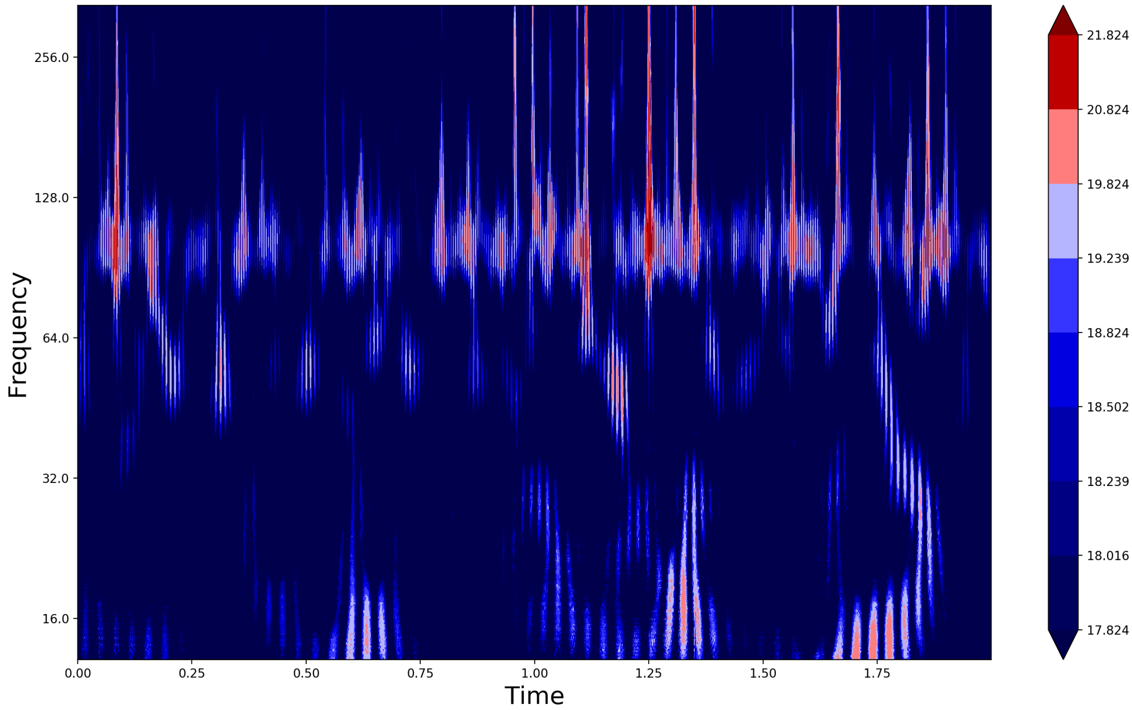


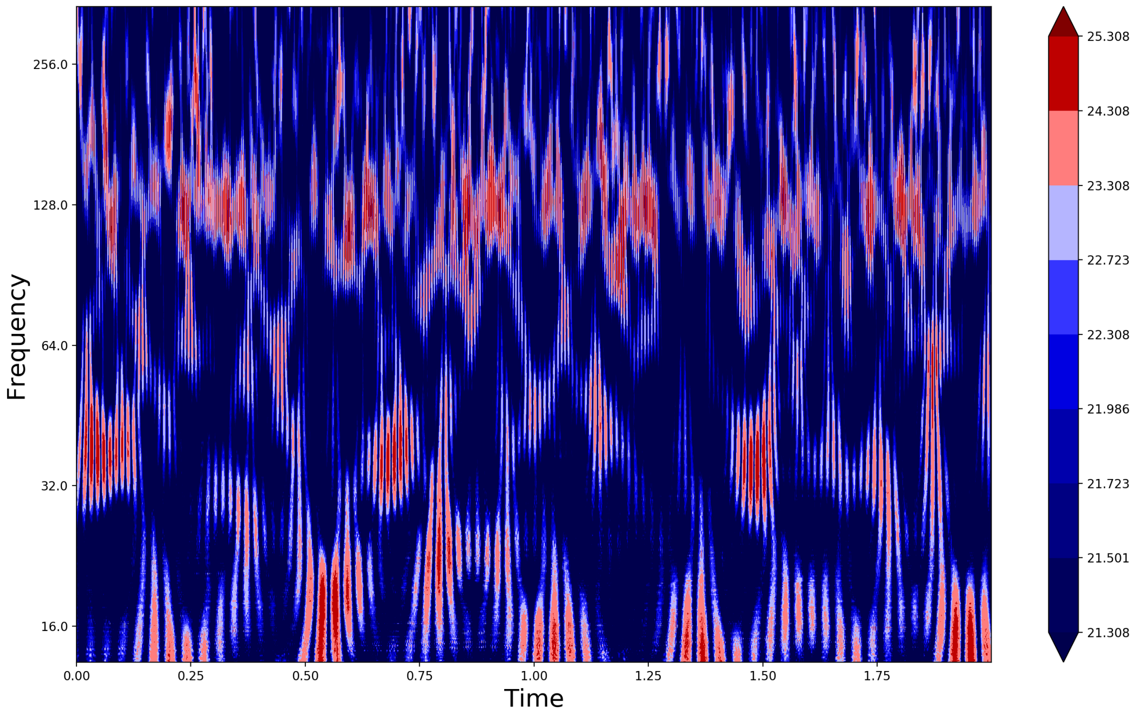
Figure 5.4. The difference between a sine-wave and a Wavelet.

In Fig.5.4 we can see the difference between a sine-wave and a wavelet. The main difference is that the sine-wave is not localized in time (it stretches out from $-\infty$ to $+\infty$) while a wavelet is localized in time. This allows the wavelet transform to obtain time-information in addition to frequency information.

Since the Wavelet is localized in time, we can multiply our signal with the wavelet at different locations in time. We start with the beginning of our signal and slowly move the wavelet towards the end of the signal. This procedure is also known as a convolution. After we have done this for the original (mother) wavelet, we can scale it such that it becomes larger and repeat the process [36].

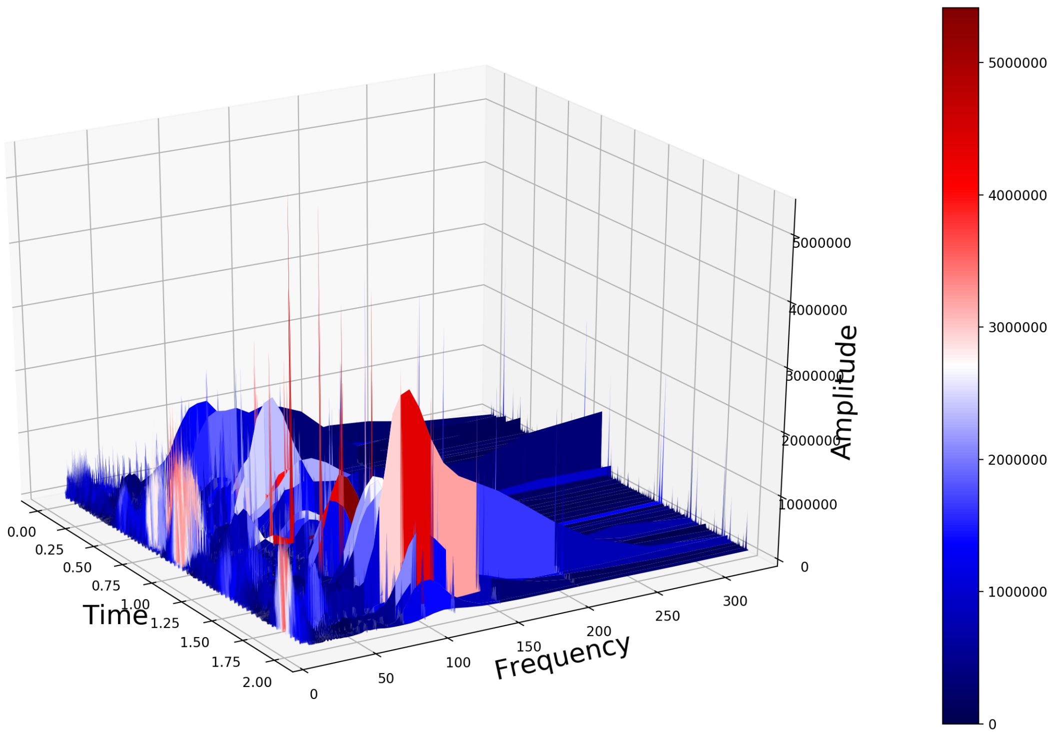


(a) Healthy DWT

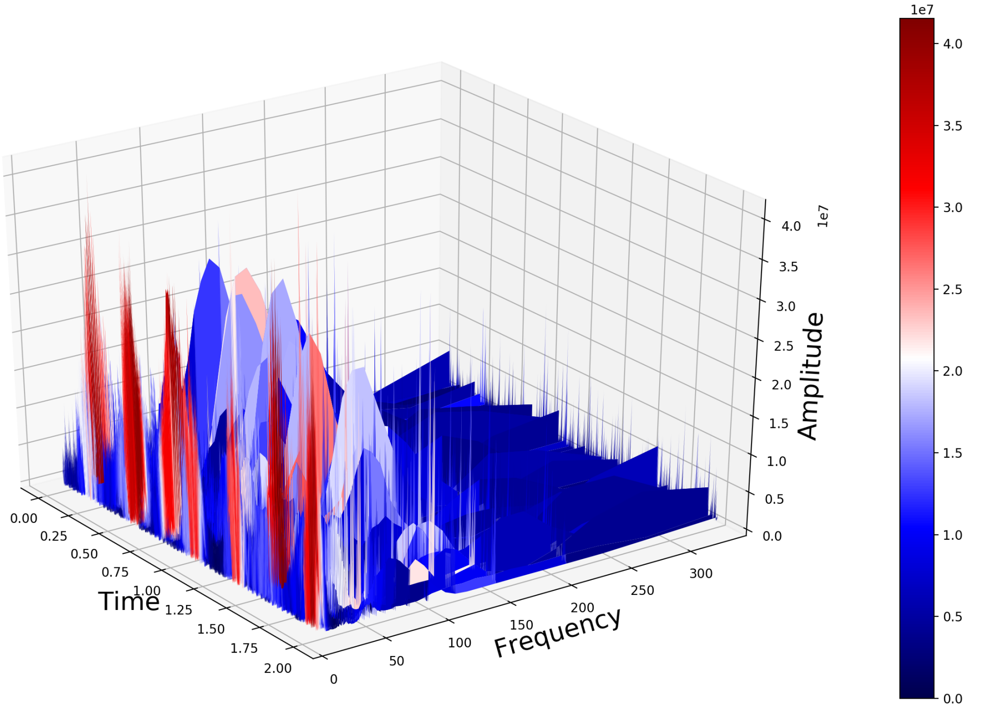


(b) Faulty DWT

Figure 5.5. Scaleogram DWT representation.



(a) Healthy DWT



(b) Faulty DWT

Figure 5.6. 3D DWT representation.

As we can see in Fig.5.5, the Wavelet transform of an 1-dimensional signal will have two dimensions. This 2-dimensional output of the Wavelet transform is the time-scale representation of the signal in the scaleogram form.
In Fig.5.6 the scaleogram is plotted in a 3D version.

Mathematically, a Continuous Wavelet Transform is described by the following equation:

$$X_w(a,b) = \frac{1}{|a|^{1/2}} \int_{-\infty}^{+\infty} \mathrm{x(t)} \bar{\psi}(\tfrac{\mathrm{t-b}}{\mathrm{a}})\, dt$$

Where $\psi(t)$ is the continuous mother wavelet which gets scaled by a factor of **a** and translated by a factor of **b**.
When we are talking about the Discrete Wavelet Transform, the main difference is that the DWT uses discrete values for the scale and translation factor.
The DWT is only discrete in the scale and translation domain, not in the time-domain.

5.1.3 Features Extraction

After the first manipulation of the raw data done in 5.1.1 and in 5.1.1, in order to reduce the input dimensionality and making known useful characteristics more evident, the next step is to extract the features.
The features that we have chosen to use are taken from the literature and are among the most used in both vibrational analysis and signal analysis in general.
They are generated in parallel, starting from the coefficients of the FFT and DWT.
In particular, the extracted features are:

- Entropy: it can be taken as a measure of complexity of the signal;

$$\sum_{i=0}^{n-1} P(x_i) log_2 P(x_i)$$

- Variance: it measures how far a set of (random) numbers are spread out from their average value;
- Standard Deviation: it is a measure of the amount of variation or dispersion of a set of values;
- Mean: it is the central value of a discrete set of numbers;
- Median: it is the value separating the higher half from the lower half of a data sample;

- 25% of value;
- 75% of value;
- Root Mean Square: it is the square of the average of the squared amplitude values;

$$\sqrt{\frac{1}{n}\sum_{i=0}^{n-1} x_i^2}$$

- Mean of Derivative;
- Zero Crossing Rate: it is the number of times a signal crosses $y = 0$;
- Mean Crossing Rate: it is the number of times a signal crosses $y = mean(y)$.

From FFT

In this way for each file containing 2 seconds of raw data on the three axes, we are able to obtain 11 features for each axis, concatenated on the same line, for a total of 33 features.

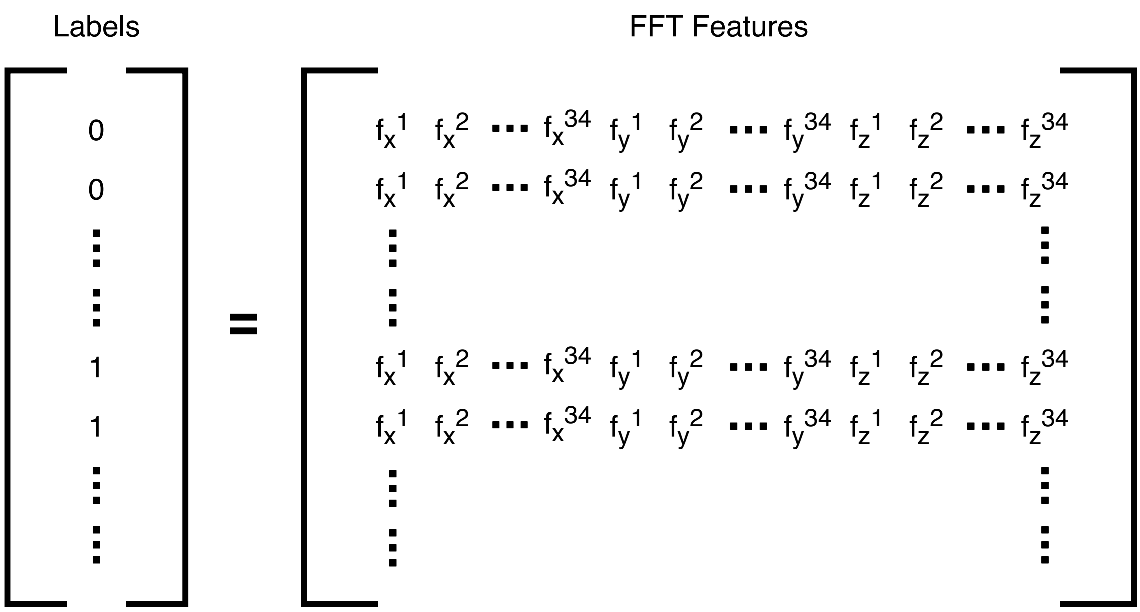


Figure 5.7. FFT features structure.

From DWT

The DWT is used to split a signal into different frequency sub-bands, as many as possible. If the different types of signals exhibit different frequency characteristics, this difference in behavior has to be exhibited in one of the frequency sub-bands. So if we generate features from each of the sub-band and use the collection of features as an input for a classifier and train it by using these features, the classifier should be able to distinguish between the different types of signals [36].

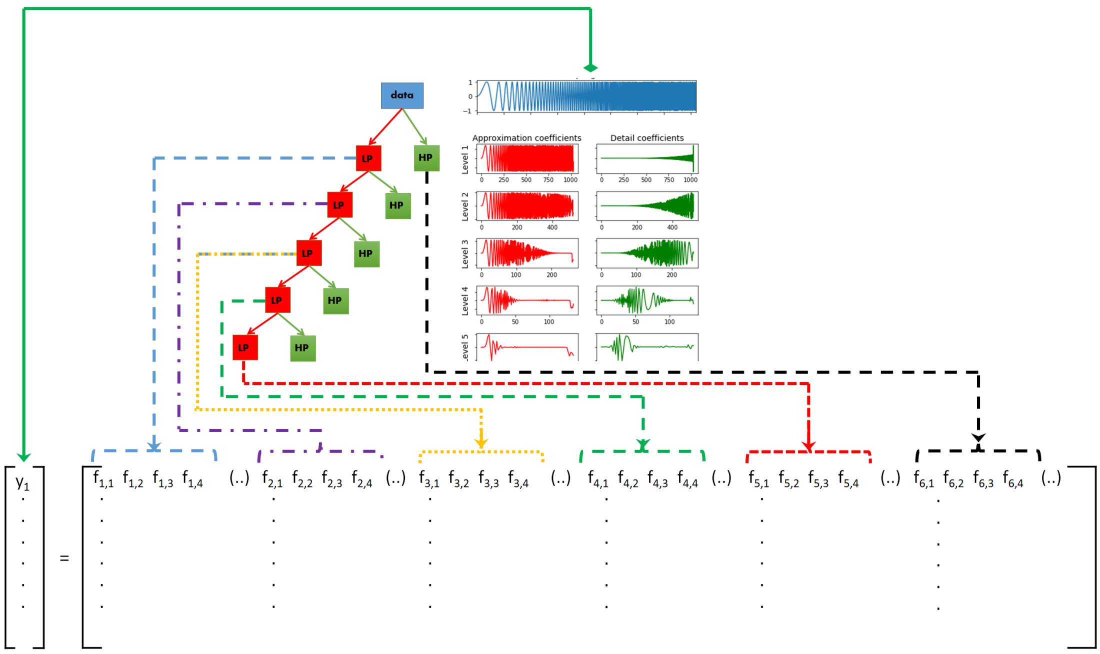


Figure 5.8. DWT features structure [36].

In our case, is used a maximum decomposition level.
Since in every subsequent stage the number of samples in the signal is reduced with a factor of two, at some stage in the process the number of samples in our signal will become smaller than the length of the wavelet filter and we will have reached the maximum decomposition level.
Therefore each raw signal contained in a file is decomposed into nine parts for axes, from each part the 12 features described in 5.1.3 are extracted.
Thus we have a Dataframe containing 324 features.

Correlation Matrix

Correlation is a statistical term that in common usage refers to how close two variables are to having a linear relationship with each other.
Features with high correlation are more linearly dependent and hence have almost the same effect on the dependent variable. So, when two features have high correlation, we can drop one of the two features [37].
For this reason, the correlation matrices are analyzed for both FFT and DWT, since especially the latter has a large number of features.

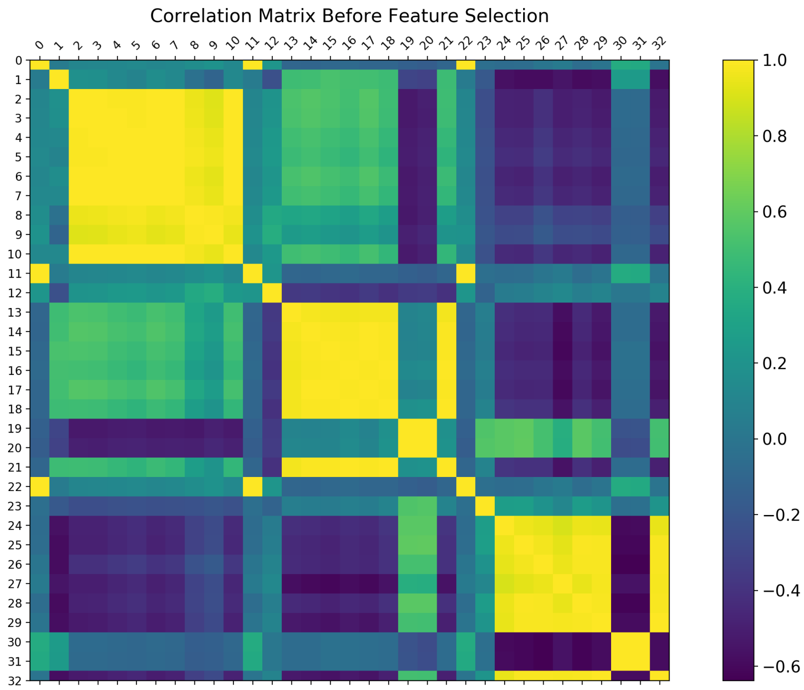


Figure 5.9. FFT Correlation Matrix.

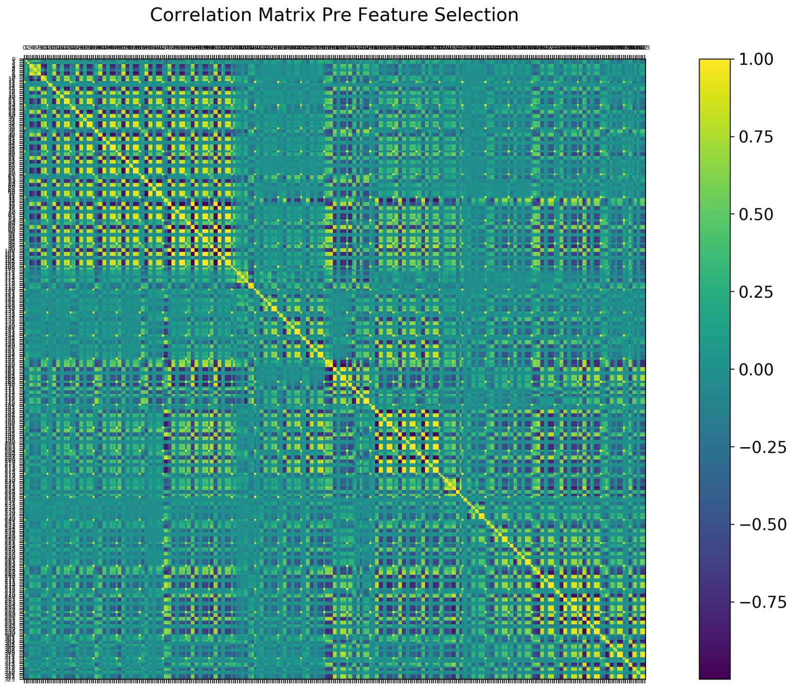


Figure 5.10. DWT Correlation Matrix.

As we can see from the figures above, in both cases we can see a high correlation of some features.

5.1.4 Features Selection

All of the features we find in the dataset might not be useful in building a machine learning model to make the necessary prediction. Using some of the features might even make the predictions worse. So, feature selection plays a huge role in building a machine learning model.
Feature selection, the process of finding and selecting the most useful features in a dataset, is a crucial step of the machine learning pipeline. Unnecessary features decrease training speed, decrease model interpretability, and, most importantly, decrease generalization performance on the test set [38].
For this reason, a *FeatureSelector* has been implemented that efficiently selects the most important features using some of the most common feature selection methods:

- Collinear features: are features that are highly correlated with one another. In machine learning, these lead to decreased generalization performance on the test set due to high variance and less model interpretability.

- Features with zero importance in tree-based model: this method is designed only for supervised machine learning problems where we have labels for training a model and is non-deterministic. The *FeatureSelector* finds feature importances using the gradient boosting machine from the LightGBM library. The feature importances are averaged over 10 training runs of the GBM in order to reduce variance. Also, the model is trained using early stopping with a validation set to prevent over fitting to the training data.

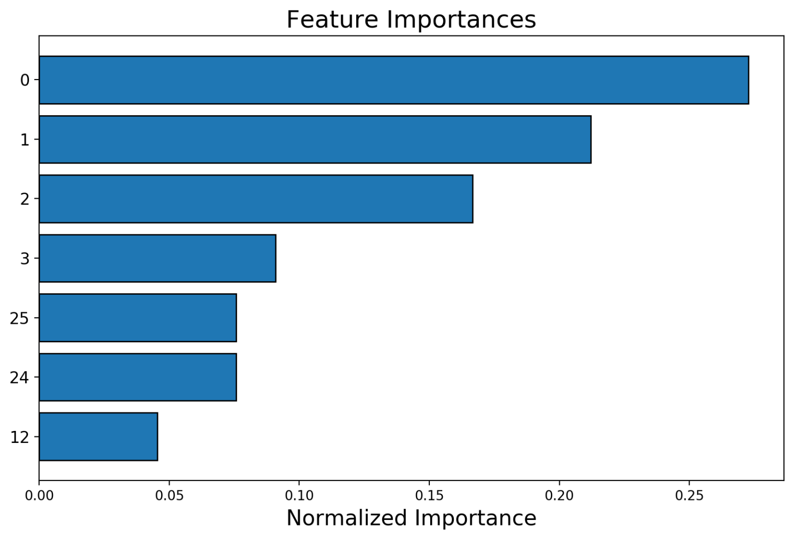


Figure 5.11. Features Importances.

- Features with low importance: using the feature importances from the model for further selection, It finds the lowest importance features that do not contribute to a specified total importance. Based on the plot of cumulative importance and this information, the gradient boosting machine considers many of the features to be irrelevant for learning.

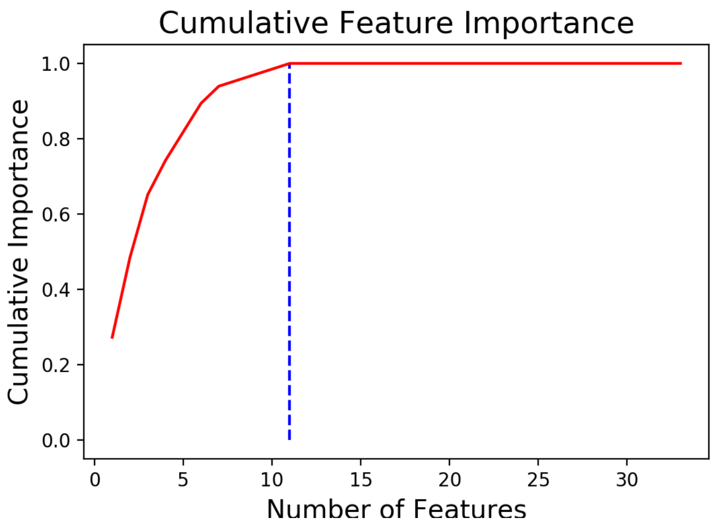


Figure 5.12. Cumulative Features Importance.

The vertical line in Fig.5.12 is drawn at threshold of the cumulative importance, in this case fixed at 99%.

Once we have identified the features to discard, the *FeatureSelector* returns a new Dataframe with the remaining features.

Correlation Matrix after Features Selection

The new database was created and the correlation matrix was calculated again with the remaining features.

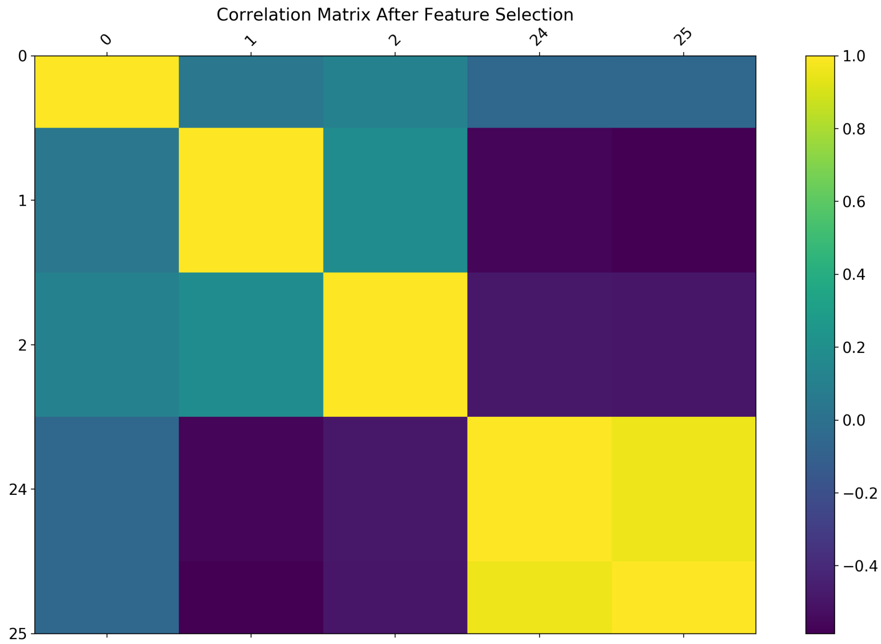


Figure 5.13. FFT Correlation Matrix.

As we can see from the figure above, respect to the correlation matrix in Fig.5.9 features generated by FFT have been reduced from 34 to 5. Namely a reduction of 84%.

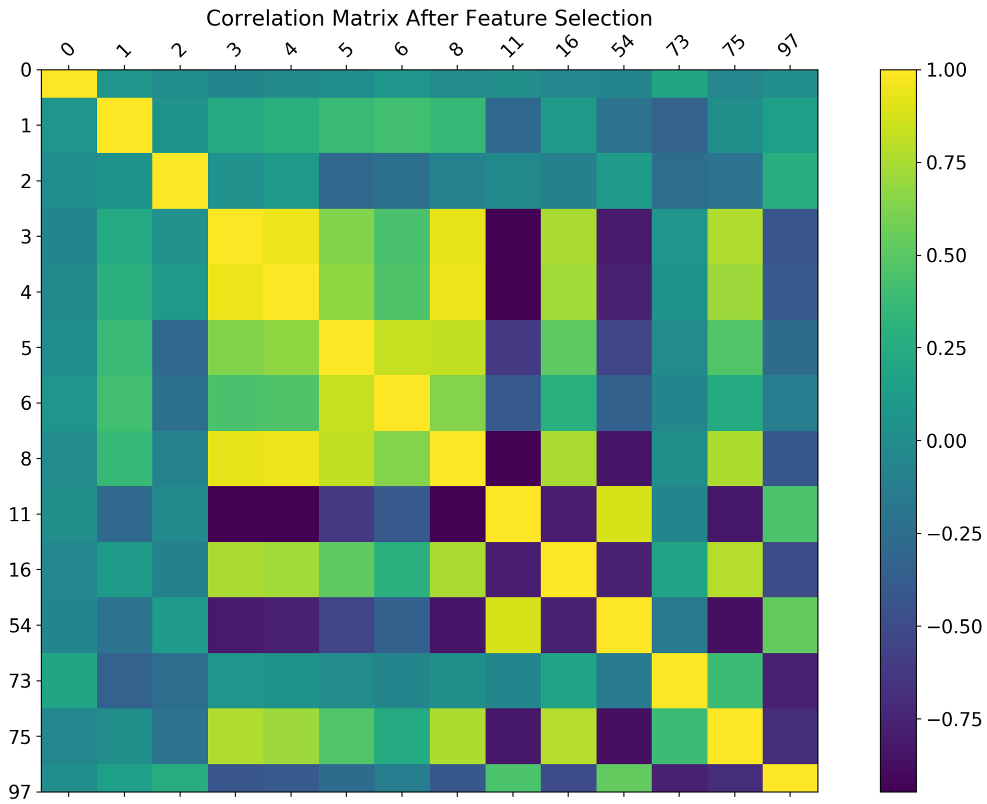


Figure 5.14. DWT Correlation Matrix.

From the DWT point of view, we have a reduction of the features from 325 to 14. Reaching a reduction of 95.6%.

5.1.5 Standardize features

The last step of the preprocessing phase was to apply a normalization technique to the values obtained after the features selection.
Normalization is very useful to ensure that all features have values contained in the same order of magnitude, and it is essential in cases where it is necessary to perform calculations of distance between the data.
In the analyzes that will be described later, the data did not always undergo a normalization process. However, several tests were carried out and the best results were subsequently chosen.
The formula chosen for normalization is the same that is widely used in statistical standardization, which leads the variables to a distribution with an average of 0 and a standard deviation of 1:

$$Z = \frac{(X - u)}{s}$$

Where Z represents the values of the features after normalization, X the non-normalized input features, while u and s are respectively the mean and the standard deviation of the distribution of a feature.
This standardization is carried out in Python through a function specially prepared by Scikit-learn called StandardScaler().

5.2 Classification

The first goal that we wanted to achieve for this project was to be able to discern between a state of normal machine operation and one with some anomalies. Therefore, we want to obtain a service capable of providing diagnostics.
We have chosen to tackle the problem by using machine learning algorithms for solving multiclass classification.
The models in question are shown below.

- **Binary Classifier**: compares two methods of assigning a binary attribute, one of which is usually a standard method and the other is being investigated;
- **Gradient Boost**: ensemble classifier which combines 100 decision trees that are used sequentially, so that each classifier of the chain is trained on the residual errors of the previous model;
- **K Nearest Neighbors**: simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g. distance functions). A case is classified by a majority vote of its neighbors, with the case being assigned to the class most common among its K nearest neighbors measured by a distance function. After several attempts, five neighbors are chosen, since they give the best result;
- **Naive Bayes**: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It assume that the value of a particular feature is independent of the value of any other feature, given the class variable;
- **Neural Network**: a feed forward neural network is used, with N input neurons corresponding to the N features that depends from the chosen Dataframe, two hidden layers both from 16 neurons and an output layer with as many neurons as the classes in the dataset used. A Drop Out function is insert between

the layers and a Callback function to early stop training and save the best model is added, both to reduce overfitting.

To analyze the scenario, a dataset was therefore created containing two different classes, one of correct functioning and one of anomalous functioning. 2200 input signals were collected for each class.

Four different groups of features are generated:

- Features Extraction from FFT;
- Features Extraction form DWT;
- Features Extraction from FFT plus Features Selection;
- Features Extraction from DWT plus Features Selection.

Each classification model is tested with these groups of features in order to understand which is the better solution.
For each classification model, 50% of the dataset was used for training and the remainder for testing.
The results obtained are reported for each algorithm, highlighting the performance in terms of:

- Precision: is the ability of a classifier not to label an instance positive that is actually negative. For each class it is defined as the ratio of true positives (TP) to the sum of true and false positives (FP);

$$Precision = TP/(TP + FP)$$

- Recall: is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives (FN);

$$Recall = TP/(TP + FN)$$

- F1-Score: is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0;

$$F1 - Score = 2 * \frac{Recall * Precision}{(Recall + Precision)}$$

- Accuracy: is the fraction of predictions our model got right;

$$Accuracy = \frac{TP + TN}{(Recall + Precision)}$$

- Training Time: the time needed to finish the training.

The overall precision and recall values are obtained by calculating the individual values for each class and averaging them. This procedure is possible because the classes of the dataset are balanced.
The results are listed in the following tables:

Features extracted from FFT

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Binary Classifier	0.989	0.988	0.988	0.988	1.0741
Gradient Boost	0.982	0.982	0.981	0.981	2.6464
K Nearest Neighbors	0.980	0.979	0.979	0.980	0.0052
Naive Bayes	0.950	0.950	0.944	0.945	0.0015
Neural Network	0.986	0.986	0.986	0.986	2.8417

Features extracted from DWT

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Binary Classifier	0.993	0.993	0.993	0.993	1.4897
Gradient Boost	0.989	0.988	0.988	0.988	3.3176
K Nearest Neighbors	0.987	0.981	0.981	0.987	0.1870
Naive Bayes	0.965	0.965	0.954	0.965	0.0138
Neural Network	0.990	0.990	0.990	0.990	3,5736

As we can be deduced from the obtained data, the results are very positive, for all the models that are used. In fact we reach more than 90% in all performances in both cases.
But on closer inspection, we can see how the performance in the case of features extracted from DWT has increased by about 1% compared to those extracted from FFT, at the expense, however, of an increase in the time required to perform the algorithmic training, ranging from 20% in the case of the Gradient Boost up to a significant 97% in the case of the K Nearest Neighbors, with an average increase on all algorithms by 50%.

In the figures below the data just described are graphically represented in order to facilitate their interpretation.

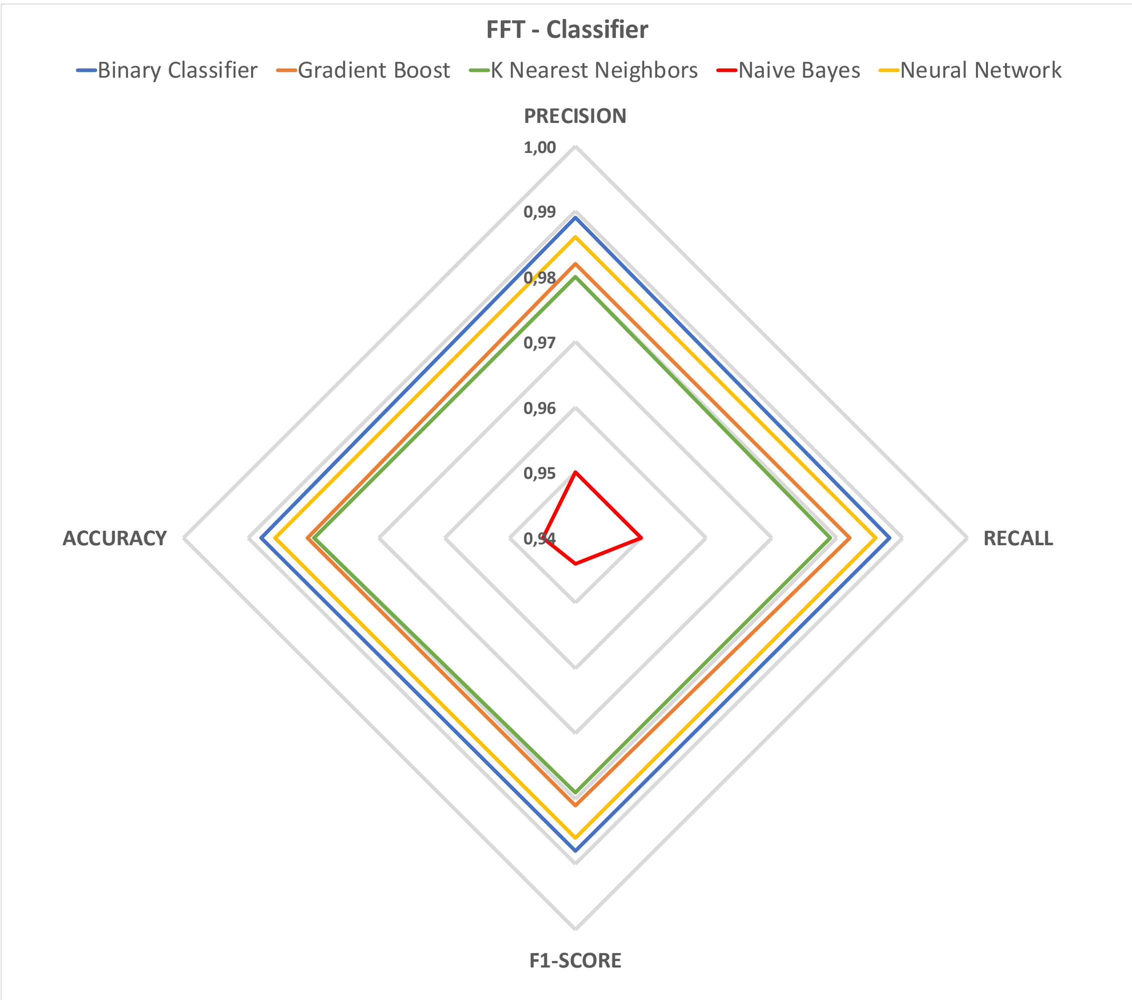


Figure 5.15. Performances of FFT Classifier.

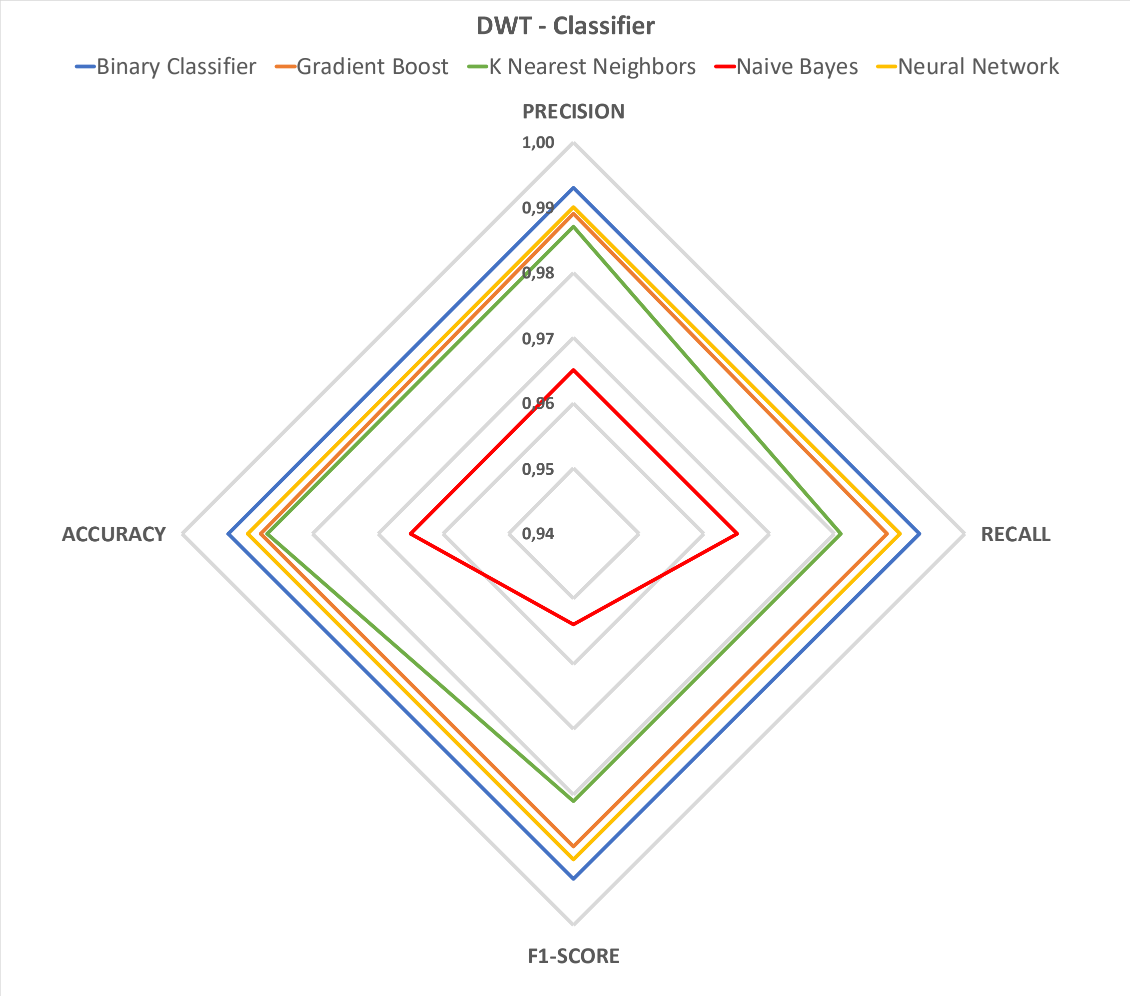


Figure 5.16. Performances of DWT Classifier.

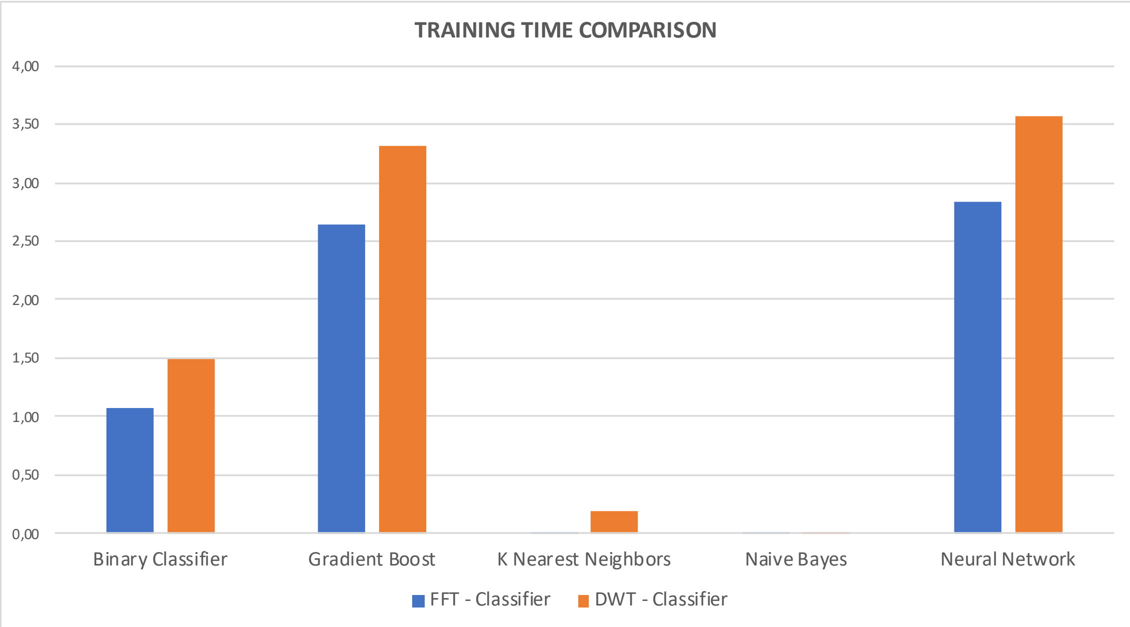


Figure 5.17. Comparison training time between FFT and DWT Classifier.

This result meets expectations since the better quality of the functionalities generated with DWT means that all algorithms can increase the overall performance, but the big difference in the number of features makes training slower in the second case, reaching more than 3s in two algorithms Fig.5.17.

In order to increase the training speed trying to maintain the highest possible performance, the algorithms behaviors were evaluated after the previously generated features were subjected to a further preprocessing step of the data, namely, the Feature Selection algorithm that we talked about in Chapter 5.1.4, by which, given the input features, it returns a restricted subset without loss of useful information for the functioning of the algorithms.

Features extracted from FFT plus Features Selection

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Binary Classifier	0.969	0.968	0.968	0.968	0.9821
Gradient Boost	0.957	0.957	0.956	0.956	2.6804
K Nearest Neighbors	0.951	0.950	0.950	0.951	0.0008
Naive Bayes	0.903	0.903	0.897	0.898	0.0008
Neural Network	0.971	0.971	0.971	0.971	3.9784

Features extracted from DWT plus Features Selection

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Binary Classifier	0.973	0.973	0.973	0.973	0.9446
Gradient Boost	0.964	0.963	0.963	0.963	2.7619
K Nearest Neighbors	0.957	0.952	0.952	0.957	0.0010
Naive Bayes	0.917	0.917	0.906	0.917	0.0008
Neural Network	0.975	0.975	0.975	0.975	4,2039

Analyzing these new results, we note different aspects that need to be explored: first of all, the highest precision is maintained by the algorithms powered by the Features extracted from the DWT compared to those powered by the FFT.
The average loss of performance compared to previous DFs without Feature Selection is of the order of 2.8% in the case of FFT reaching the maximum peak in the Naive Bayes algorithm with a Precision loss of 5%.
While in the case of DWT there is an average loss of 1.9% with a maximum of 3% in Naive Bayes.
Regarding the execution time of the training, there is an average decrease of 35% in the case of FFT with Feature Selection compared to the case without it, since we have a decrease the features from the original 34 to 4. While in the case of DWT with FS the average time decreases by 62% compared to the first case, with a reduction of 311 features. This means that the execution times of the algorithms with Features generated via FFT and DWT both processed via Feature Selection are more similar than in the previous case.
The only algorithm that behaves differently is the Neural Network, as in both cases the smaller number of features means that the number of Epochs needed to reach an adequate level of accuracy increases, consequently increasing the training time.

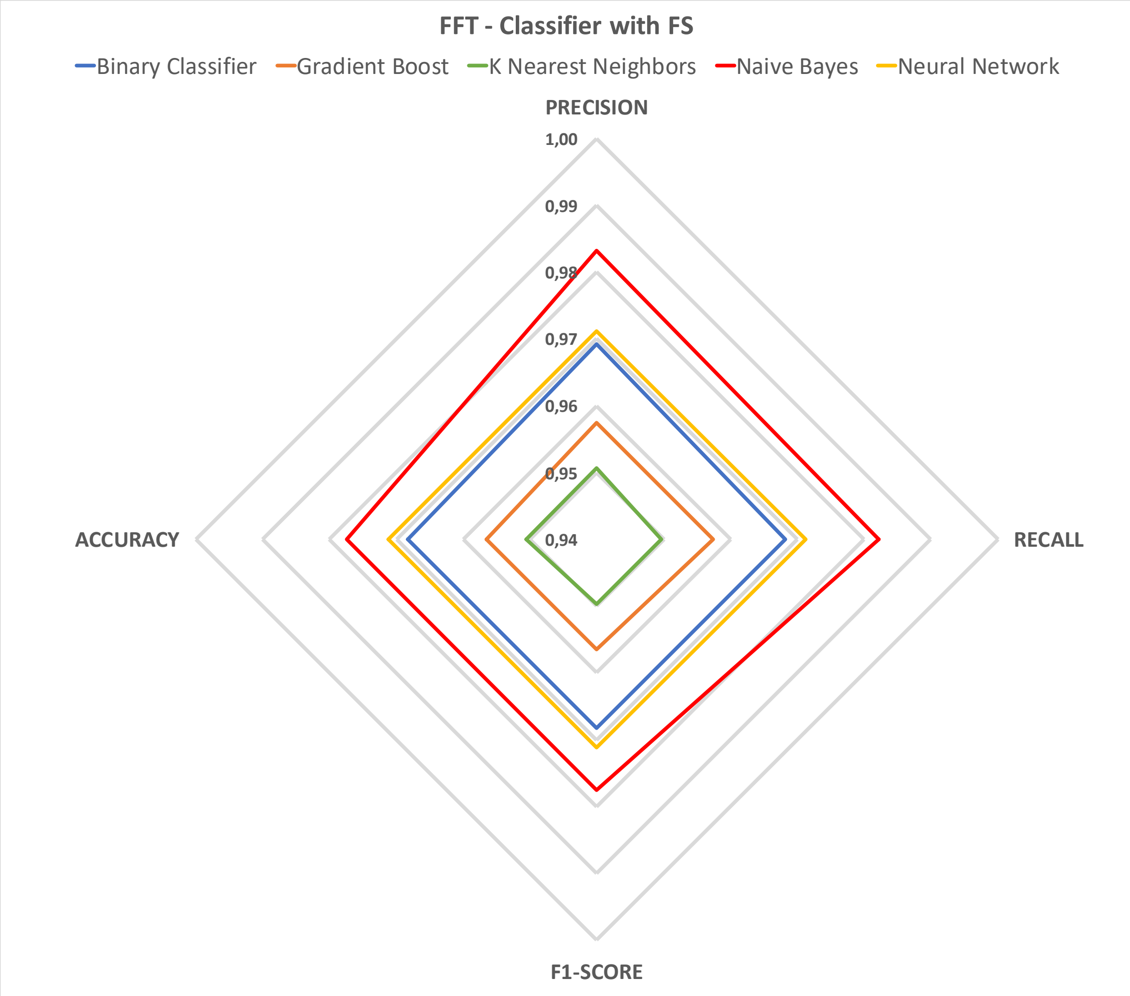


Figure 5.18. Performances of FFT Classifier with FS.

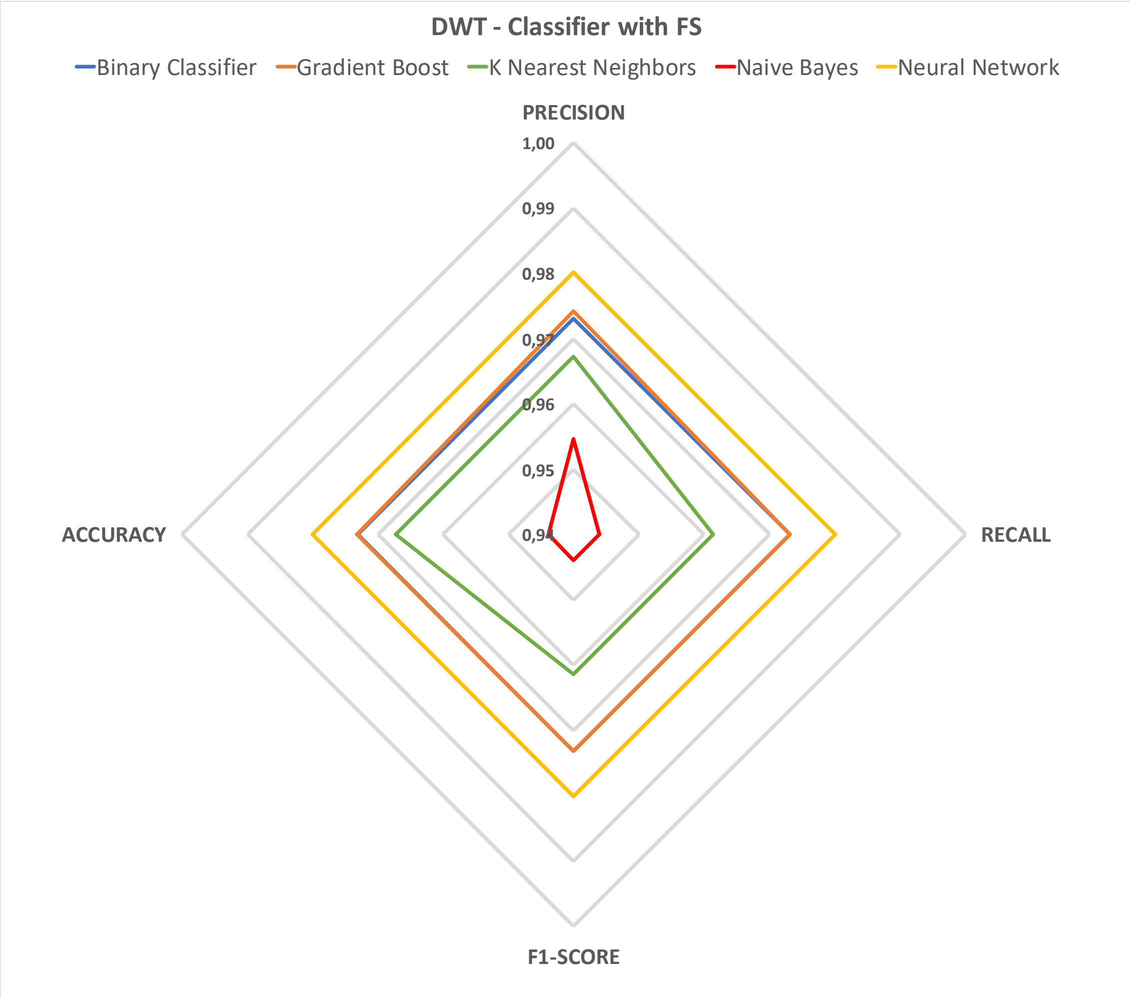


Figure 5.19. Performances of DWT Classifier with FS.

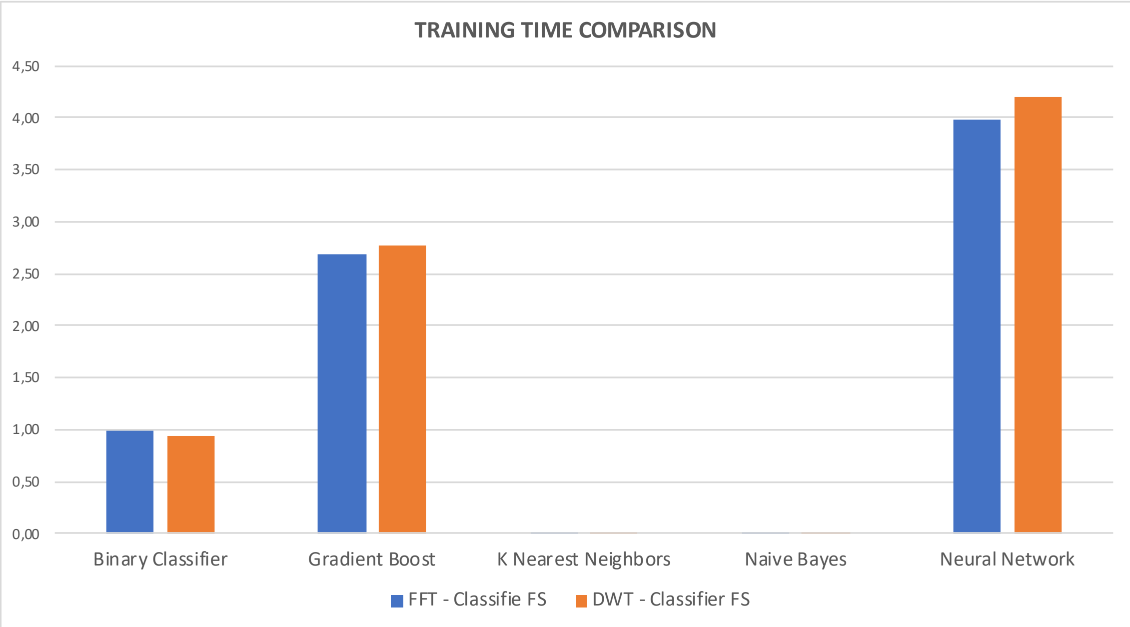


Figure 5.20. Comparison training time between FFT and DWT Classifier after Feature Selection.

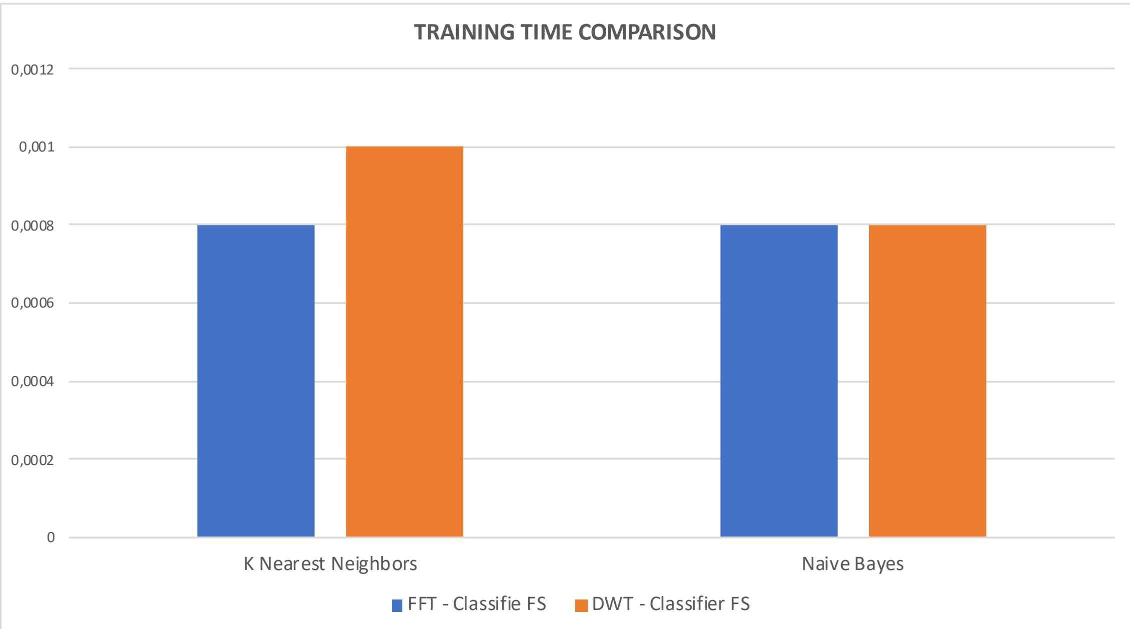


Figure 5.21. Zoom in K Nearest Neighbors and Naive Bayes models.

5.2.1 Observations

Thanks to the tests described in this section, it is possible to demonstrate that the machine learning models are able to distinguish different operating states of the machine through the values collected by the chosen sensor.

In particular, it has been shown that the features extracted from the coefficients calculated using the DWT and through a subsequent selection of the features themselves we can generate an excellent dataframe for classifying the operating states of a rotating machine, with greater precision than a features calculated through the FFT, while maintaining comparable training times, thus making the choice of the DWT plus Feature Selection excellent as regards Classification problems.

However, two serious critical issues have been identified.

Classification models require a training dataset in which there must be examples relating to all the classes that you want to recognize in the testing phase. This means that the machine must actually go into a state of failure for the creation of the dataset. If in the scenario built specifically for the thesis project it was easy to obtain it, the same does not apply to real use cases where you want to apply the system to machines already in production. It is too long and expensive to be carried out in real production lines.

Furthermore, the models of classification during testing are limited to labeling the inputs they receive with one of the classes present in the dataset. This means that if a new type of anomaly occurs on the machine, it will necessarily be included in one of the known cases even if the patterns of the particular input differ from each of them.

A classification model is therefore considered insufficient, but models for anomaly detection problems are analyzed.

5.3 Anomaly Detection

An *anomaly* is defined as something that differs from what is normal or regular. The idea behind this approach is precisely to be able to define the characteristics or patterns of elements considered normal and then highlight when something distances itself from this modeling.

By applying it to the case study, we want to create a model trained exclusively on signals collected during the normal operation of the machine, to then be able to identify a generic anomaly. Obviously, the possibility of distinguishing the various faults is lost, but it allows us to solve the critical points highlighted for the classification models, which is the need to wait for fault occurrences or cause physical damage to the machine and the restriction to known fault cases.

The models used for this approach are:

- **Isolation Forest**: in order to isolate a data point the algorithm recursively generates partitions on the sample by randomly selecting an attribute and then randomly selecting a split value for the attribute, between the minimum and maximum values allowed for that attribute;

- **One Class SVM**: a particular type of support vector machine specialized for novelty detection tasks, namely, recognition of rare or never encountered events. Specifically a Radial Basis Function Kernel is used, with parameter **nu** at 0.1;

- **Elliptic Envelop**: a method that tries to figure out the key parameters of our data general distribution by assuming that our entire data is an expression of an underlying multivariate Gaussian distribution.

- **Local Outlier Factor**: it computes the local density deviation of a given data point with respect to its neighbors. It considers as outliers the samples that have a substantially lower density than their neighbors.

They are all an unsupervised learning algorithm, this means that we are able to leave the model to work on its own to discover information.
Assuming to use good state signals as training, the model will aim to identify the signals collected in the state of failure as a novelty.

To verify the ability to distinguish fault situations with anomaly detection, the same 4 groups of features generated during classification are used. However, the models have been trained using only 50% of the features corresponding to a good state of the machine.

For the test phase, the remaining 50% of the data in which the machine is in good condition and the data collected in fault situations were used, in order to verify both the machine capable of recognizing the anomalies but also that it is able to recognize the normal operation of the machine.

Below are listed the models results:

Features extracted from FFT

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Isolation Forest	0.871	0.878	0.935	0.933	0.1478
One Class SVM	0.882	0.891	0.942	0.939	0.0080
Elliptic Envelope	0.786	0.777	0.874	0.877	0.6921
Local Outlier Factor	0.895	0.904	0.949	0.947	0.0157

Features extracted from DWT

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Isolation Forest	0.981	0.984	0.992	0.991	0.1870
One Class SVM	0.773	0.735	0.737	0.752	0.0662
Elliptic Envelope	0.790	0.783	0.883	0.881	9.6550
Local Outlier Factor	0.875	0.868	0.870	0.873	0.1300

The analysis of the results shows that the detection of the anomaly status is possible using both data frames.
We achieve the best result with the usage of Isolation Forest algorithm which uses Features extracted from DWT, with an increase in training time of 20% respect the FFT model.

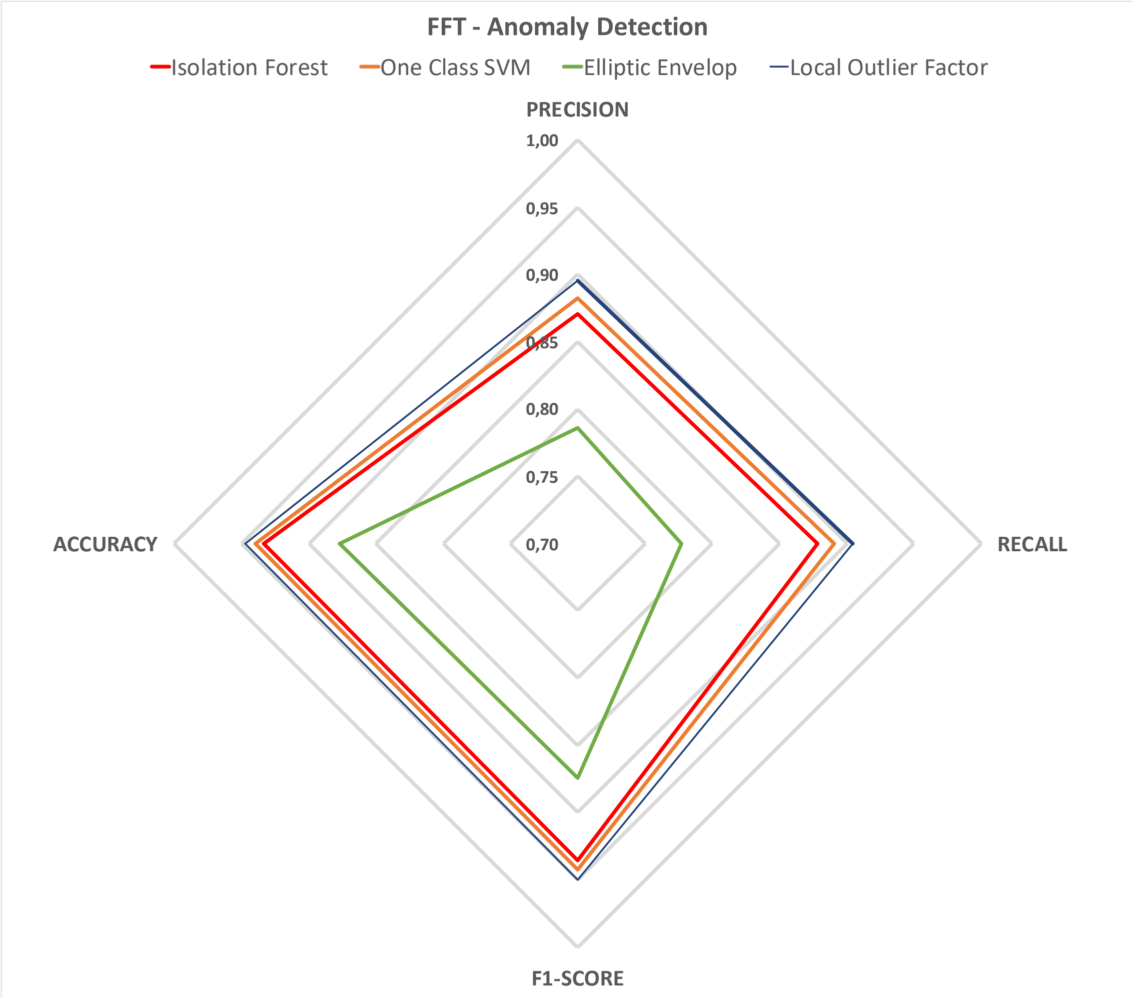


Figure 5.22. Performances of FFT Anomaly Detection.

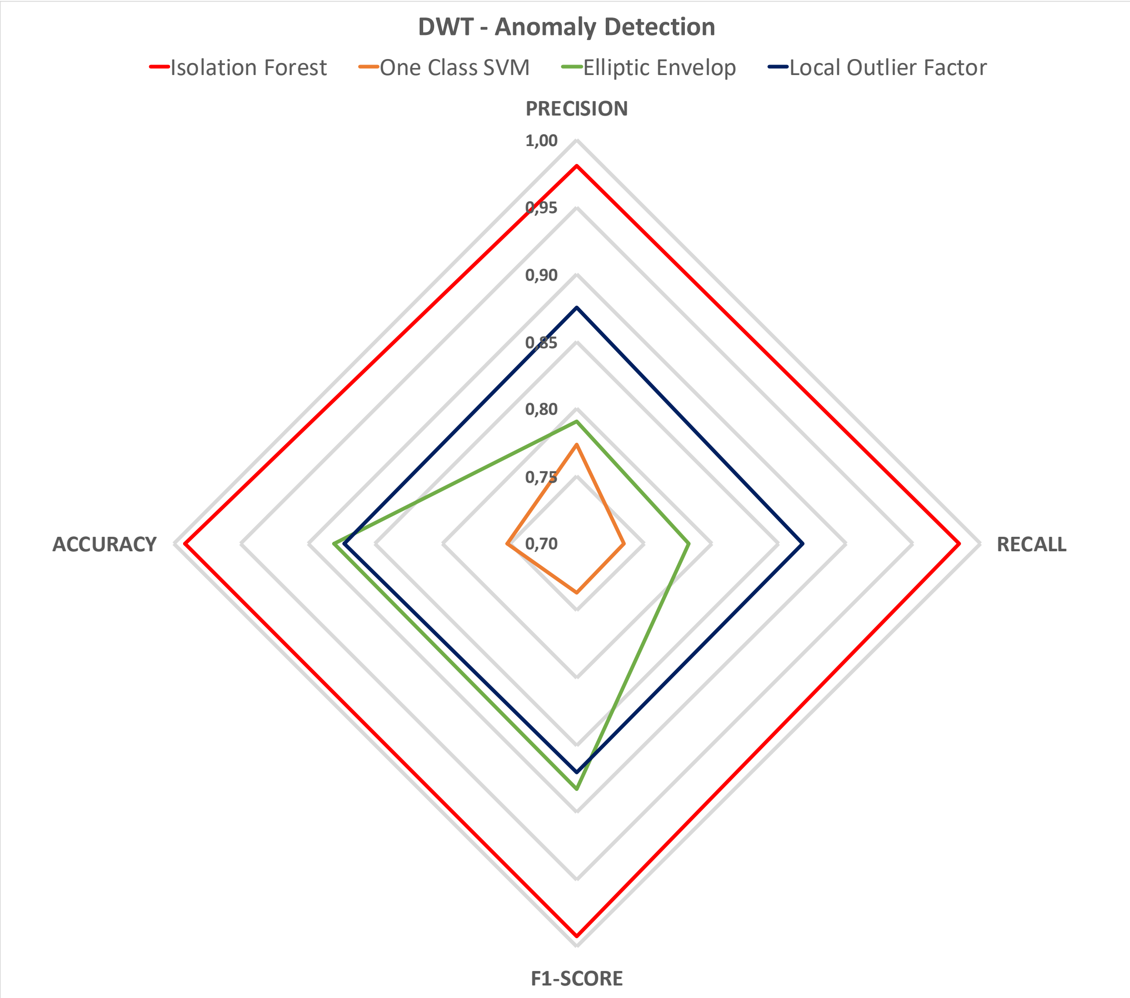


Figure 5.23. Performances of DWT Anomaly Detection.

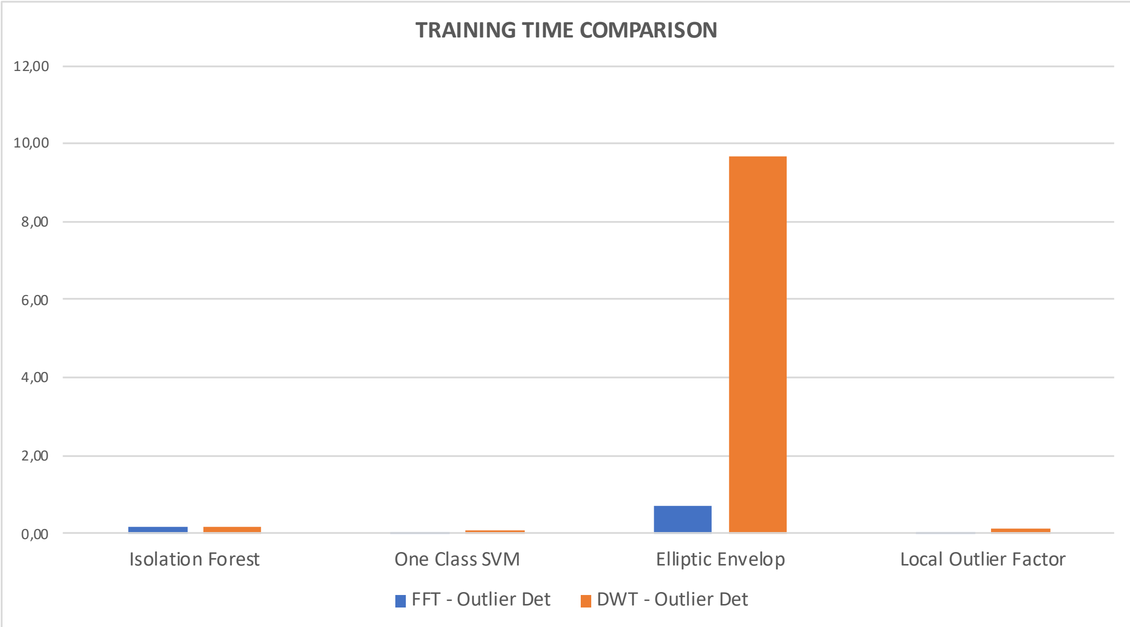


Figure 5.24. Comparison training time between FFT and DWT Outlier Detection.

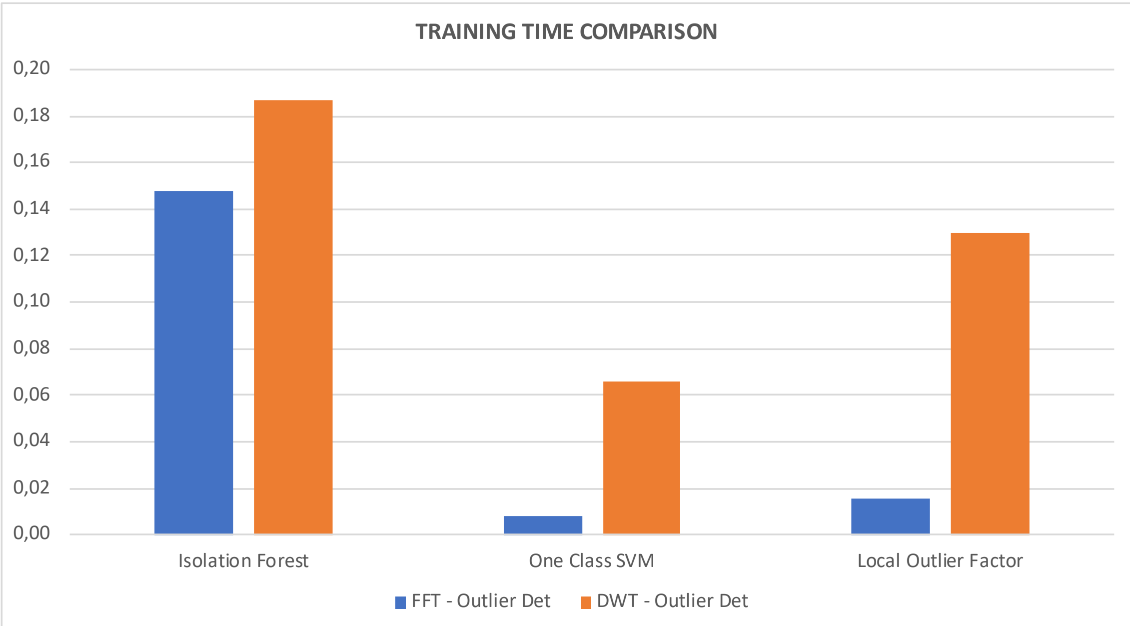


Figure 5.25. Zoom in Isolation Forest, One Class SVM and Local Outlier Factor.

In general, what is noted is that the performance has decreased compared to the use of the classification methods described in the previous section.

The other models such as SVM and LOF perform better using the features extracted from FFT, while Elliptic Envelop perform better in DWT model but covariance estimation not perform well in high-dimensional settings also the excessive training time makes it unusable.

As done in the previous chapter the above mentioned algorithms are analyzed by operating the Feature Selection.
In this case, since it is not a supervised machine learning problem, it was not possible to apply the method for the elimination of the Zero Importance Features which we discussed in the Chap.5.1.4. Therefore we will not have a massive reduction in features as in the previous cases but we will still get good results.

Features extracted from FFT plus Features Selection

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Isolation Forest	0.941	0.945	0.940	0.940	0.1324
One Class SVM	0.941	0.945	0.939	0.940	0.0065
Elliptic Envelop	0.893	0.889	0.878	0.878	0.5874
Local Outlier Factor	0.952	0.956	0.952	0.952	0.0087

Features extracted from DWT plus Features Selection

Model	Precision	Recall	F1-Score	Accuracy	Training Time (s)
Isolation Forest	0.966	0.959	0.962	0.962	0.1674
One Class SVM	0.633	0.580	0.551	0.611	0.0587
Elliptic Envelop	0.889	0.883	0.871	0.871	8.4443
Local Outlier Factor	0.652	0.590	0.560	0.621	0.0957

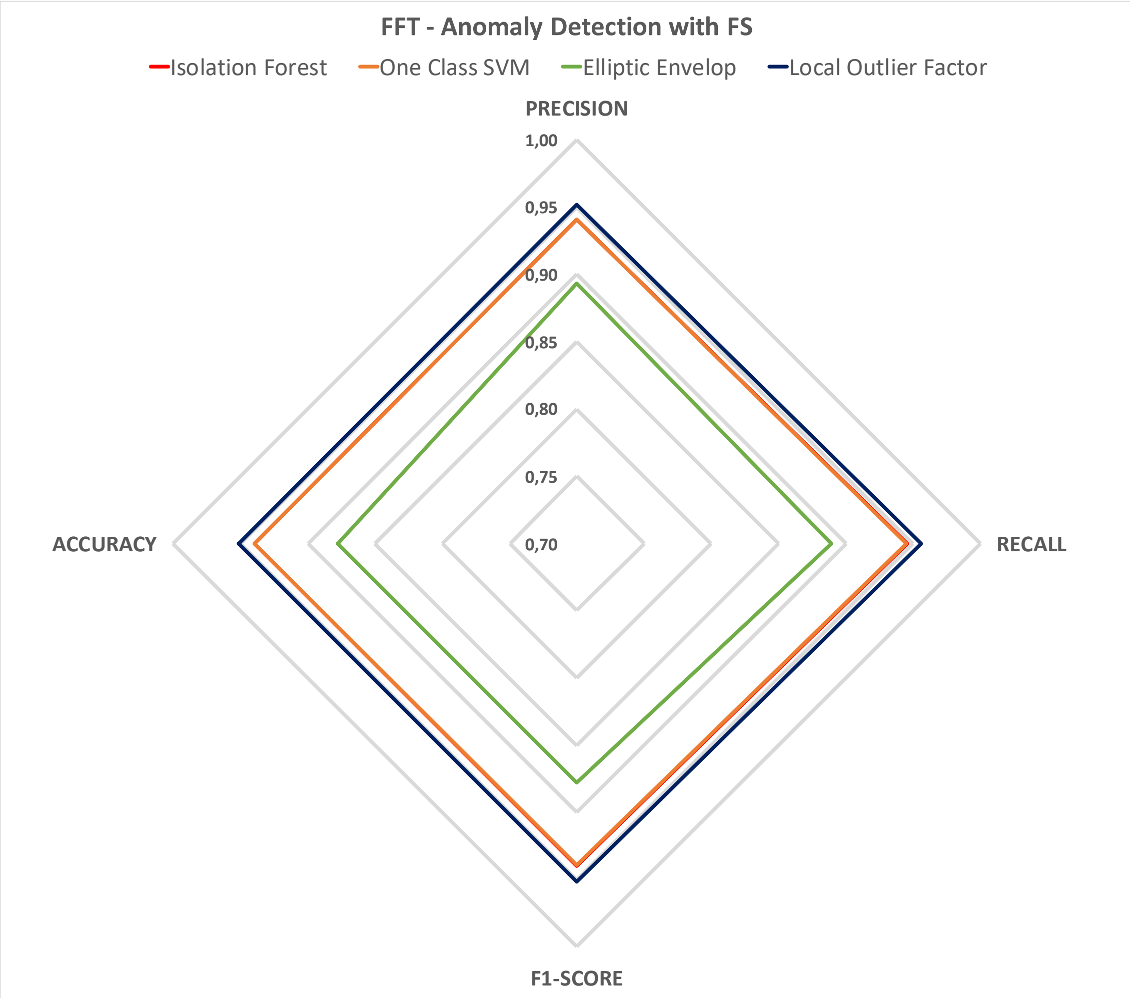


Figure 5.26. Performances of FFT Anomaly Detection with FS.

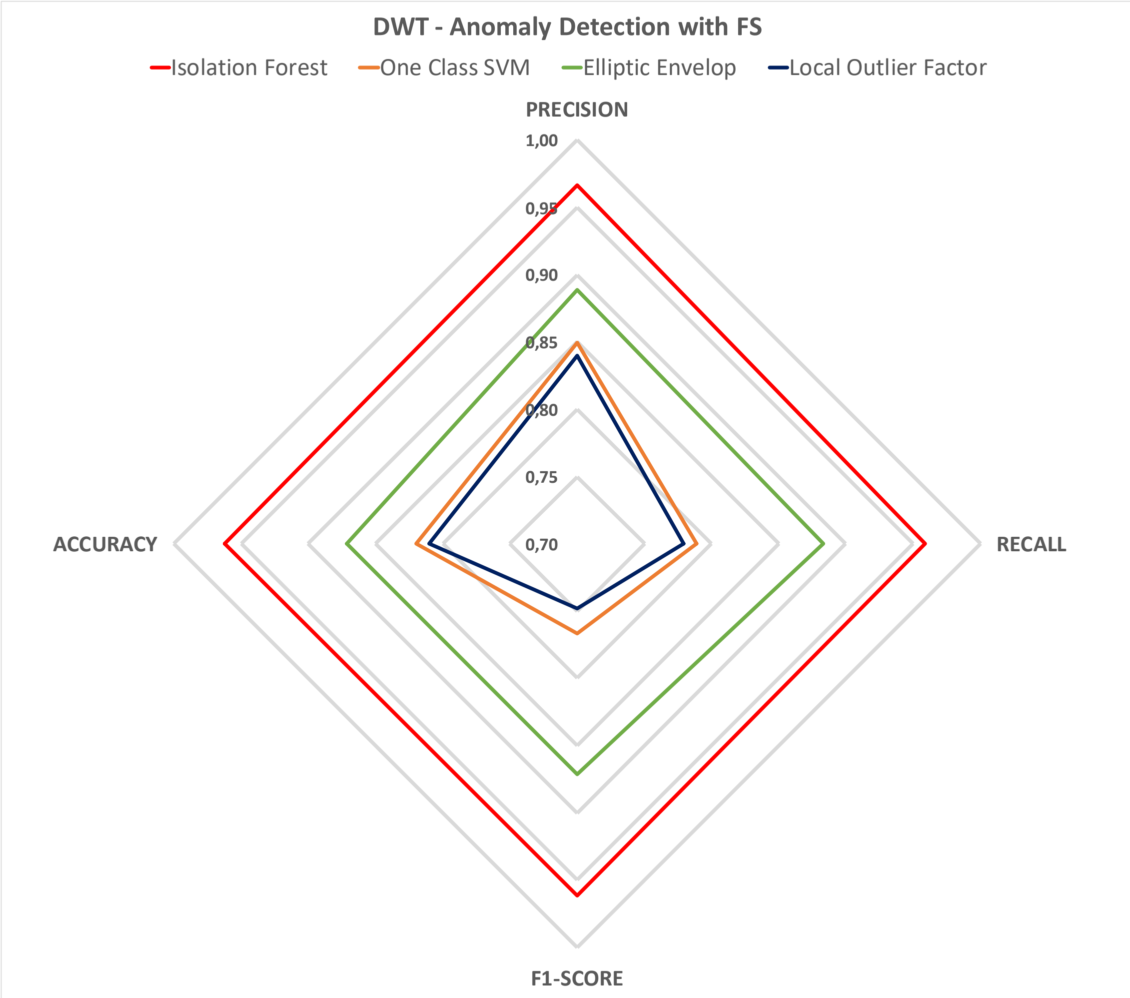


Figure 5.27. Performances of DWT Anomaly Detection with FS.

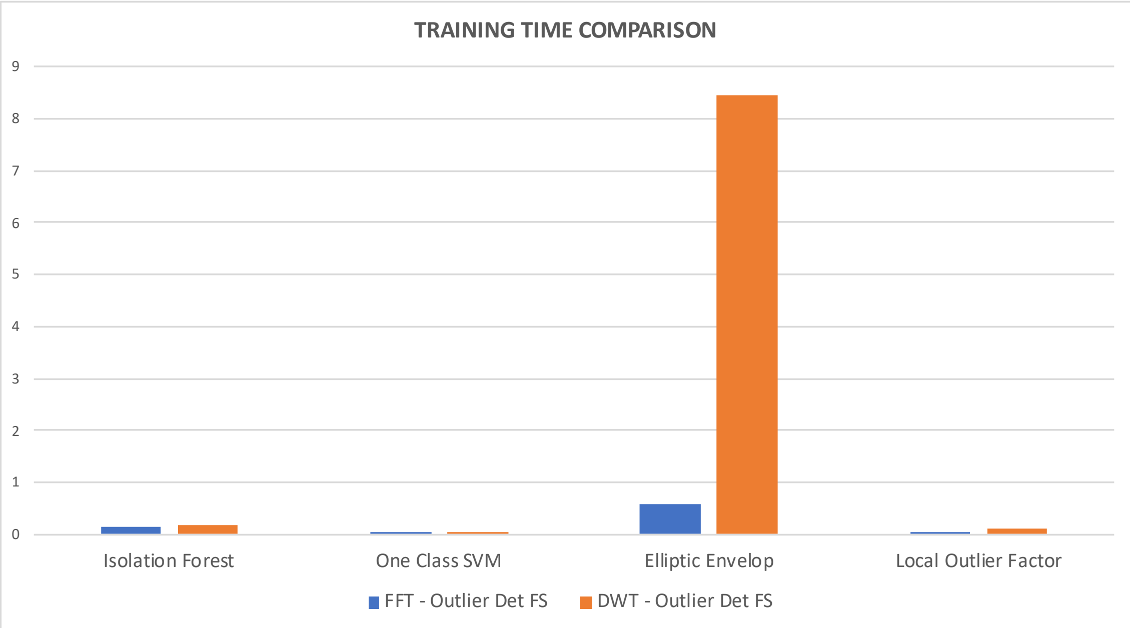


Figure 5.28. Comparison training time between FFT and DWT Outlier Detection after Feature Selection.

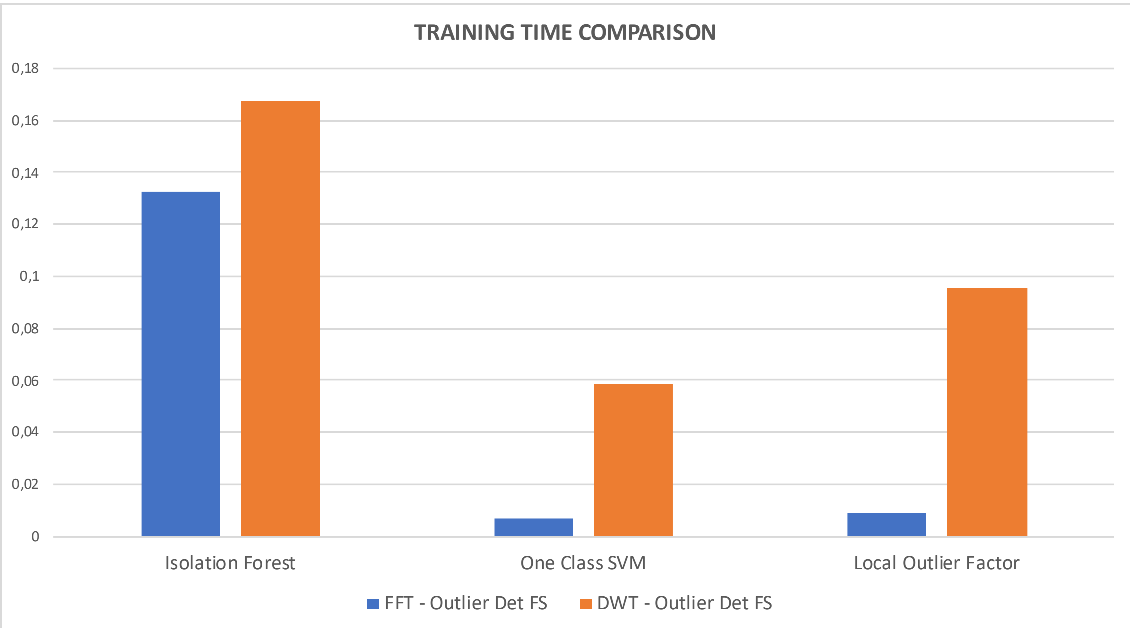


Figure 5.29. Zoom in Isolation Forest, One Class SVM and Local Outlier Factor.

Contrary to what happened during the Classification, the Features Selection has decreased the features in the FFT by 58% compared to 88% in the previous case, while in the DWT the features were reduced by 27% compared to 96% in the previous case.

Performance only increased in the algorithms that used a Dataframe generated by the FFT. In both cases, we find a significant reduction in Training Time, but the reduction of the features in the Dataframe generated by the DWT causes an average reduction in the performance of 13% that makes these algorithms unusable, with the exception of the Isolation Forest which with a 3% reduction in performances and an 11% reduction in training time can still be taken into consideration for a possible use.

5.3.1 Observations

The tests carried out regarding anomaly detection have shown that it is possible to recognize with good precision when entering a generic state of the anomaly by using in the training phase only examples collected during normal operation.
The choice, in particular, is limited to two types of different models that can be used according to the specific use case, that is, if you want more performance at the expense of a slower training time, you can use the Isolation Forest algorithm powered by features generated by the DWT without operating the Feature Selection.
Instead, if you want to reduce the number of features to be stored and make training more streamlined and faster at the expense of lower performances, you can use the Local Outlier Factor powered by features generated by the FFT and subsequently selected.

The disadvantage of this approach, however, is to lose the distinction and classification between the different failures.

To overcome this limitation, a method that combines different models could be implemented, each trained on a specific class of failure.
The main advantage of this technique is that it is possible to carry out training separately between the models and to combine their results only later.
This means that to introduce a new anomaly class it is sufficient to create a new model based on Isolation Forest, train it on the available examples concerning only the anomaly in question without having to modify the other models. During the distance verification phase, only one more model will have to be consulted.

A further property deriving from this approach is that of being able to identify when a certain input does not belong to any of the known classes.

This is achievable by checking the distance values returned by the models. If the signal to be classified is far from all models, then it is most likely an unknown class.

Chapter 6

Future Developments

The final objective of the reported research is to use the models for prognostic applications. To this, a future required development is to proceed with the validation of the models on (different) operating machines, using a specific industrial sensor, in order to obtain raw data of higher quality than that used for the tests. A first test application is foreseen on machinery used in the waste transformation, thanks to the collaborative project BioEnPro4TO, co-funded by Regione Piemonte. To this, sensors with proper IP protection and, in some cases, an embedded wireless communication module, have been shortlisted. Moreover, from the observations produced in the previous chapter, it is clear that in the case of Classification, machine learning models are able to distinguish with remarkable results between the different operating states of the machine once properly labeled, but this potential cannot be exploited when we are dealing with machinery already in production that does not have adequate historical data. Furthermore, the classification is limited to known faults, this means that if a new fault occurs it will be classified into one of the known ones even if the input pattern will differ from each of them.
Regarding Anomaly Detection, the developed models have shown good precision in recognizing when the machine enters a generic state of the anomaly, using for the training phase only data regarding the normal operating state. The disadvantage is losing the ability to distinguish between different failures.
In order to maintain the good accuracy of the classification models without losing the advantages introduced by the anomaly detection, a solution for CBM problems is proposed that combines the two types of models.

6.1 Combination of Classification and Anomaly Detection

The solution uses the Binary Classifier with Dataframe generated by the DWT plus the selection of features, presented in the Chap.5.2, as a classification model with the average of the best results and an Isolation Forest with Dataframe generated by the DWT without selecting the features for each state wants to classify.
The main feature of the proposed technique is that it is not necessary to have all the data for training for each class from the beginning, but they can be added later. The entire process is described below.

- It is supposed to start using the system without having already collected and labeled a dataset;
- The machine's normal operating status is trained, collecting data from the sensors and using them to train an anomaly detection model implemented by a Binary Classifier;
- At this point you can already go into the testing phase, using the anomaly detection model to detect any deviations from the normal case, which will then be reported as a generic anomaly;
- When an anomaly is found, a system user can perform labeling on the individual signals, classifying them with a specific class of fault. In this case, a new Binary Classifier is created, trained only on the new set of labeled data. The procedure can be repeated several times for each new fault encountered over time;
- When more than a single anomaly detection model is available, in addition to the generic anomaly detection, it is also possible to classify the known states, using the method described in Chap.5.3 even if the main purpose is to highlight when an input signal is distant from all models, thus signaling the presence in an unknown state;
- In parallel to what has been described so far, from the moment in which two or more classes are known, an Isolation Forest is also used, which must be readjusted each time the user classifies new inputs. The classifier is used when the other models indicate that it is not a new class, to distinguish the specific type of anomaly.

6.1.1 Observations

The proposed solution shown allows us to detect and classify anomalies that occur on the machine.
It eliminates the need to have a dataset already at the start, allowing you to start and only perform anomaly detection and introduce the classification as the faults actually occur on the machine, using them to be able to distinguish them the following times.

The main limitations encountered are the time required to train the Isolation Forest classes for advanced test phases, in which there are full-bodied Dataframe in terms of classes and inputs. In fact, the classifier must be re-trained on the whole dataset every time it is increased or modified. For this reason, it makes sense to use a Dataframe generated from DWT with feature selection, in this way it is possible to maintain a low amount of stored data and a good training speed of the algorithm. It is therefore assumed that in real scenarios it is an operation that must be carried out only periodically, in the background or at times when the machines are not in production, limiting themselves to using the anomaly detection models as classifiers during the process.

Furthermore, however, the strong dependence on the human factor remains. It is in fact the users who make the classifications. In particular, the problem is important for the training of the machine's normal functioning class. It is essential to better inspect the machine before starting the training period of normal operation, to avoid assuming as normal a state that in reality is not.

Chapter 7

Conclusions

The thesis project carried out aimed at the design and construction of a system capable of enabling and supporting conditions and predictive maintenance activities in Industry 4.0 context.

The developed system has proven to be able to classify and detect anomalies, thus enabling diagnostic functions.
After testing different combinations to extract and select the features, the best method of preprocessing raw data was assessed according to the machine learning model to be used, in particular in the case of Classification and Anomaly Detection.

All the data that was used comes from an ad hoc realized model and not from a simulation. The sensor positioned on the machine has collected physical quantities during the operation of the machine, which has actually been brought into failure states.
The sensor chose to be used is a low-cost and easily available device, demonstrating the fact that it is possible to obtain positive results even without sophisticated instrumentation.

As regards data analysis techniques and anomaly detection, a data-driven approach was chosen, based on machine learning algorithms.

The classifiers have indeed shown excellent accuracy in the distinction between states, but they need to train on a complete dataset of all the faults to be identified.
The parallel use of anomaly detection models allows overcoming this limitation, as they are able to detect when a new input does not fall into any of the known fault classes.
The proposed solution is based precisely on this concept of progressive learning, which starts from the recognition of a generic anomaly by knowing only the state

of normal operation and which allows maintenance workers to increase the set of known faults as they occur on the machines.
Furthermore, a possible future development has been proposed which integrates a combination of Classification and Anomaly Detection algorithms, so as to be able to maintain the good accuracy of Classification models without losing the advantages introduced by Anomaly Detection, such as not needing of all the data from the beginning for training, but to be able to add it later. This allows you to apply predictive maintenance even to machinery that does not have a substantial dataset, thus extending the fields of application.

Bibliography

[1] F. Guedea M. Macchi C. Emmanouilidis E. Sezer, D. Romero. An industry 4.0-enabled low cost predictive maintenance approach for smes. *IEEE*, 1-8, 2018.

[2] R. K. Mobley. An introduction to predictive maintenance. *Elsevier*, 2002.

[3] R. Drath and A. Horch. Industrie 4.0: Hit or hype? *IEEE Industrial Electronics Magazine*, 8(2):56–58, 2014.

[4] W. Lukas H. Kagermann and W. Wahlster. Industrie 4.0: Mit dem internet der dinge auf dem weg zur 4. industriellen revolution. 2011.

[5] W. Wahlster Kagermann, H. and eds. J. Helbig. Recommendations for implementing the strategic initiative industrie 4.0: Final report of the industrie 4.0 working group. 2013.

[6] H. Kagermann. Change through digitization - value creation in the age of industry 4.0. *Management of Permanent Change*, pages 23–45, 2015.

[7] Cyber physical system. `https://ptolemy.berkeley.edu/projects/cps/`.

[8] What is iot? `https://www.iotforall.com/what-is-iot-simple-explanation`.

[9] Algoritmi di machine learning. `https://www.cwi.it/tecnologie-emergenti/intelligenza-artificiale/machine-learning-124626`.

[10] Peter Russell, Stuart J.; Norvig. Artificial intelligence: A modern approach (third ed.). *Prentice Hall*, 2010.

[11] Afshin; Talwalkar Ameet Mohri, Mehryar; Rostamizadeh. Foundations of machine learning. *The MIT Press*, 2012.

[12] UNI. 13306:2018. page 2.1, 2018.

[13] F. Guedea M. Macchi E. Sezer, D. Romero and C. Emmanouilidis. An industry 4.0-enabled low cost predictive maintenance approach for smes. *IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–8, 2018.

[14] R. K. Mobley. An introduction to predictive maintenance. *Elsevier*, 2002.

[15] D. Lin A. K. Jardine and D. Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical systems*

and signal processing, 20:1483–1510, 2006.

[16] H. K. Elminir H. M. Elattar and A. Riad. Prognostics: a literature review. *Complex and Intelligent Systems*, 2:125–154, 2016.

[17] P. Jahnke. Machine learning approaches for failure type detection and predictive maintenance. *Technische Universitat Darmstadt*, 19, 2015.

[18] L. Terrissa A. Lahmadi and N. Zerhouni. A data-driven method for estimating the remaining useful life of a composite drill pipe. *International Conference on Advanced Systems and Electric Technologies (IC ASET)*, pages 192–195, 2018.

[19] Azure ai guide for predictive maintenance solutions. `https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/cortana-analytics-playbook-predictive-maintenance`.

[20] Reti neurali: cosa sono e a cosa servono. `https://www.ai4business.it/intelligenza-artificiale/deep-learning/reti-neurali/`.

[21] Manutenzione predittiva: Tipi di manutenzioni a confronto e vantaggi. `https://www.toolsforsmartminds.com/it/insight/blog/178-manutenzione-predittiva-tipi-di-manutenzioni-a-confronto-e-vantaggi`.

[22] Verso la vera predictive maintenance, come fondere iot e machine learning. `https://www.internet4things.it/industry-4-0/verso-la-vera-predictive-maintenance-come-fondere-iot-e-machine-learning/`.

[23] Volare in sicurezza con easyjet. `http://meccanica-plus.it/volare-in-sicurezza-con-easyjet_73246/`.

[24] Del Pico J. Garcia M. C., Sanz-Bobi M. A. Simap: intelligent system for predictive maintenance: application to the health condition monitoring of a windturbine gearbox. *Computers in Industry*, 57:522–568, 2006.

[25] He Q. Qian B. Li Z. Fang D. Hampapur A. Li H., Parikh D. Improving rail network velocity: a machine learning approach to predictive maintenance. *Transportation research part C: emerging technologies*, 45:17–26, 2014.

[26] J. A. Gaikwad S. S. Patil. Vibration analysis of electrical rotating machines using fft. *IEEE*, 31661, 2013.

[27] Nasa bearing data set. `https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/`.

[28] Mqtt. `http://mqtt.org/`.

[29] Json encoder and decoder. `https://docs.python.org/3/library/json.html`.

[30] Numpy. `https://numpy.org/`.

[31] Pandas. `https://pandas.pydata.org/`.

[32] Matplotlib. `https://matplotlib.org/`.

[33] Scikit-learn. `https://scikit-learn.org/stable/`.

[34] Keras. `https://keras.io/`.

[35] Hanjun Kim MArtin B.G. Jun John W. Sutherland Wo Jae Lee, Haiyue Wu.

Predictive maintenance of machine tool system using artificial intelligence techniques applied to machine condition data. *ScienceDirect*, 506-511, 2019.

[36] A guide for using the wavelet transform in machine learning. http://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/.

[37] Feature selection correlation and p-value. https://towardsdatascience.com/feature-selection-correlation-and-p-value-da8921bfb3cf.

[38] A feature selection tool for machine learning in python. https://towardsdatascience.com/a-feature-selection-tool-for-machine-learning-in-python-b64dd23710f0.