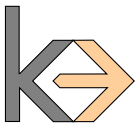
# Machine Learning Approaches for Failure Type Detection and Predictive Maintenance

**Maschinelle Lernverfahren für die Fehlertypenkennung und zur prädiktiven Wartung**
Master Thesis submitted by
Patrick Jahnke
June 19, 2015

TECHNISCHE UNIVERSITÄT DARMSTADT



Knowledge Engineering Group

Department of Computer Science

Prof. Dr. Johannes Fürnkranz
Hochschulstrasse 10
D-64289 Darmstadt, Germany

www.ke.tu-darmstadt.de

Supervisor: Prof. Dr. Johannes Fürnkranz
Advisor: Dr. Frederik Janssen, Dr. Immanuel Schweizer, Dr. Andrei Tolstikov

# Thesis Statement pursuant to §22 paragraph 7 of APB TU Darmstadt

I herewith formally declare that I have written the submitted Master Thesis independently. I did not use any outside support except for the quoted literature and other sources mentioned in the paper. I clearly marked and separately listed all of the literature and all of the other sources which I employed when producing this academic work, either literally or in content. This thesis has not been handed in or published before in the same or similar form.
In the submitted thesis the written copies and the electronic version are identical in content.

Darmstadt, June 19, 2015

_____

(Patrick Jahnke)

# Erklärung zur Abschlussarbeit gemäß §22 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Patrick Jahnke, die vorliegende Master-Thesis ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.
In der abgegebenen Thesis stimmen die schriftliche und elektronische Fassung überein.

Darmstadt, den 19. Juni 2015

_____

(Patrick Jahnke)

## Abstract

With an increasing number of embedded sensing computer systems set up in production plants, machines, cars, etc., there are new possibilities to monitor and log the data from such systems. This development makes it possible to detect anomalies and predict the failures that affect maintenance plans.
This thesis divides the field of failure type detection and predictive maintenance into subsections that focus on its realization by a machine learning technique, where each area of failure type detection and predictive maintenance explains and summarizes the most relevant research results in recent years. Each technique introduced is considered from a failure type detection and predictive maintenance perspective, highlighting its assets and drawbacks. An overview in tabular form shows the direction of the current research activities. It will be shown that many issues in the area of feature extraction, feature selection, and data labeling have not been researched so far.

## Zusammenfassung

Die zunehmende Anzahl eingebetteter Sensorsysteme, die in Produktionsanlagen, Maschinen, Autos, usw. implementiert werden, eröffnet neue Möglichkeiten, um ein solches System zu überwachen und Objektdaten zu erfassen. Diese Entwicklung ermöglicht es, Anomalien zu detektieren und Fehler, die wiederum Auswirkungen auf Wartungspläne haben, vor dem eigentlichen Auftreten zu erkennen.
Diese Arbeit teilt das Forschungsfeld der Fehlertypenkennung und prädiktiven Wartung in Teilgruppen auf und fokussiert sich auf Umsetzungen mithilfe maschineller Lernverfahren. Jeder Teilbereich der Fehlertypenkennung und prädiktiven Wartung wird erklärt und die Forschungsergebnisse der letzten Jahre in den einzelnen Bereichen zusammengefasst. Jede vorgestellte Technik wird mit deren Vor- und Nachteile für die Fehlertypenkennung und prädiktiven Wartung betrachtet. Ein tabellarischer Überblick zeigt die aktuelle Forschungsrichtung der Teilbereiche. Weiter werden Problemstellungen in den Bereichen Feature-Extraction, Feature-Selection und Data-Labeling aufgezeigt, die bislang noch nicht untersucht worden sind.

## Acknowledgment

At first, I would like to thank my thesis advisors, Prof. Johannes Fürnkranz and Dr. Frederik Janssen, who accepted my thesis proposal and made this work possible by offering their advice and support.
Furthermore, I would like to express my gratitude to Dr. Immanuel Schweizer for the considerable amount of time that he invested in answering my early questions, as well as for the interesting discussions and organizational support.
Moreover, I would like to thank Dr. Andrei Tolstikov, Dipl. Inf. Sebastian Kauschke, Dr. Axel Schulz and Simon Hofmann for their technical support and efforts.
Finally, I would like to thank my wife Kerstin and my children, Paul Maximilian and Hendrick Lennard, as well as my parents, who have always stood by me and dealt with my absence from many family occasions with a smile and never-ending support.

## Contents

## List of Figures

## List of Tables

# 1 Introduction

As awareness increases of the Internet of Things, more possibilities open up to connect physical objects with one another, making possible access to object-related data, such as sensor data and reports. This represents a huge amount of data that can be used in many ways. One possibility is predictive maintenance, where the data is used to detect upcoming failures before they occur, and schedule maintenance actions. The task of predictive maintenance is to determine when a machine must be maintained. Assessing the failure that occurs is called 'failure type detection' in this thesis. Failure type detection and predictive maintenance are related to each other. But there is also a difference between them. For example, sometimes it is not possible to detect the type of failure but an anomaly can be discerned. Thus, the task of predictive maintenance can be fulfilled, as well as there is no failure type detection possible. On the other hand, failures that occur could be detect and distinguished from one another. In this case, the failure type detection works, but by the time the failure is detected, it is too late to schedule any maintenance action.
A good technique in this area can lower the costs of damage, improve security and reduce the number of unnecessary maintenance actions. The process of such a technique consists of several areas as self-research fields. The structure of this thesis is related to this process. More types of models are briefly introduced in the following (see Chapter 3). Such a model can be very different from one implementation to another. Before the data can be analyzed, it must be determined how the data can be acquired, what kind of data is accessible, and which format it takes. This area is called data acquisition, whereby the following techniques depend upon its results (see Chapter 4).
After the data is acquired, it has to be processed. The steps from the raw data to the features are called feature engineering and involve signal processing, feature extraction, and feature selection (see Chapter 5).
Before a data-driven approach can analyze the data from a real-world system, the approach has to learn the relation between a given input and expected output. This is usually done by analyzing and labeling historical data. Data labeling techniques are discussed in Chapter 6.
There is a very wide range of machine learning techniques that learn the relation between a given input and expected output. This thesis discusses the most commonly used techniques in recent literature for failure type detection and predictive maintenance are discussed. Chapter 7 describes the difference between parametric and non-parametric approaches and every single approach with a focus on failure type detection and predictive maintenance.
To evaluate the different approaches, the common evaluation strategies used in recent literature are not satisfactory. The most likely reasons and a completely new approach are discussed in Chapter 8.
The conclusion discusses the results from the thesis, and then briefly introduces the outlook on future works (see Chapter 9).
This thesis, a survey of more than sixty papers published during the last five years, focuses on approaches to solve problems related to failure type detection and predictive

maintenance problems. Surprisingly, the approaches of feature selection, feature extraction, and data labeling are not well researched in the papers mentioned herein; these steps are very important for choosing the right machine-learning approach from an analytical point of view, and receive a good accuracy of a system. All papers reviewed choose the machine learning technique by empirical experimentation. Dependent to this there are two main parts. The first one is about signal processing, which is very well researched in the recent literature. For building up ideas for feature selection, feature extraction or data labeling techniques in the context of failure type detection and predictive maintenance, the understanding of this signal processing is essential. The second part is about machine learning, a key technology in a failure type detection and predictive maintenance scenario. In this scenario there is a variety of techniques used, which will be discussed with their pro and cons. One can derive other tendencies from the usage of a given machine learning technique, but as mentioned above, there is currently no analytical decision process available for choosing a machine learning technique, which, therefore, makes a broad overview useful and appropriate.

## 2 Related Work

In recent literature, surveys have focused on failure type detection and predictive maintenance techniques. [79] provides a short summary about the current literature. The paper does not introduce the entire process, nor goes into detail about every research area like the one covered in this thesis. Sheng Si et al. provides a review of statistical driven approaches in [84], focusing on parametric machine learning approaches. In contrast, this thesis explains the whole process with a focus on non-parametric approaches, because such techniques have been much more commonly used in research literature in recent years. Peng et al. [70] describe the process of a failure type detection and predictive maintenance approach, focusing on data-driven methodologies, whereby knowledge-based and physical model-based techniques are introduced. The other process steps are not discussed in such detail as compared to the present thesis. Jardine et al. reviewed machinery diagnostics and prognostics implementing condition-based maintenance in [48], describing data acquisition, data processing, maintenance decision support and multiple sensor data fusion. However, given that this paper is from 2006, this present thesis contains more recent information about trends and technology. Moreover, it also does not discuss the feature extraction, data labeling and evaluation strategies. Feature selection, and machine learning approaches are not discussed in such detail as in this thesis.

# 3 Prognosis and Analysis Models

In this thesis, a model describes the behavior of a real world system across its lifetime. As in real world systems, a model typically displays dynamic behavior, which is important for a prognosis of a future state, or to analyze the current state of a real world system. Thus, a prognosis model realizes predictive maintenance tasks and an analysis model is used for failure detection. As described above in Chapter 1, failure type detection and predictive maintenance are related to each other; therefore, a prognosis model is related to an analysis model.
There are many possibilities to build such models. Every prognosis and analysis system works in the same abstract manner (see Figure 3.1). The data of a real world system has to be collected before the feature engineering is able to transform the data into a representation that the model can process. Finally, after the model has processed the data, the prognosis and analysis system outputs the results.
The following subsections describe the classes into which the model approaches can be categorized.


**Figure 3.1:** Workflow of prognosis and analysis System

## 3.1 Physical Model-Based Methodology

Physical models are usually realized on the basis of mathematical models. Differential equations are used to specify physical processes that have an influence on the health of related components. Such models are created by domain experts (see Figure 3.2). They must have knowledge about the physical process of a given real world system, as well as skills in modeling such systems. After modeling a system, the resulting model has to be validated using a large set of data to verify the correct behavior. If the model is implemented correctly, monitoring systems can be used to collect the input parameters of the model. The model will produce the output value, and statistical techniques are used to define the threshold for detecting the state of the real world system with the given output.
Physical models are useful for very dynamic real world systems under many operation conditions. Creating a physical model requires a very high understanding of the real world system. Usually, the correctness of analysis and prognosis for given monitored parameters increases with the complexity of the model.
Among the disadvantages of such models are their high costs and the component specialty, which means that they cannot be applied to other types of components [14]. Moreover, it is very difficult to build a physical model with a good accuracy, because real world systems are influenced by a lot of environmental factors that are hard to monitor. Physical models are usually researched by the same disciplines that research

the real world system. For example, the physical model of electronic circuits are implemented and examined in the area of electronic engineering, and the physical model of mechanical components are implemented and examined in the area of mechanical engineering. This thesis focuses on data-driven models because they are more generic and can be implemented in different real world systems. For more information about physical models, please consider references [68], [62] and [69].

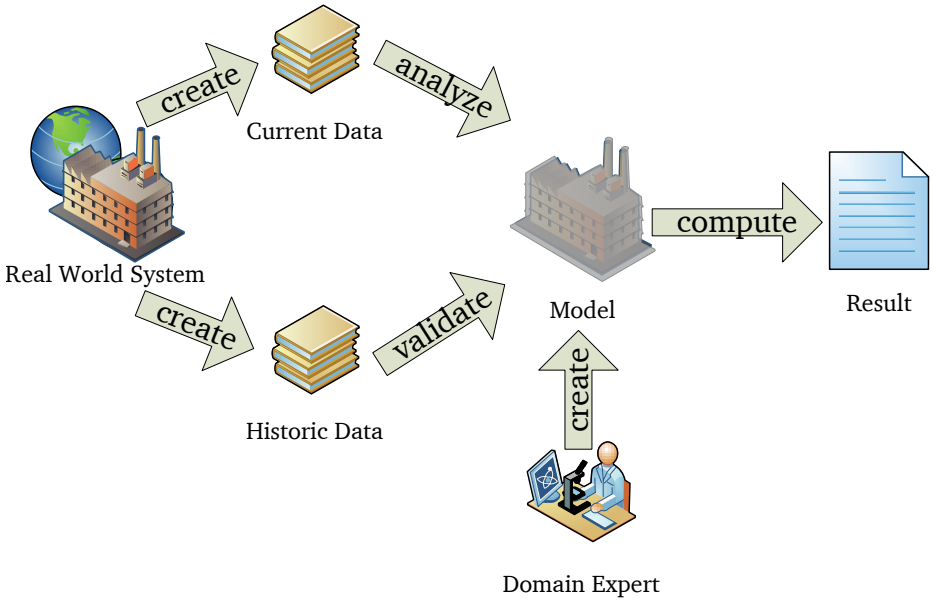



**Figure 3.2:** Workflow of a physical model

## 3.2 Knowledge-Based Models

Knowledge-based methodology, like physical model-based methodology, is created by domain experts, although there is no mathematical model to cover its physical behavior. The quality of a knowledge-based model depends on the experience of the domain expert. Such systems try to formalize the knowledge of the domain expert. In the fowling the expert models and fuzzy logic describe the most common technologies in the research area of failure type detection and predictive maintenance.

### 3.2.1 Expert models

Expert models are suitable for problems which are usually solved by specialists. They are defined by rules and describe states of a real world system. Rules are expressed in the form: IF condition, THEN consequence, ELSE consequence. Such rules can be combined with logic operators. Conditions describe a state of the real world system, with the consequence of a result, or another rule.
It is difficult to obtain domain knowledge and convert it into a rule-based system [98]. An expert model cannot handle situations that are not covered in the rule-based system. Furthermore, an expert model can run into a combinatorial explosion, which occurs

when the number of rules dramatically increases. Therefore, an expert model is not suitable as standalone approach for failure type detection and predictive maintenance in a real-world system. [121] define an expert model and refine them with a data-driven approach, which will be introduced in Section 3.3. This solution achieves results better than either, the data-driven model, or the expert model alone. Accordingly a simple expert model can be used in combination with a data-driven approach to converge faster and nearer to the optimal solution.
In [18] Butler proposed a framework, based on the expert model, for incipient failure type detection and predictive maintenance. Biagetti and Sciubba created a prognostic and intelligent monitoring expert system, which detects faults in real time and provides forecasts for detected and likely faults. Furthermore, the system gives suggestions on how to control the problem [11].

### 3.2.2 Fuzzy logic models

Fuzzy logic [114] is a simple way to describe a real-world system based on vague, imprecise, noisy or missing inputs. It is a superset of the conventional Boolean logic. The main difference between a conventional Boolean logic and fuzzy logic is the decision-making process. With the conventional Boolean logic, an element is either a member of a set or not. For a given element, the fuzzy logic returns the degree of membership to a set. With a fuzzy logic model, states can be described in a continuous and overlapping manner, like the state transitions in a real world system. Therefore, a description of a real world system in fuzzy logic is more intuitive and less specific than a numerical or mathematical description. A fuzzy logic model can create simpler, more intuitive and better behaved models [70].
As described above, fuzzy logic states can be overlapping. Every input value belongs to a fuzzy rule with a degree of membership. The range covering the degree of membership starts from 0 (definitely not a member) to 1 (definitely a member). In contrast to conventional Boolean logic, whether an input value matches a rule or not, a fuzzy logic model can consider all degrees of membership. This means that all degrees of membership can influence the result and the rule, which represents a state, whereby the highest numerical result is not automatically the state of the calculated result .
In [36] Frelicot developed a prognostic adaptive system based upon fuzzy pattern recognition principles. The fuzzy classification rule performs fault detection, including membership rejection (non-detection) and multiple detection. Choi et al. designed an alarm filtering and diagnostic system based upon an online fuzzy expert model [22].

## 3.3 Data-Driven Models

Data-driven models are based upon statistical and learning techniques. In this thesis, the learning techniques of machine learning involve developing software that optimizes a performance criterion on the basis of historical data and/or experience. With the growing number of sensors in a real-world system, the possibility to detect its behavior and the current state increases. Therefore, most approaches in recent literature to failure type detection and predictive maintenance account for data-driven models. These

models are more generic than physical and knowledge base models. Hence, this thesis focuses on data-driven models for failure type detection and predictive maintenance. Algorithms behind data-driven models are determined from the given data. There are three different learning techniques, which are described in the following sections.

### 3.3.1 Supervised learning

In supervised learning, data collected in the past from a real-world system is used. This data set is termed historic data in this thesis. As not all collected data records can be used as learning examples, the data has to be filtered. Each record of training data is labeled according to its expected result. The resulting data set is termed training data. It is processed by a machine learning technique, which is discussed in Chapter 7. This technique examines the relationship between a data record and the labeled output, and then creates a data-driven model. For any new data, the data-driven model tries to give the best result based upon the data learned. Figure 3.3 visualize the described work flow of a supervised learning data-driven model.
In the area of failure type detection and predictive maintenance, the supervised learning technique is the most commonly used learning technique, because the real world system is monitored and the historic data is available. Moreover, the historic data can relate to the real-world system state. This improves the accuracy prior to the system going online.

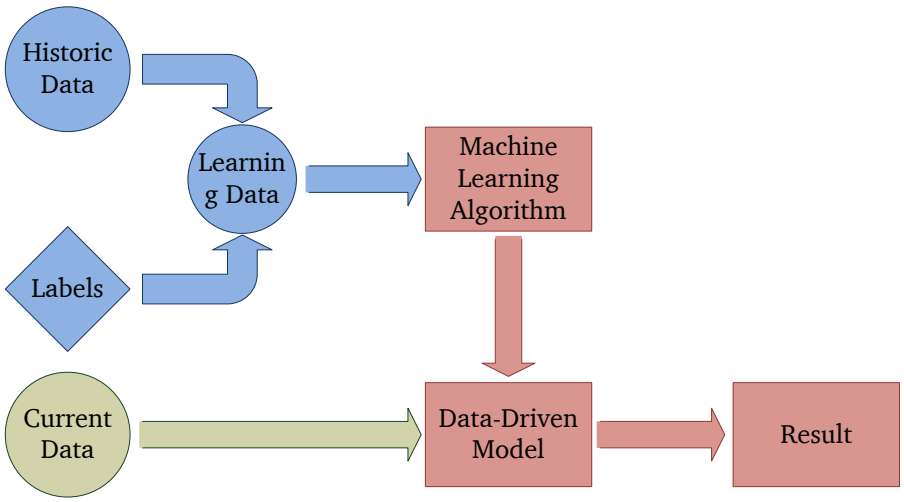



**Figure 3.3:** Supervised learning

### 3.3.2 Unsupervised learning

In other data-driven model problems, the historic data has no corresponding target values. The scope of such unsupervised learning issues is to find or discover groups of similar examples within the historic data. This process is called clustering. Another

aim of the unsupervised learning approach can be to determine the distribution of data within an input space, in a process called density estimation. Figure 3.4 shows the work flow of an unsupervised learning data-driven model.
For failure type detection and predictive maintenance problems, the unsupervised learning technique is an unusual learning technique, because the clustering and density estimation of the historic data is not efficient for precise failure type detection and predictive maintenance. In the recent literature of failure type detection and predictive maintenance, there is no approach uses the unsupervised learning technique, so it is mentioned in this thesis as a matter of completeness.

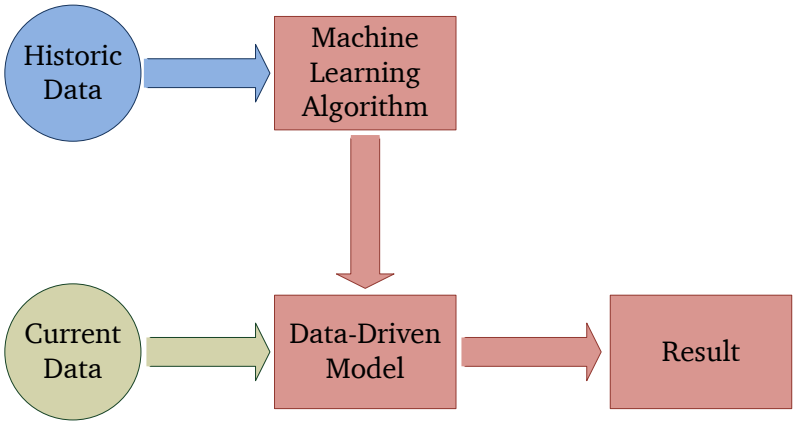



**Figure 3.4:** Unsupervised learning

### 3.3.3 Reinforcement learning

The reinforcement learning technique of [88] has tried to find a suitable result for the given input data. Unlike the other learning techniques, it has no learning phase for historic data. Thus, the machine learning algorithm is based on a trial and error scheme.
At the beginning of the learning phase, a machine learning algorithm, in respect to some aspects, defines a data-driven model. Consequently, the aspects of the machine-learning algorithm that define the data-driven model must also be defined. This model creates a result, depending on the current data in the real-world system. That result will be assessed if it matches the expected result. The machine-learning algorithm received this assessment as feedback about the quality of the result, from which the machine learning algorithm propagates more successful strategies and rejects unsuccessful ones. Figure 3.5 visualize the reinforcement learning technique.
However, the supervised or unsupervised learning technique can be used in combination, whereby the machine learning approach first learns from historic data (labeled or not). The reinforcement learning is widely used in data-driven models, where the result is an action. For example, by using the reinforcement learning technique, a machine learning technique (see Section 7.2.4) can learn to play Backgammon in expert mode [93]. There are two instances of the machine learning technique playing against each

other. In this particular example, the current data is the board position. The result is the outcome at the end of the game (win or lose). With this result, the machine learning algorithm proliferates more strategies where it won a game and rejects strategies where it loses.
In recent literature of failure type detection and predictive maintenance, there is no approach in which the reinforcement learning technique is used, because the learning process happen under runtime condition. This implies that a wrong estimate regarding the state of a real-world system could have disastrous consequences. But as mentioned above, the supervised learning technique in combination can refine a data-driven model in runtime mode, which is impossible using supervised learning alone.

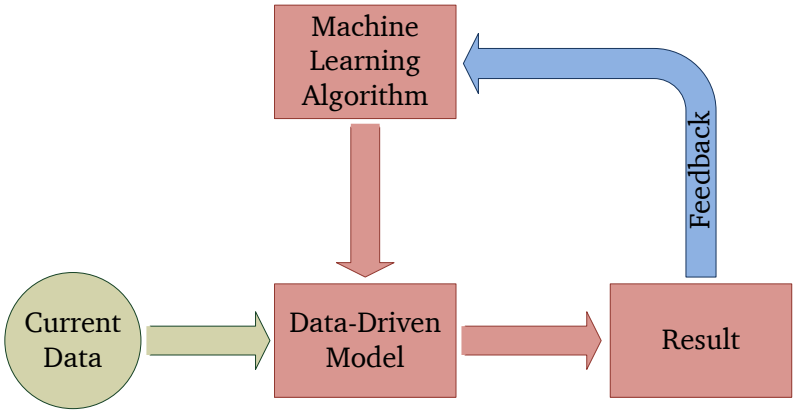



**Figure 3.5:** Reinforcement learning

## 3.4 Combinations

As mentioned above, there are three different types of models. The physical model, the knowledge-based model and the data-driven model are clearly distinct from one another. Nonetheless, there are possibilities to gain advantage by combining the different types of models.
There are some implementations that use combinations of a data-driven model with the supervised learning technique and an expert model. These models try to use the domain knowledge of an expert model with the collected real-world information of data-driven models to achieve better results. Such a model will be discussed in Chapter 7.

## 3.5 Summary

This chapter provided an overview and classification of different types of failure type detection and predictive maintenance models. The advantages and drawbacks of each type of model were briefly introduced. As established in Chapter 1, this thesis focused on the data-driven models. Therefore, the different learning strategies of data-driven

models that are relevant to the area of failure type detection and predictive maintenance are introduced. A decision tree can help in finding and selecting the right model for a given failure type detection and predictive maintenance issue (see Figure 3.6).

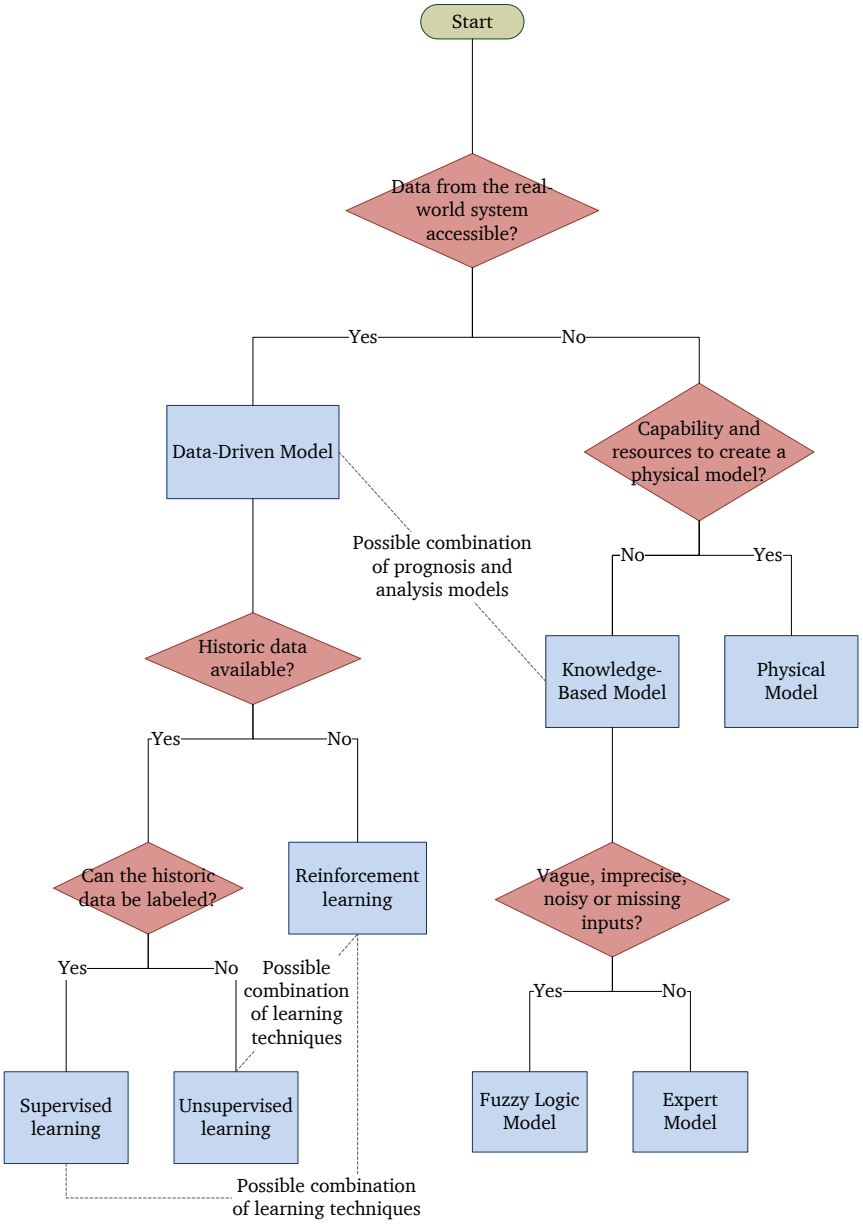



**Figure 3.6:** Decision tree to visualize the selection process of a prognosis and analysis model for failure type detection and predictive maintenance.

# 4 Data Acquisition

Every failure type detection and predictive maintenance approach begins with the data acquisition part. Therefore, the decision of the data acquisition technique influences the steps of the work flow displayed in Figure 3.1 as follows. In most cases, a failure type detection and predictive maintenance approach is implemented along with an existing data acquisition technique. The focus of an existing data acquisition technique does not necessarily correspond to the major aim of a failure type detection and predictive maintenance approach. For example, the primary objective of the sensor data acquisition task is to attain energy efficiency and reduce communication costs [2]. If such a system exists, the failure type detection and predictive maintenance approach try to optimize the outcome of the given data acquisition technique.
Data acquisition techniques collect information about a real-world system, using different sources. The most important data types of failure type detection and predictive maintenance are described in the next subsections, followed by a subsection about different data acquisition techniques.

## 4.1 Types of Data

Data acquisition systems, as well as the signal processing described in Section 5.1, needs to know the type of the data the are dealing with. The two most important types of data for failure type detection and predictive maintenance are data gathered by sensors and data gathered by logfiles:

### 4.1.1 Sensor data

In recent years, sensors have become smarter, smaller, easier to implement in existing systems, as well as cheaper and more reliable [99]. A sensor converts physical values into electrical values (voltage, current or resistance). Usually, one sensor measures one mechanical value; for example, the mechanical values of acceleration, pressure, flow, torque and force. With this mechanical value, one can interpret the vibration data, acoustic data, temperature, humidity, weather, altitude, etc. Table 4.1 shows a summary of the components and the mechanical values usually acquired. As described in [48], this data falls into three categories:

- Value type:
  Data collected at a specific time epoch for a condition, whereby monitoring variables are of a single value.
- Waveform type:
  Data collected at a specific time epoch for a condition, whereby the monitoring variables are of a time series, which is often called time waveform. For example, vibration data and acoustic data are of the waveform type.
- Multidimension type:
  Data collected at a specific time epoch for a condition, whereby the monitoring variables are multidimensional. The most common multidimensional data is image data, such as infrared thermographs, X-ray images, visual images, etc.

A good description about sensor data in a data acquisition system can be found in [7].

| Equipment | Vibration | Humidity | Ambient Temperature | Ambient Pressure | Acoustic Signal | Thermography | Motor Current | Insulation Resistance | Electrical Capacitance | Electrical Inductance |
|---|---|---|---|---|---|---|---|---|---|---|
| Pump | X | | X | X | X | X | X | X | | |
| Valve | | X | | X | X | | | | | |
| Motor / Fan | X | | X | | X | X | X | X | | X |
| Heat Exchangers | X | X | X | X | | | | | | |
| Steam Turbine | X | X | X | X | X | | | | | |
| Electrical & Electronic Equipment | | | X | | | X | | X | X | X |
| Cables and Connectors | | | X | | | X | | X | X | X |
| Pump Seal | | X | | X | X | | | X | | |
| Piping / Structures | X | | | | X | | | | | |
| Compressor | X | | | | X | X | X | | | |

**Table 4.1:** Real world system components and their measurable mechanic value [39]

### 4.1.2 Logfiles

Events of a system can be recorded to a logfile. In this manner, the declaration of an event is very broad. A changing value can be such an event, as it executes a maintenance action. Logfiles can only contain fused sensor values. Descriptive log files can be written by humans describing maintenance actions, or any failures or errors detected.

## 4.2 Data Acquisition Techniques

In failure type detection and predictive maintenance, there are two major data acquisition techniques, either push- or pull-based data acquisition techniques. In a pull-based data acquisition system, the user defines a query and sends it to the system which will then return the matched value. The frequency the query has to be executed can be determined in the query itself, or by submitting the query in the desired frequency.
In push-based data acquisition systems, a former described state or a behavior will be communicated by system autonomously whenever values recently changed. Figure 3.1 shows the push- and pull-based method.

The following subsections discuss examples of the most commonly used push- (see Section 4.2.3) and pull-based (see Section 4.2.1 and Section 4.2.2) methods for data acquisition systems of failure type detection and predictive maintenance.

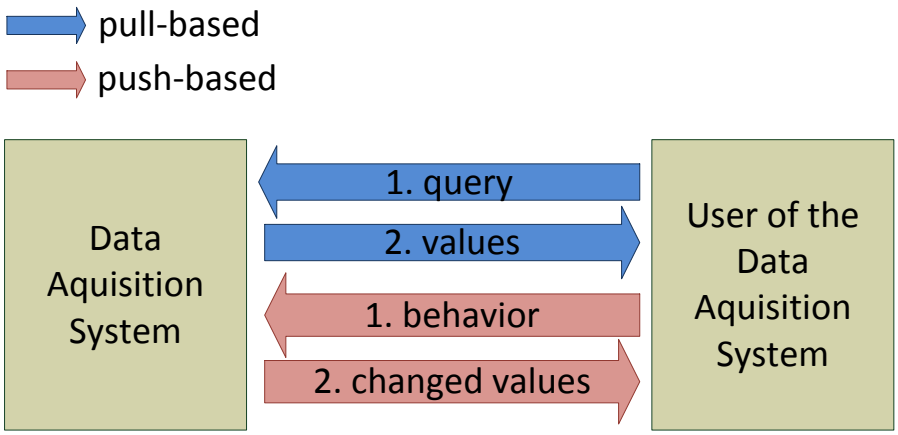



**Figure 4.1:** Push- and pull-based data aquisition techique

### 4.2.1 Real time

The real-time data acquisition technique is pull-based. Essentially, there is no system that can process data without any delay, although in terms of failure type detection and predictive maintenance, real-time means that there is no perceivable delay. A parallelization of the data acquisition is often possible solution when it has to be conducted in real-time, and the data acquisition technique represents a bottleneck in the failure type detection and predictive maintenance workflow. However, this decision of parallel data acquisition can influence the remaining failure type detection and predictive maintenance workflow steps (see Figure 3.1).
The fastest way is usually the detection of state changes in a real world system with real-time data acquisition systems, where significant data has to be transferred. As discussed in Chapter 4, the costs of energy efficiency and communication are high in real-time systems. Moreover, a real-time system may transfer a considerable amount of useless information to a failure type detection and predictive maintenance system.

### 4.2.2 Interval

In some cases, it is not possible or necessary to use a real-time data acquisition technique. In contrast to a real-time data acquisition system, an interval data acquisition system sends the acquired data in time intervals. An interval data acquisition system represents a pull-based system. The time interval for transferring the data to the user can be of different lengths, especially when the transfer is constrained by thresholds of values.

This type of system is used when communication costs are high. In some cases, it is necessary to have synchronized clocks with the communication partner. They have to determine a interface for how and what data has to be exchanged. Such interfaces can differ, from being very abstract (i.e., all values will be transferred) to very special (transfer a value only when a constraint is fulfilled). The more abstract the interface is specified, the more users can use the interval data acquisition technique with different requirements.

### 4.2.3 Event driven

The event driven data acquisition technique is push-based. Such systems are also called pub/sub systems [34], because the user of such systems subscribes to a publisher of events. In failure type detection and predictive maintenance, a subscription is a definition of behavior of the real-world system that the user wants to observe. The data acquisition system publishes the value changes detected (called Event) and sends them to all users registered to this value or the properties of the value change.
In terms of failure type detection and predictive maintenance, an event in an event-driven data acquisition can have many interpretations. An event can represent a value change in a critical way, or it can be a combination of value changes with or without any special order. Usually, in failure type detection and predictive maintenance, an event does not represent a state change of the real-world system, although it can lead to a state change.
Event-driven data acquisition systems has had a growing significance in the recent years [8]. In many distributed systems, it is a good opportunity to design the communication among anonymous parties. In some cases, an event-driven data acquisition can increase the memory consumption and processing time of the system, because a complex event depending on several parameters may need to store values for a long time and may have many rules to check [15].

## 4.3 Summary

This section presents the different approaches to acquire data and describes the most common types of data in a real-world system. Usually, most of this systems use an existing data acquisition system with a failure type detection and predictive maintenance approach.
An event-driven system is a specific system for a given failure type detection and predictive maintenance task. A well-designed event-based system can reduce the volume of data, and thus the computational costs. But the effort for designing and implementing such a system is high. In addition, expert knowledge is required to determine when and what kind of event will occur.
The simplest and most generic data acquisition system is the real-time approach. All available data are transferred in real time, where the subsequent processes selects the relevant data. Such a data acquisition technique increases the communication and computational costs of the failure type detection and predictive maintenance technique.
Between the event-based data acquisition technique and the real-time data acquisition

technique, which represent the two extremes when considering the costs and benefits, the interval data acquisition system is a compromise. On the one hand, an interval data acquisition system can be simply designed with a fix interval, where the computational and the communication costs depend on the length of the interval. A long interval has the drawback that critical issues may be detected too late, whereas a short interval increases the computational and communication costs. A dynamic interval could solve the described problem as such, but will then increase the implementation costs, as well as requiring domain knowledge.
The next section discusses how a data-driven system preprocesses the acquired data and learns from it.

## 5 Feature Engineering

The observing of a property by machine learning is called a feature. Usually more than one feature is observed at a time and the more independent these features are from one another, the greater the effort for the machine learning approach. The set of features for a machine learning approach is called the feature vector.
For most failure type detection and predictive maintenance applications, the values obtained from the data acquisition system must typically be preprocessed before transforming them into a new space of variables for better performance of the machine learning algorithm [117]. These preprocessed values from the data acquisition system are an example of a feature vector. In this paper, the preprocessing stage is called feature engineering. Feature engineering is not a formally defined term; rather, it is much more a series of tasks related to designing feature sets for machine learning applications. The most important processes of the feature engineering for failure type detection and predictive maintenance are as follows:

- Signal Processing
  The interpretation, generation and transformation of raw unprocessed data.

- Feature Selection
  Selects a subset of the most representative features.

- Feature Extraction
  Generate new information by combining features.

Figure 5.1 shows the work flow of feature engineering within these components. The following sections describe the main components of the feature engineering process from a failure type detection and predictive maintenance perspective.

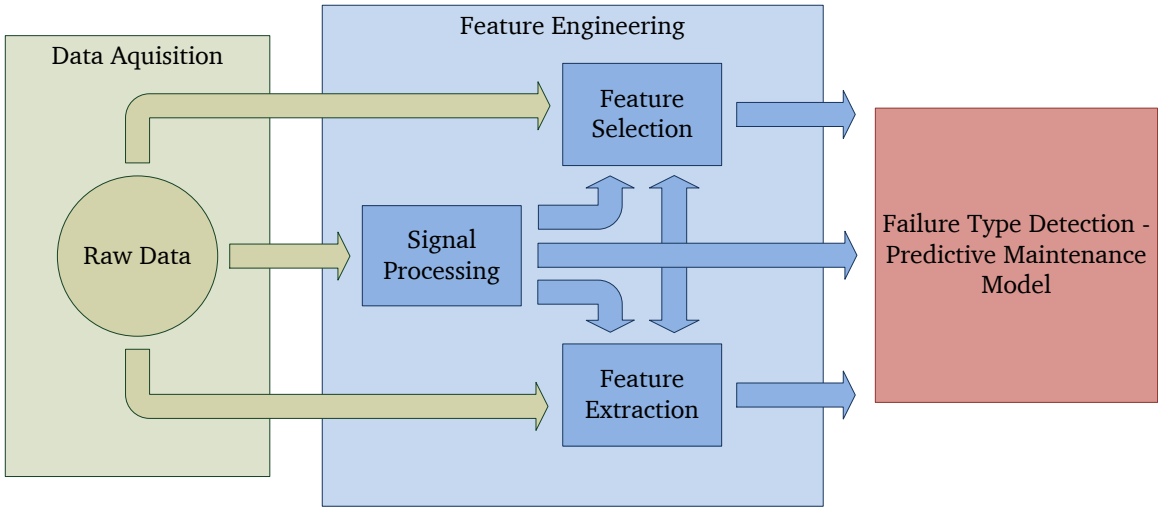



**Figure 5.1:** The feature engineering process

## 5.1 Signal Processing

The interpretation, generation and transformation of raw unprocessed data is called signal processing. As mentioned in Section 4.1.1, the data, especially the sensor data, is versatile. Signal processing uses mathematical, statistical, computational, heuristic and/or linguistic representations, formalisms, modeling techniques and algorithms for generating, transforming, transmitting, and learning from analog or digital signals. It consists of the theory, algorithms, architecture, implementation, and applications related to processing information, contained in a variety of different formats, broadly designated as signals. Signal processing may involve filtering, recovery, enhancement, translation, detection, and decomposition [101]. Moreover, [91] mentions many useful techniques applied for signal analysis. This thesis will focus on waveform signal processing because many failure type detection and predictive maintenance approaches discussed in Chapter 7 are based upon waveform data. These techniques can be classified into three types: time domain, frequency domain and time-frequency domain.
In recent research of failure type detection and predictive maintenance, sensor signals that have time and frequency components (i.e., vibration signal, acoustic signal) to analyze are the most common type of signal for detecting the state of a real world system. Therefore, the time-frequency analysis (see Section 5.1.3) is the most common technique. For a better understanding of the time-frequency analysis, the time-domain (see Section 5.1.1) and the frequency domain analysis (see Section 5.1.2) with the popular Fast Fourier Transformation is introduced. At the end of the signal processing section, Table 5.1 provides an overview of the recent research literature of failure type detection and predictive maintenance that mention the signal processing part.

### 5.1.1 Time domain analysis

Time domain analysis is an analysis of a waveform. From a mathematical perspective, the waveform of a signal is a chronological sequence of the value of a random variable. An example waveform (blue line) is visualized in Figure 5.2. A random variable has expected value, variance, skewness and kurtosis. The value of these terms are typically results of a time domain analysis for a given random variable, and can increase the expressiveness of it.
Time series modeling is an extension of the time-domain analysis [23]. The main idea is to fit the waveform data to a parametric time series model. The autoregressive model is a common time series model, which specifies that the output variable depends linearly on its own previous values [75]. Another time series model is the fractal time series [59], which detects the dependency of two waveforms, for long-range dependence or short-range dependence, and global or local self-similarity.
As mentioned in Section 5.1, vibration signals are very common among signals for failure type detection and predictive maintenance approaches. As with all sensor signals, noise is a component of vibration signals. Such components can be environmental conditions or the imprecision of the sensor itself. Noise can distort the result. A special time-domain analysis technique for vibration data is the time synchronous average technique [10] can be used, as it tries to denoise a vibration signal by analyzing exam-

ples from the real world system. To denoise a given signal, it need to be filtered. This technique has a large computational cost when filtering the signal in the time domain. Such filtering operations are simple multiplications in the frequency domain, which will be discussed below, and therefore the computational effort is lower than in the time domain.

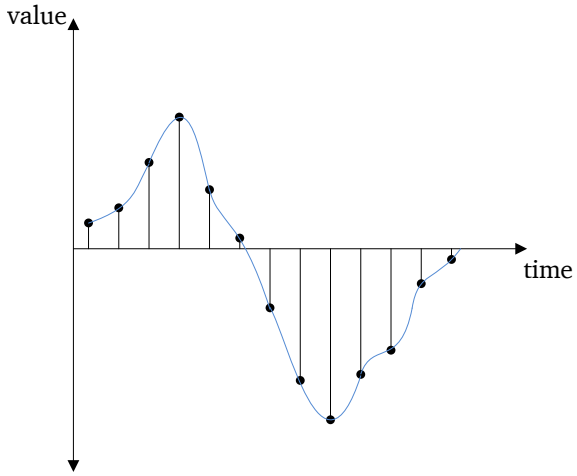



**Figure 5.2:** An example of a waveform (blue line) for a given random variable.

### 5.1.2 Frequency domain analysis

The frequency domain analyzes a given signal with respect to the frequency. It is not relevant to know when a signal has a particular frequency; the fraction that a particular frequency has in the signal is relevant. A signal in the time-domain can be transformed into the frequency domain and back again. Such a transformation has two reasons. It is important to know the distribution of the frequency share, and the signal needs to be filtered. As mentioned in Section 5.1.1, sensor signal have noise components. These noise components need to be filtered out of the sensor signal. Filtering is usually performed in the frequency domain, because filtering causes convolution in the time domain and multiplication in the frequency domain. Therefore, it is considerably more efficient to transfer a signal from the time domain to the frequency domain, perform the filtering (multiplication), and transform the filtered signal back to the time domain. Signals from a data acquisition system are mostly available in the time domain. To transfer such signals into the frequency domain, the techniques [35] most commonly used are as follows

- Fourier Transformation
- Laplace Transformation
- Z Transformation

The time and the frequency components are relevant in recent research on the failure type detection and predictive maintenance approaches. Accordingly, the frequency domain analysis is almost irrelevant. Only the Fast Fourier Transformation is infrequently used and will be discussed in the following section.

#### 5.1.2.1 Fast Fourier transformation

The signals from a data acquisition system are time discrete and finite, and therefore, the Fourier transformation for such signals is called discrete Fourier transformation [104]. Fast Fourier Transformation (FFT) is an algorithm to efficiently compute a discrete Fourier Transformation [32]. There are some different algorithms for the Fast Fourier Transformation, and all of them work with interim results and reduce the number of arithmetic operations. The complexity of a Fast Fourier Transformation is represented by $\mathcal{O}(n \log n)$. The Fast Fourier Transformation of a discrete, finite signal is explained in [105]. From the results of a fast Fourier transformation some features can be extracted; for example, the rate of a certain frequency or the most common frequencies (peaks) of a signal. On the left of Figure 5.3 is shown a discrete, finite signal in the time domain with two different frequencies and different amplitudes, on the right the FFT of that signal.
The FFT is applied to all values of the signal. Therefore, the frequencies with a very small percentage of the overall signal can be excluded. In many failure type detection and predictive maintenance approaches, an upcoming failure is detected with a changing frequency, but as long as the percentage of the new frequency does not significantly increase in relation to the overall signal, this could be also interpreted as noise. This is main drawback of FFT in failure type detection and predictive maintenance.

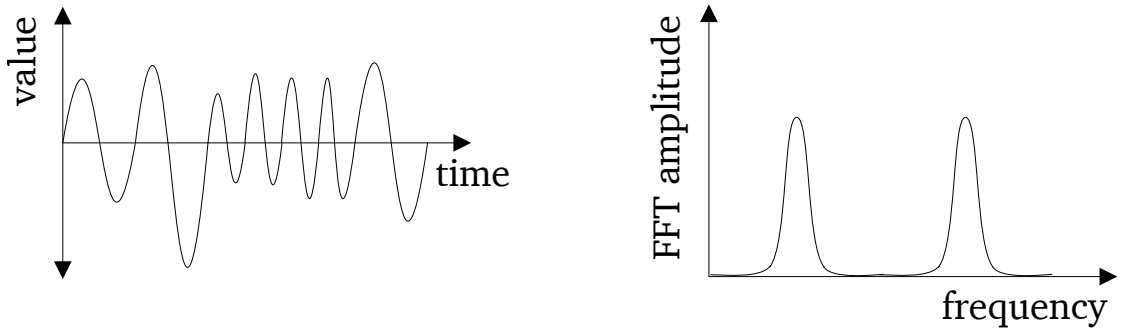



**Figure 5.3:** An example of a signal in time domain (left) and frequency domain after a FFT (right).

### 5.1.3 Time frequency domain

A failure type detection and predictive maintenance approach analyzes a given signal and tries to determine the state of the real world system. The detection of significant signal changes is a time-based problem. In many cases, the most distinguished information is hidden in the frequency content of signals, so signal processing techniques

in a time and frequency domain is widely used in literature. The following sections describe the most common time-frequency domain techniques for failure type detection and predictive maintenance.

#### 5.1.3.1 Short-time Fourier transformation

The Short Time Fourier Transform (STFT) belongs to the Fourier-related transformation techniques introduced in Section 5.1.2.1. In contrast to the fast Fourier transformation, the STFT analyzes a signal for a defined time window only. The STFT solves the drawback of the Fourier transformation as it evaluates the sinusoidal frequency and phase content of local sections of a signal as it changes over time. The resolution of a STFT window function is fixed, which implies that the width of the window is a trade off between the time and the frequency resolution. A wide window has a good resolution in the frequency domain, but a poor resolution in the time domain, whereas a narrow window has a poor resolution in the frequency domain and a good resolution in the time domain. The two different types of resolutions are depicted in Figure 5.4. Usually, the complexity of the STFT is $\mathcal{O}(n \log n)$, but there exists some modification of the STFT algorithm that can achieve the complexity of $\mathcal{O}(n)$ [26].

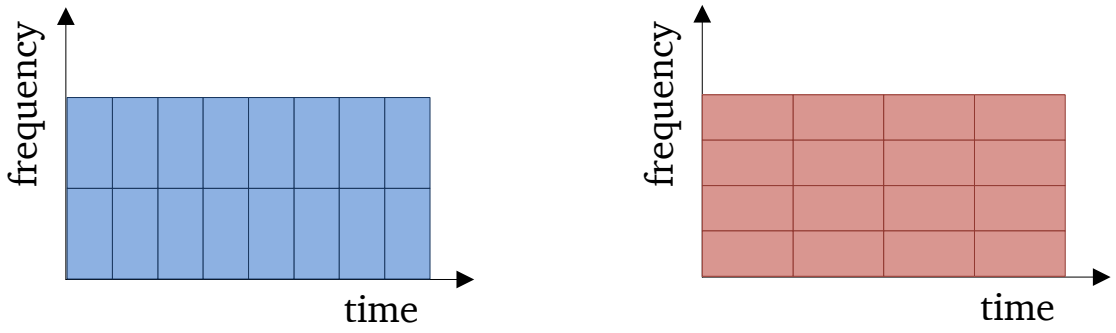



**Figure 5.4:** An example of the STFT the rectangle represents the window function, which either has a good resolution in the time domain, as well as a poor resolution in the frequency domain (left), or a good resolution in the frequency domain and a poor resolution in the time domain (right).

#### 5.1.3.2 Wavelet transformation

As described before, transforming signals into the time-frequency domain has its weakness in the resolution. A fix resolution is always a tradeoff between good time or good frequency resolution (see Section 5.1.3.1). The wavelet transformation (WT) has a (time) shifted version of the original signal, and it also has a scaled version. This implies that with the wavelet transformation, it is possible to get a poor frequency and good time resolution at high frequencies and good frequency and poor time resolution at low frequencies. This temporal resolution is a key advantage in using the Fourier transformation. There are two categories of wavelet transformation:

- Continuous Wavelet Transformation
- Discrete Wavelet Transformation

The continuous wavelet transformation can change the scale continuously in the window at every iteration, which implies that the resolution of frequency and the time can change at every iteration to fit the signal as well as possible. The following equation represents the continuous wavelet transformation:

$$CWT(a,b) = \int_{-\alpha}^{\alpha} x(t)\psi^*_{(a,b)}(t)dt \tag{1}$$

$x(t)$ represents the time function (signal), where $\psi^*_{(a,b)}(t)$ is a continuous function with complex conjugate in the time and frequency domain. $\psi^*_{(a,b)}(t)$ is called the mother wavelet and can be expressed as:

$$\psi^*_{(a,b)}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-a}{b}\right) \text{ where } a,b \in \mathbb{R}, a \neq 0 \tag{2}$$

The mother wavelet function is used to generate daughter wavelets, which are a translated and scaled down version of the mother wavelet. In Equation (1), there are two parameters, $a$ and $b$, which adjust the frequency and the time location of the wavelet. A small $a$ produces a wavelet with a high frequency resolution and a low $a$ produces a wavelet with a low frequency resolution. $b$ defines the length of the signal in the current iteration.
For computation, it is more efficient to discretize the continuous wavelet transformation, which is called discrete wavelet transformation. The most common discretization is the dyadic method:

$$DT(a,b) = \int_{-\alpha}^{\alpha} x(t)\psi^*_{(j,k)}(t)dt \tag{3}$$

$$\psi^*_{(a,b)}(t) = \frac{1}{\sqrt{2^j}}\psi\left(\frac{t-2^j k}{2^j}\right) \tag{4}$$

[29] and [63] replaced $a$ and $b$ with $2^j$ and $2^j k$, which is the difference between equation Equation (2) and Equation (4). This implies that the signal $x(t)$ passes through two complementary filters.
It is possible to transform the same length of a signal more than once with a DWT. This process is called wavelet packed transformation. The wavelet packet transformation creates a wavelet tree (see Figure 5.5) of wavelet transformations. The process starts with the original signal (displayed in Figure 5.5 as $C_{0,0}$). On every hierarchy level, the available filter banks in the upper level split the signal with a low-pass and high-pass filter (displayed in Figure 5.5 as $g_{LP}$ and $g_{HP}$). This implies that on every deeper hierarchy level, the frequency band is divided by two (displayed in Figure 5.5 as $\downarrow 2$). A

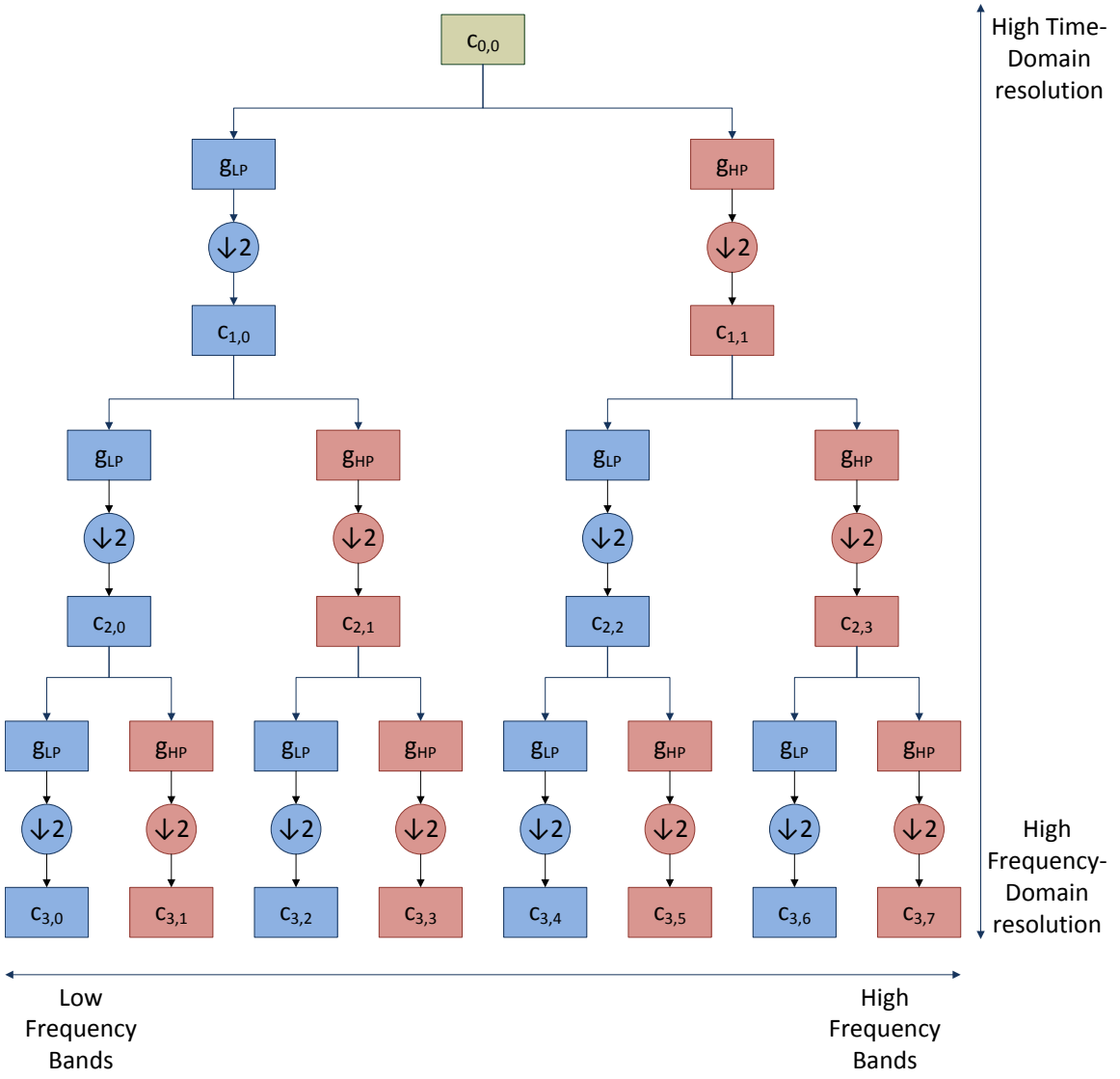



**Figure 5.5:** A wavelet-packet-tree with binary distribution. $C_{0,0}$ represents the original signal (mother wavelet). The red elements represent a high-pass filter and the blue elements represent the low-pass filter [55].

transformed part of the root signal is the result, and is displayed in Figure 5.5 as c with two indices. The first defines the hierarchy depth. The second index starts from the lowest frequency, and ends at the highest transformed frequency. Therefore, a wavelet packet tree implies that the deeper the level of the wavelet packet tree, the higher the frequency resolution and the lower the time resolution. The binary distribution on the frequency domain and the time domain is shown in Figure 5.6. This image illustrates how the time domain and frequency domain changes from the hierarchy level to hierarchy level for the same root signal (mother wavelet).
In failure type detection and predictive maintenance, a signal mostly needs a high time domain resolution and low frequency resolution for high frequencies, and a low time domain resolution and high frequency resolution for low frequencies of the given signals, because failures are often represented as a combination of short recurring frequency components in a signal. To ascertain such properties, it is useful to create a wavelet packet tree. Some combinations of wavelets with different resolutions can be significant for a failure type detection and predictive maintenance. Figure 5.7 visualizes an example for a combination of a wavelet packet tree with different transformation levels and the related scaled window of the time domain and frequency domain.

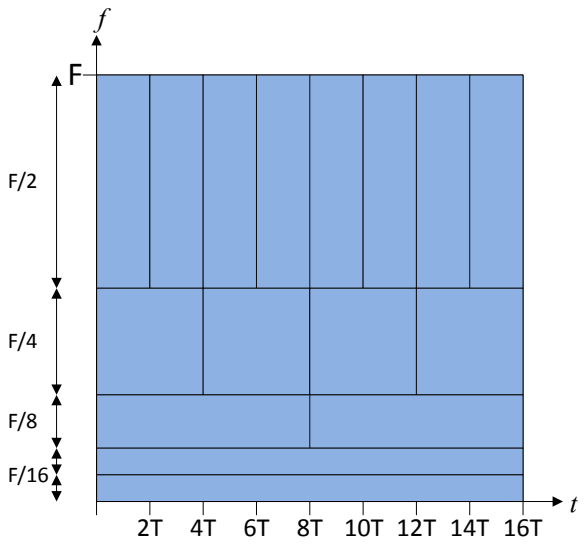



**Figure 5.6:** The change of the time domain resolution and frequency domain resolution with the equations Equation (3) and Equation (4).

#### 5.1.3.3 Hilbert Huang transformation

The Hilbert Huang transformation (HHT) is a recent approach in signal processing and was proposed by Huang [45]. The approach analyzes non-linear and non-stationary signals. Therefore, an implementation into a failure type detection and predictive maintenance approach is possible.

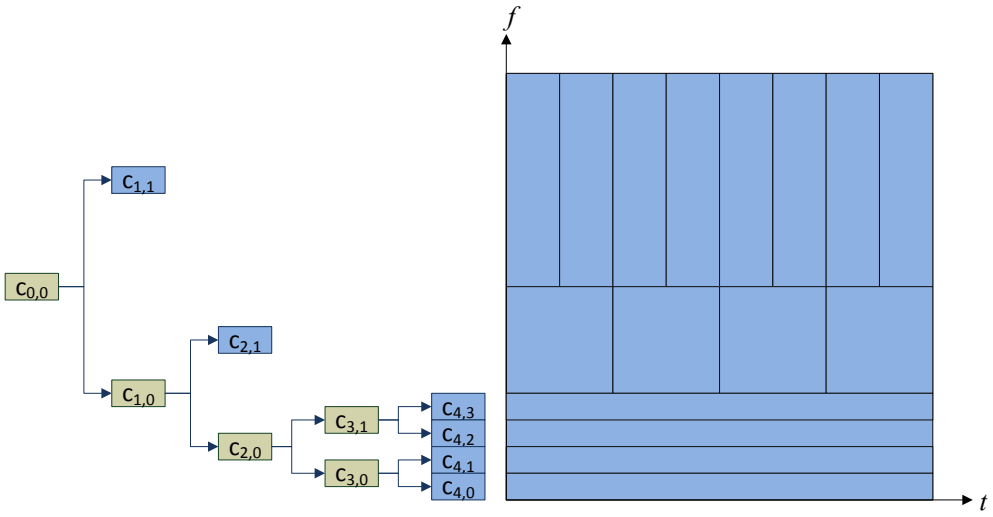



**Figure 5.7:** Example for the distribution of a mother wavelet in the time domain and frequency domain. The wavelets $C_{1,1}$, $C_{2,1}$, $C_{4,0}$, $C_{4,1}$, $C_{4,2}$ and $C_{4,3}$ are used to analyze the mother wavelet $C_{0,0}$

The first step of the Hilber Huang transformation is the decomposition of a non-linear non-stationary signal into a finite number of intrinsic mode functions. In the Hilbert Huang transformation, this process is named empirical mode decomposition and can expressed as [109]:

$$x(t) = \sum_{k=1}^{m} c_k(t) + r_m(t) \tag{5}$$

$x(t)$ is the original signal at time $t$ and $c_k(t)$ is the $k$-th intrinsic mode function. $r_m(t)$ is residue, which can be a constant (mostly in stationary signals) or the mean trend of the signal. This implies the assumption that any signal can be interpreted as a composition of intrinsic mode functions and a residue, and the original signal at time $t$ can compute by the frequency values of the intrinsic mode functions and the value of the residue at time $t$. The intrinsic mode functions are mono-components of oscillation within a certain frequency band, and fulfill the following conditions:

1. The number of extreme (minima and maxima) and the number of zero-crossing must be equal to or differ by one in the whole dataset.
2. At any point, the mean value between the local minima and the local maxima is zero.

In the empirical mode, decomposition can relate to undesirable problems, which are mode mixing and the end-effect problem. In mode mixing, disparate frequencies exist in a single intrinsic mode function, which is mostly caused by signal intermittency.

Orthogonality is a fundamental characteristic of intrinsic mode functions, although the mode mixing functions phenomenon contradicts the important property. The end-effect problem, which is the second problem of the empirical mode decomposition, describes a drift in the length of the signal.
The ensemble empirical mode decomposition was proposed to resolve the problems discussed above. This method injects white noise, which is a random signal with a constant power spectral density, into the original measured signal, which can be expressed as:

$$x_j(t) = x(t) + w_j(t) \tag{6}$$

Now, the $j$th white noise is added to the original signal $x(t)$. The result $x_j(t)$ has to be decomposed in the same way as described in Equation (5), with the difference that the intrinsic mode function and the residual, as a result of adding the white noise, gain a new dimension :

$$x_j(t) = \sum_{k=1}^{m} g_{jk}(t) + r_{jm}(t) \tag{7}$$

The steps Equation (6) and Equation (7) are repeated many times, and the final result is computed by means of the corresponding intrinsic mode function and residual:

$$h_k(t) = \frac{1}{N_e} \sum_{j=1}^{N_e} g_{jk}(t) \text{ where } k = 1, 2, ..., m \tag{8}$$

$$r_m(t) = \frac{1}{N_e} \sum_{j=1}^{N_e} r_{jm}(t) \tag{9}$$

$m$ is the number of intrinsic mode functions, and $N_e$ is the ensemble number of the empirical mode decomposition process ensemble.
To derive the minimal ensemble number, and a low signal-to-noise ratio that does not significantly affect the final result of the intrinsic mode functions, a statistical rule was established to inject the white noise:

$$N_e = \frac{\epsilon^2}{\epsilon_n^2} \tag{10}$$

$\epsilon$ is the amplitude of the white noise, and $\epsilon_n$ is the standard deviation of the error, which is defined as the difference between the sum of the intrinsic mode functions plus the trend and original signal.
The result of the embedded empirical mode decomposition do not necessarily involve intrinsic mode functions. Therefore, a postprocessing of the embedded empirical mode decomposition is unavoidable. The steps of the postprocessing process are:

1. The first component of the embedded empirical mode decomposition $h_1(t)$ is re-decomposed by the empirical mode decomposition in Equation (5): $h_1(t) \rightarrow c_1(t) + q_1(t)$. $c_1(t)$ dominates $h_1(t)$; therefore, $c_1(t)$ is used as the first intrinsic mode function.

2. For all $h_k(t)$ where $k = 2, 3, ...m$, the residual from the previous step is added to get a new sequential signal $D_k(t) = h_k(t) + q_{k-1}(t)$. This signal is processed by the empirical mode decomposition in Equation (5): $D_k(t) \rightarrow c_k(t) + q_k(t)$ and $c_k(t)$ is the $k$th intrinsic mode function.

3. Finally, the residual is calculated with $r_m(t) = q_{m-1}(t) + q_m(t)$.

The second part of the Hilbert Huang transform is a Hilbert transform of the intrinsic mode functions [81]. As discussed, the Hilbert Huang transform distributes the original signal in components called intrinsic mode functions, which are orthogonal. This implies that an intrinsic mode function represents a physical meaning, and thus practical for failure type detection and predictive maintenance [109].
The evolution from the empirical mode decomposition to embedded empirical mode decomposition is described in this thesis, because it is a common method in signal processing to add some random white noise into a given signal to achieve a more stable result, i.e. for detecting outliers.

#### 5.1.3.4 Wigner Ville distribution

The Wigner Ville distribution (WVD) is one of the most studied methods in time-frequency analysis. It was first mentioned in the area of quantum mechanics by Wigner and later extended by Ville to signal processing [24]. In contrast to the short time Fourier transformation or the wavelet transformation, the Wigner Ville distribution has no window function. Therefore, it has no leak effects and the best spectral resolution of all time frequency methods.
The Wigner Ville distribution is a quadratic integral transformation, meaning that it is a two-dimensional Fourier transformation of the ambiguity function in relation to time and frequency. The ambiguity function $A_{xx}(\tau, \upsilon)$ is a time frequency autocorrelation and has two properties:

- Time shift
  A time shifted signal lead to a modulation of the ambiguity function regarding the frequency shift $\upsilon$.

- Frequency shift
  A frequency shifted signal lead to a modulation of the ambiguity function regarding the time shift $\tau$.

This corresponds to a time and frequency energy density spectrum [52] and can be expressed as:

$$WVD_{xx}(\tau, f) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} A_{xx}(\tau, \upsilon) e^{-j2\pi\upsilon t} d\upsilon \; e^{-j2\pi f \tau} d\tau \tag{11}$$

The inner integral $\int_{-\infty}^{\infty} A_{xx}(\tau, \upsilon) e^{-j2\pi\upsilon t}$ defines the Fourier transformation of the ambiguity function in relation to the frequency. [55] shows that the inner integral can be

expressed as: $x(t+(\tau/2))x^*(t-(\tau/2))$ where $x(t+(\tau/2))$ is the original signal and $x^*(t-(\tau/2))$ is complex conjugate for it. Therefore, the Wigner-Ville Distribution can be expressed as the Fourier transformed of the time shift $\tau$:

$$WVD_{xx}(\tau,f) = \int_{-\infty}^{\infty} x(t+(\tau/2))x^*(t-(\tau/2))e^{-j2\pi f\tau}d\tau \tag{12}$$

With the Wigner-Ville Distribution, it is possible to define a cross ambiguity function of the two signals $x(t)$ and $y(t)$:

$$A_{xy}(\tau,\upsilon) = \int_{-\infty}^{\infty} x(t+(\tau/2))y^*(t-(\tau/2))e^{-j2\pi\upsilon t}dt \tag{13}$$

hence, the cross Wigner-Ville distribution is:

$$WVD_{xy}(\tau,f) = \int_{-\infty}^{\infty} x(t+(\tau/2))y^*(t-(\tau/2))e^{-j2\pi f\tau}d\tau \tag{14}$$

If a signal $z(t)$ is composed by the two signals $x(t)$ and $y(t)$, then the Wigner-Ville distribution of $z(t)$ is:

$$WVD_z(\tau,f) = WVD_x(\tau,f) + WVD_y(\tau,f) + 2Re\{WVD_{xy}(\tau,f)\} \tag{15}$$

This implies that the Wigner-Ville distribution of a composite signal is the combination of the $WVD$ of every signal and the cross Wigner-Ville distribution of every two components. Hence, the Wigner-Ville distribution of $z(t)$ has $n$ $WVD$ and $\binom{n}{k}$ cross Wigner-Ville distribution, so every pair of signals has its cross term.
A cross term $WVD_{xy}$ in Equation (15) occurs at the mid-time, mid-frequency of the terms $WVD_x$ and $WVD_y$ [50].
Cross terms can distort the result of a spectral analysis of the signals $x(t)$ and $y(t)$. Many approaches have tried to reduce the cross-term interference while keeping the high resolution of Wigner-Ville distribution. The cross terms can also be helpful because they have significant information about the signals $x(t)$ and $y(t)$ and thus it could also be helpful for failure type detection and predictive maintenance. For example, if the vibration signals are different, the cross-term interferences are also different, which will also be helpful to distinguish the signals $x(t)$ and $y(t)$ [102].

### 5.1.4 Summary

Signal processing is an important step in failure type detection and predictive maintenance applications for real-world systems monitored by sensor data. For sensor data it is important to know which information is relevant. If only change of the sensor value

| Reference | Time Domain | FFT | STFT | WT | HHT | WVD |
|---|---|---|---|---|---|---|
| [65] | | X | | | | |
| [31] | | | | X | | |
| [119] | | X | | X | | |
| [116] | X | | | | | |
| [9] | | | X | | | |
| [60] | | | | X | | |
| [95] | | | | X | | |
| [102] | | | | | | X |
| [96] | | | | | | X |
| [107] | | | | | X | |
| [109] | | | | | X | |
| [108] | | | | | | X |
| [57] | | | | | X | |
| [40] | | | | | X | |
| **Sum** | **1** | **2** | **1** | **4** | **4** | **3** |

**Table 5.1:** Usage of signal processing in recent literature

over time is involved, the time domain analysis can be used to extract more information as a new value (see Section 5.1.1). If only the change in the frequency of sensor value is relevant, the frequency domain analysis must be used (see Section 5.1.1). As mentioned in Table 5.1, the most discussed signal processing technique in the research area of failure type detection and predictive maintenance is the time frequency analysis. Here, the value change over time and the frequency of value changes are relevant. The Short Time Fourier Transform (STFT) is discussed as an introduction to the terms of time frequency analysis (see Section 5.1.3.1). The STFT works with window functions where the window slides over time, and processes the sensor data inside the window. As discussed in Section 5.1.3.1, the drawback of the STFT is the fixed time and the frequency resolution. A technique related to the STFT is the Wavelet Transformation (see Section 5.1.3.2), which has wavelets instead of windows functions. These wavelets can have different time resolutions for different frequencies. A drawback of Signal processing techniques with windows functions, or wavelets, is that leakage effects can occur at the beginning or the end of the time window / wavelet [55].
The Wigner Ville distribution (WVD) has the best spectral resolution of all time frequency methods, and can analyze two signals at a time (see Section 5.1.3.4). Two signals that are compared by the WVD give rise to cross terms. These cross terms are generally not preferable, but they could be interesting in terms of failure type detection and predictive maintenance.
Hilbert Huang transformation (HHT) consists of two steps, first, the decomposition into

a finite number of intrinsic mode functions, and second, their transformation. It is possible to identify the physical meaning of the signal using this signal processing technique. This is a very interesting feature for failure type detection and predictive maintenance, but it is also expensive, and it is very special for a real-world system to define a HHT which interprets the physical meaning.
A decision tree can help to choose the to find the right signal processing technique for a given failure type detection and predictive maintenance problem (see Figure 5.8).

## 5.2 Feature Extraction

After the data acquisition and the signal processing, all features of the real-world system are available, and that feature set that can be enormous. It is sometimes necessary to reduce the number of features and increase the information content of the features to increase accuracy, and to reduce the computational effort of a machine learning approach for failure type detection and predictive maintenance. Feature extraction and/or feature selection can help in performing this. Feature selection selects a subset of the most representative features, whereas feature extraction transforms the original feature space and receive new information by combining features. For example, the combination of vibration sensors in a real world system, can give information about the environmental conditions. Feature extraction can be a non-linear process, and thus the results are not self-explanatory. An advantage of feature extraction over feature selection is that the feature space can be reduced to a much greater extent [61].
The linear discriminant analysis is a machine learning approach that will be discussed below (see Section 7.1.1). However, for the discriminant function that reduces the dimensionality of the feature, a vecture linear can be used for feature extraction.
Feature extraction is not often used in the area of failure type detection and predictive maintenance. In some works, the decomposition in the Hilbert Huang transformation (see Section 5.1.3.3) or the cross terms in the Wigner-Ville distribution (see Section 5.1.3.4) are called feature extraction. In [106], the features are normalized between 0 and 1, although there is no feature extraction process defined for failure type detection and predictive maintenance. Therefore, the feature extraction can be an area of future research, which will be discussed in Chapter 9.

## 5.3 Feature Selection

Feature selection is a common and useful technique to reduce computational cost and increase accuracy. As described in Section 5.2, a feature selection selects a subset of features that are defined on a representative basis. Of the different techniques that will be discussed in the following, there is no technique that selects features, especially for failure type detection and predictive maintenance scenarios. Therefore, feature selection can be an area of future research that will be discussed in Chapter 9.

### 5.3.1 Correlation analysis

The correlation analysis is one of the most common techniques to select features. This approach evaluates a linear relationship between pair-wise inputs with a correlation

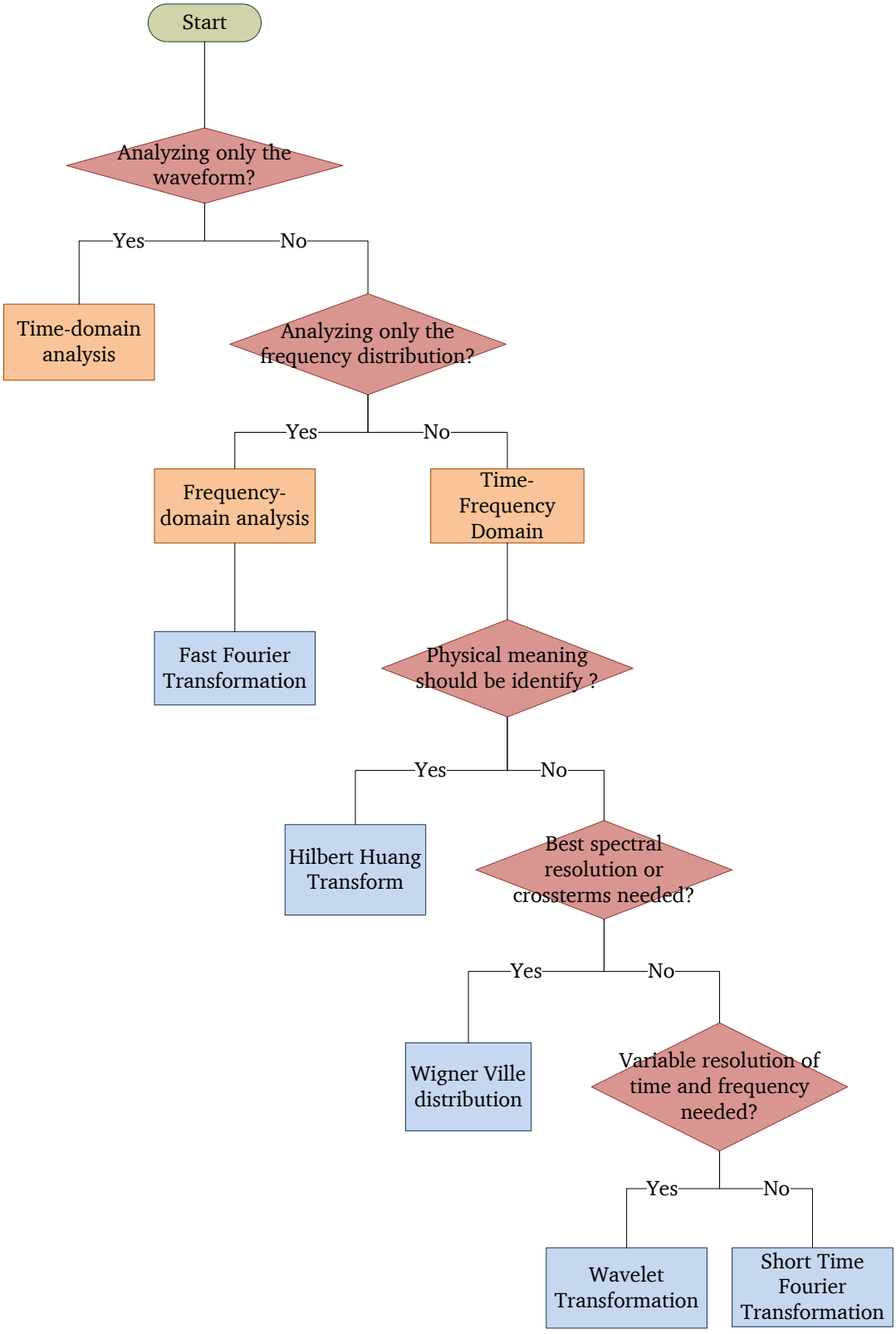



**Figure 5.8:** Decision tree to visualize the selection process of a signal processing technique.

function [38], which can remove features with redundant behaviors. This technique, however, fails with if the amount of sample data is low, or if the relationship between features is non-linear.

### 5.3.2 Principal component analysis

A principal component analysis [12] is an orthogonal transformation technique that converts a set of features into a set of linear uncorrelated features, called principal components. The assumption of the principal component analysis is that features with the largest variance have the largest informative content. The samples are centered and rotated in accordance to most relevant features. The outcome of the principal component analysis yields the features with the largest variance that are orthogonal to one another (see Figure 5.9).

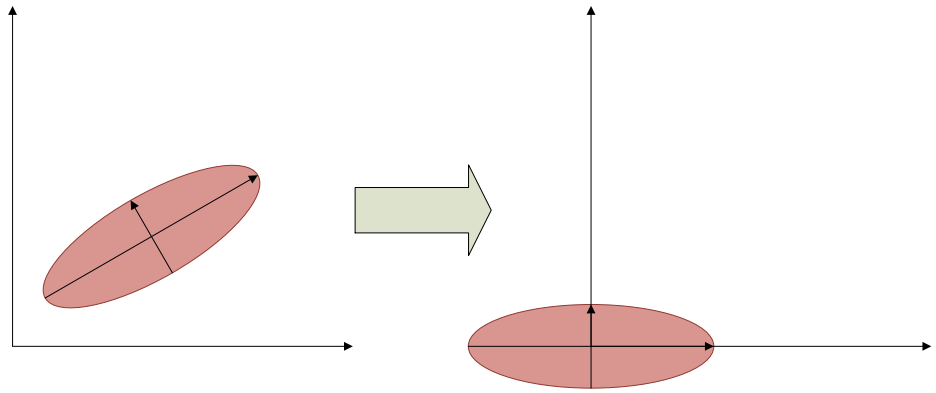


**Figure 5.9:** Visualization of the Principal Component Analysis. On the left, the original sample space (red ellipse) with the two largest variances that are orthogonal to each other (black arrows). On the right, the transformed sample space accorded to the most significant features [6].

### 5.3.3 Entropy

The term entropy was introduced by Shannon [80] in 1948. He tried to make assertions about the information content based upon statistical foundations. The requirements for the information quality are:

- The more rarely an event occurs, the higher the information content.
- The information content of an event chain is the sum of all events that occurred. $I(x_1, ..., x_n) = I(x_1) + ... + I(x_n)$ where $x_1$ to $x_n$ are events.
- The information content of a definite event is 0

The logarithm fulfills the requirements, and thus information content is defined as:

$$I(x) = \log_2(\frac{1}{p(x)}) \tag{16}$$

where $x$ is the event and $p(x)$ is the probability that the event $x$ occurs. The entropy $H$ of an event set is defined as:

$$H = \sum_{i=1}^{n} p(x_i) I(x_i) \tag{17}$$

$n$ is the number of events. The highest entropy $H$ exists when all events have the same probability ($H = \log_2(n)$). This implies that the lower the entropy, the more the information content concentrates on few events, and the higher the entropy $H(x)$ for a single event, the higher the information content for event x. This means, the less an event $x$ occurs, the higher the information content.
Entropy holds relevance for feature selection, because the lower the entropy $H$ of a feature, the higher its information content.
It could also be useful to measure the complexity of a feature. Accordingly, sample entropy [74], or multiscale entropy [25] based on sample entropy can be used. In sample entropy, the values in a time range of a feature are compared (i.e. with the Euclidean distance [27]) with those of the same feature of another time range. Multiscale entropy uses the process of sample entropy, but with the time window scaled.

### 5.3.4 Other approaches

Machine learning approaches, such as generic algorithms and decision trees, which we will discuss below in Chapter 7, can also been used for feature selection [89], [90]. The idea behind using machine learning approaches for feature selection is to train a machine learning algorithm and analyze the structure learned. Generally, some features are more relevant, and others less relevant to reaching a correlation between the input value and a given output. The more commonly used features are the important ones. To reach less computational cost and more accuracy, another machine learning approach uses only the more frequently used features of the first machine learning approach [20]. After feature engineering, the learning process starts. The next section discusses opportunities of labeling historic data, which, like in feature engineering, also have influence on the accuracy and computational costs.

# 6 Data Labeling

As described in Section 3.3, a data-driven model is realized by a machine learning approach based upon statistical and learning techniques. Supervised learning is the most common learning technique for failure type detection and predictive maintenance (see Section 3.3.1). For supervised learning, historical data is needed (see Figure 3.3), which has to be labeled. Hence, the machine learning approach can create relationships between the input feature vector and the expected result.
Previous sections showed that there was no difference between a failure type detection and a predictive maintenance approach. However, this changes with data labeling, and the reasons will be discussed below.
In the literature of failure type detection and predictive maintenance, two different types of data labeling techniques exist:

- State-Driven Data Labeling
- Residual Useful Lifetime

Machine learning can distinguish between classification and regression. Classification means that an output variable takes on defined class labels. Regression means that an output variable takes on continuous values. The state-driven data labeling technique is used for classification in failure type detection and predictive maintenance. The residual useful lifetime estimation is a regression problem for predictive maintenance.
The following two sections will examine the techniques and their implementation, followed by a discussion of the prediction window, which is important to schedule maintenance actions.
Table 6.1 gives an overview of the data labeling technique used in the research area of failure type detection and predictive maintenance. The table shows that most approaches use the state-driven data labeling technique. This result is expected because failure type detection can only realize with a state-driven data labeling technique and for a predictive maintenance scenario it is easier to archive a sufficient accuracy by classification. The reason is the finite number of possible classes. It is more sophisticate to archive a good accuracy for a regression problem for predictive maintenance because the solution space is infinite and therefore the degradation process must be better understood.

## 6.1 State Driven Data Labeling

The state-driven data labeling technique labels the historic data into different states. There are two different types of states in failure type detection and predictive maintenance:

- Failure type state
- Lifetime state

One component can have one or more failure types. A failure type can also depend on a failure that occurred before (see Figure 6.1).

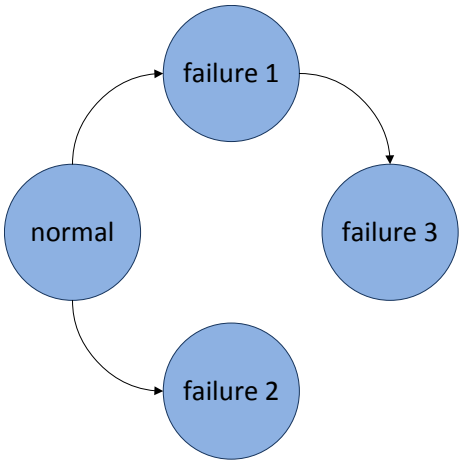



**Figure 6.1:** An example of a state machine for failure type detection.

The lifetime state is usually equally distributed over the lifetime of a component. Figure 6.2 visualize a four-state equal distribution of the lifetime of a component. The states of failure type and lifetime can be combined into a more complex model. Such a model shows which failure can occur at which lifetime (see Figure 6.3), which could be helpful in scheduling maintenance actions.

According to recent literature, the historic data with logged failure types is needed for

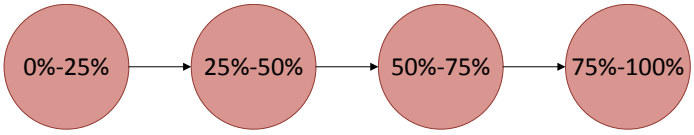



**Figure 6.2:** A state machine for a component with equal distributed states over the lifetime. Such state machines can be used in a predictive maintenance scenario.

detecting these different failure types. This implies that every failure type must occur in the historic data, and the more often it occurs, the higher the possibility to eliminate noise factors and achieve a better accuracy. The problem is that failures, especially in most failure type and predictive maintenance scenarios, are often costly, and therefore it is not always possible to direct the system into a failure situation. Accordingly, many real-world systems are maintained with a preventive technique, where components are replaced after a predefined time, before a failure can occur. It does not matter whether the component is close to failing or not. Failure type detection, without the failure type of historic data, is not easy and not mentioned in the literature. However, it might be useful in analyzing failure types, because some these do not lead to a system crash,

but can influence other components and their healthy state. The necessity of logged failures and their influence on other components will be discussed in the future work (see Chapter9).
In lifetime-state data labeling for predictive maintenance, it is necessary to categorize the historic data into the lifetime states used. As mentioned above, these states are generally equally distributed over the lifetime of a component. There are two minimum number of states for predictive maintenance (<100% and 100%). In contrast to the current failure type detection methods, for predictive maintenance reasons it is possible to log only healthy data, and when an observed real-world system leaves the healthy state, it must be failed [64]. In a predictive maintenance scenario, a failed state means that a breakdown of the component and the lifetime is 100%. This means that failures of the component that might occur before are not interesting for a predictive maintenance scenario.
In recent literature on failure type detection and predictive maintenance with machine learning techniques and state-driven data labeling, the accuracy of classification usually changes, and becomes better at the end of the useful lifetime of a component. The reason for this is that in the lifetime of a component, events often occur that correlate with the degradation state, and such events occur especially at the end of the useful lifetime.

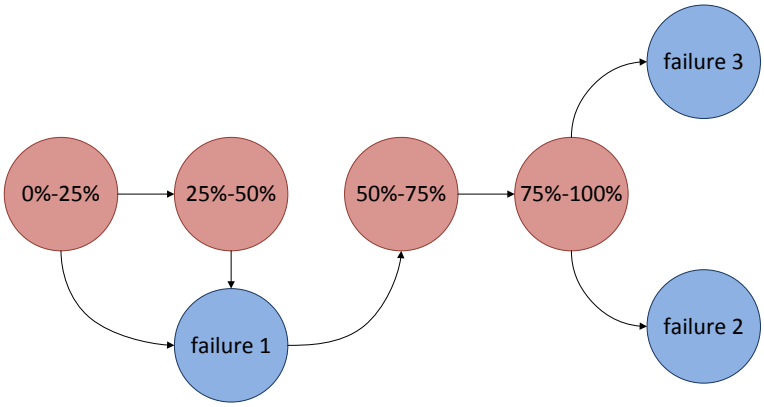



**Figure 6.3:** A combination of distributed states over the lifetime and failure types.

## 6.2 Residual Useful Lifetime

The residual useful lifetime prediction is a regression problem only relevant in predictive maintenance scenarios, because it returns one continuous value, which represents the rest of useful lifetime and not the type of failure. There are different ways to estimate the residual useful lifetime. It is possible to calculate it on top of a state-driven data labeling technique [60]. It is also possible to calculate it using statistical approaches [83].
The accuracy of machine learning approaches with state-driven data labeling techniques

are usually higher than in estimating the residual useful lifetime. The reasons are obvious; namely, for a continuous estimation, it is easier to classify the residual lifetime into a discrete number of states, usually not more than ten. Furthermore, the literature shows that the closer the system is to failure, the higher the accuracy. This could be relevant for the prediction window, which is discussed as follows.

## 6.3 Prediction Window

A prediction window defines the time prior to when the system fails and the failure type detection and predictive maintenance approach can or has to know that the system is failing. Therefore, there are two ways to define a prediction window:

- The prediction window is defined with respect to the time needed to schedule the repair.
- The prediction window is defined with respect to the possibility of recognizing a system failure.

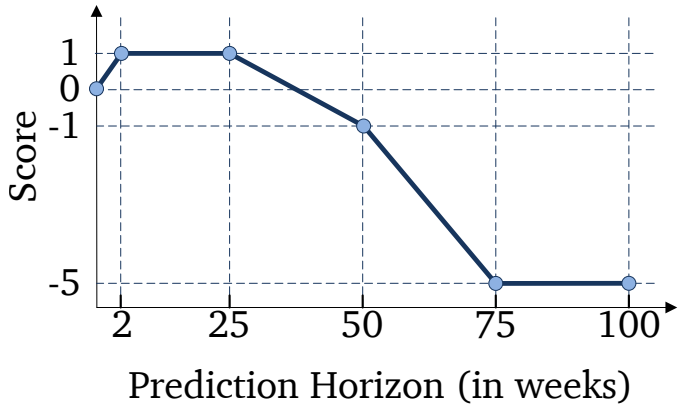



**Figure 6.4:** Score function for evaluating classification rules [67].

There are some ways to handle this requirement. [67] defines a score function that not only rates the accuracy, but also the prediction horizon. The approach generate a rule based system with two states (healthy, and faulty). To asses the generated rules in terms of prediction horizon Figure 6.3 shows the score function. The preferred time for issuing the warning is between 2 and 25 weeks before the fault occurs. Less than that means that the replacement operation may not be possible to schedule. Longer means that the costs are to high for predictive maintenance, because the component is still functional. [65] defines a cost matrix for the time it makes sense to repair or change a component. In this cost matrix misclassification for an unsafe condition to be a safe condition is much more severe than misclassifying a safe condition to be in an unsafe condition.

## 6.4 Summary

This chapter discusses the two different labeling techniques and the properties of the prediction window for failure type detection and predictive maintenance. It is shown that the state-driven data labeling technique is used in a failure type detection scenario (see Section 6.1). It is possible to use the state-driven data labeling technique also for predicting future maintenance actions. Therefore, the lifetime state labeling technique is used. Usually, these states are equally distributed over the lifetime of a component.
A more precise but less accurate way is to estimate the residual useful lifetime by regression (see Section 6.2). It is shown that this technique is only used for predictive maintenance problems.
The definition of prediction window shows that there are two possibilities to define such a window (see Section 6.3).
The recent literature on failure type detection and predictive maintenance mentions data labeling only marginally. This is an interesting fact, as an adequate data labeling technique can improve the results of a data-driven model. Accordingly, there is a lot of future work potential in the area of data labeling which will discussed in Chapter 9.

| Reference | Failure Type | State Driven | RUL |
|---|---|---|---|
| [44] | | 4 | |
| [65] | | 2 | |
| [28] | | | X |
| [51] | | | X |
| [37] | | 2 | |
| [20] | 4 | | |
| [94] | | | X |
| [31] | | 4 | |
| [119] | | 4 | |
| [42] | 9 | | |
| [111] | 8 | | |
| [110] | 7 | | |
| [41] | | 3 | |
| [116] | 4 | | |
| [19] | | 2 | |
| [64] | | 2 | |

| Reference | Failure Type | State Driven | RUL |
|---|---|---|---|
| [115] | | 2 | |
| [121] | 2 | | |
| [9] | | 3 | |
| [60] | | | X |
| [95] | | | X |
| [83] | | | X |
| [54] | 39 | | |
| [82] | 4 | | |
| [58] | | 5 | |
| [67] | | 2 | |
| [102] | 8 | | |
| [106] | 5 | | |
| [107] | 5 | | |
| [109] | 5 | | |
| [117] | 10 | | |
| [108] | 7 | | |
| [57] | 7 | | |
| [56] | 5 | | |
| [40] | 2 | | |
| [30] | | | X |
| **Sum** | **17** | **12** | **7** |

**Table 6.1:** Usage of data labeling technique in recent literature of failure type detection and predictive maintenance. The number in the Failure Type column is the number of different failure types which the approach tries to separate. The number in the State Driven column is the number of different states dividing the lifetime that the approach tries to classify.

# 7 Machine Learning Approaches

Machine learning involves programming a computer to assess data in respect of samples given and/or the experience of domain experts. A machine learning approach tries to identify the relation between an input vector and a result. The machine learning algorithm has to optimize the parameter of its model.
In the recent years, machine learning has become increasingly important in computer science because data could be collected and stored much more easily. The data collected is usually so extensive that it is not practical to analyze the data manually. In such a scenario, the machine learning technique plays a key role.
Another reason for the growing popularity of machine learning are decreasing computational costs. With the evolution of hardware in recent years, the usage of machine learning approaches has become efficient in terms of both time and money, especially for failure type detection and predictive maintenance.
As discussed in the previous chapter, the following are important for obtaining results from the machine learning algorithm: deciding on the learning technique (see Section 3.3), the quality of the input features (see Chapter 5), and in case of supervised learning, the labeling technique (see Chapter 6). This section reviews the methods of analysis based on machine learning techniques for predicting and analyzing failures in a real-world system. The techniques considered have been used in the recent literature of failure type detection and predictive maintenance. The techniques reviewed there are separated into two categories:

- Parametric machine learning approaches
- Nonparametric machine learning approaches

## 7.1 Parametric

A statistic is an arbitrary computed value based on samples. In a statistical inference process, decision are made based upon the information obtained from a sample. A parametric approach makes the assumption, that the samples are distributed in a known model, for example in a Gaussian distribution. The advantage of such a parametric approach is that the model is based upon few parameters, like mean value or variance, the so-called satisfactory statistics of distribution. The machine learning approach estimates the parameters with respect to the given samples and makes the assumption, that this distribution is valid for all data. The distribution forms the basis of the decisions made by the machine learning approach.
A parametric approach computes a decision faster than a nonparametric approach, but it needs exact assumptions for the distribution to achieve a good result. If the assumption about the parameters is not exact, the result will seem to be interpreted arbitrarily.
In the following three parametric examples, linear discriminant analysis and Bayesian networks and the Hidden semi-Markov model will be discussed. These are the only three parametric approaches which are used in the considered literature.

### 7.1.1 Linear discriminant analysis

Linear discriminant analysis is a method for dimensional reduction and classification. The technique of linear dependent dimensional reduction with the discriminant function can also be used for feature extraction (see Section 5.2).
The simplest representation of a linear discriminant analysis is the linear function:

$$y(\mathbf{x}) = \mathbf{w}^{\mathbf{T}}\mathbf{x} + w_0 \tag{18}$$

where $\mathbf{x}$ is the input vector $\mathbf{w}^{\mathbf{T}}$ is a weighted vector and $w_0$ is a bias. For a binary classifier with classes $C_1$ and $C_2$, a classification of input vector $\mathbf{x}$ will be $C_1$ if $y(\mathbf{x}) \geq 0$, otherwise $C_2$. The projection from $\mathbf{x}$ to $\mathbf{w}^{\mathbf{T}}$ is a dimensional reduction from $d$ to 1. The vector $\mathbf{w}$ is orthogonal to every vector lying within the decision surface. Similarly, if $y(\mathbf{x}) = 0$, this implies that $\mathbf{x}$ is a point on a decision surface, the normal distance from the origin to the decision surface is expressed with

$$\frac{\mathbf{w}^{\mathbf{T}}\mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_o}{\|\mathbf{w}\|} \tag{19}$$

The illustration on the left in Figure 7.1 shows the geometry for a two dimensional discriminant function in two dimensions. The decision surface, shown in red, is perpendicular to $\mathbf{w}$, and its displacement from the origin is controlled by the bias parameter $w_0$. Also, the signed orthogonal distance of a general point $\mathbf{x}$ from the decision surface is given by $\frac{y(x)}{\|\mathbf{w}\|}$ [12].
The best way to handle a classifier with more than two classes is to define $k$ linear

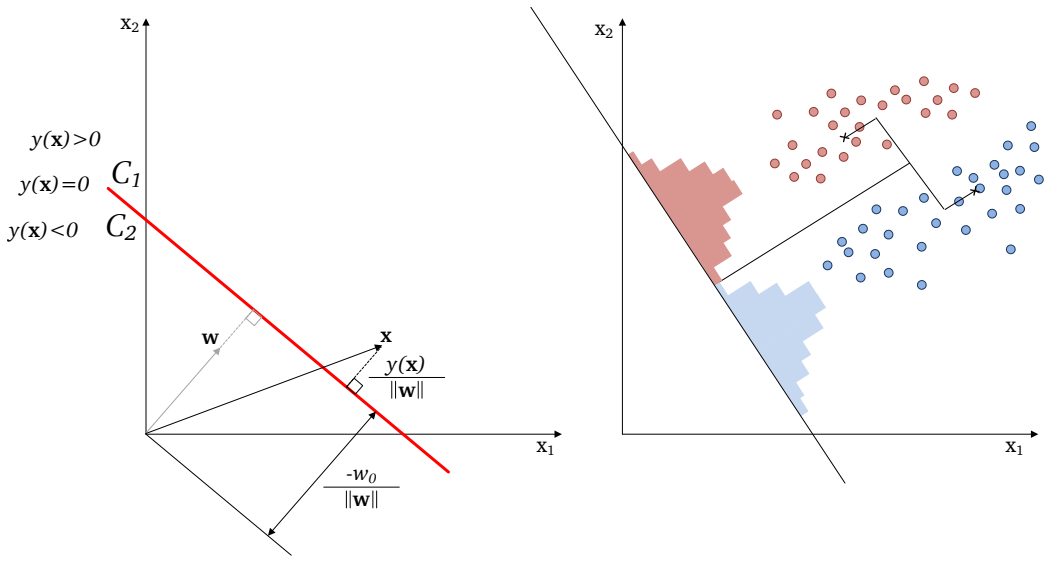



**Figure 7.1:** An illustration of a discriminant function in two dimensions (left) and a plot on the right which samples two classes (red and blue) by using the Fisher linear discriminant. [12]

functions for $k$ classes

$$y_k(\mathbf{x}) = \mathbf{w_k^T x} + w_{k0} \tag{20}$$

a vector $\mathbf{x}$ will be assigned to a class $C_k$ if

$$y_k(\mathbf{x}) > y_j(\mathbf{x}) \qquad \forall j \neq k \tag{21}$$

The Fisher's linear discriminant can be used to define a weighted vector $\mathbf{w^T}$ that maximizes the class separation. If a class $C_1$ has $N_1$ points and $C_2$ has $N_2$ points, the mean vector of $C_1$ and $C_2$ can be expressed as

$$\mathbf{m_1} = \frac{1}{N_1} \sum_{n \in C_1} \mathbf{x_n}, \qquad \mathbf{m_2} = \frac{1}{N_2} \sum_{n \in C_2} \mathbf{x_n} \tag{22}$$

The idea is to choose $\mathbf{w}$ to maximize

$$m_1 - m_2 = \mathbf{w^T}(\mathbf{m_1} - \mathbf{m_2}) \tag{23}$$

where

$$m_k = \mathbf{w^T m_k} \tag{24}$$

As described in equation Equation (18), when a vector transforms $\mathbf{x}$ to a one-dimension space $y$, the variance of the transformed data from class $C_k$ can expressed as

$$s_k^2 = \sum_{n \in C_k} (y_n - m_k)^2 \tag{25}$$

where $y_n = \mathbf{w^T x_n}$. To separate the two classes from each other, the mean value after a transformation should have a distance as large as possible by having a small variance. Therefore, a high value for $|m_1 - m_2|$ and a low value for $s_1^2 + s_2^2$ is desirable. Fisher's criterion maximizes $\mathbf{w}$ to fulfill the described definition with

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} \tag{26}$$

With Equation (18), Equation (24) and Equation (25) the Fisher criterion can be rewritten as

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S_B w}}{\mathbf{w}^T \mathbf{S_W w}} \tag{27}$$

where $S_B$ is the between-class covariance matrix defined as

$$\mathbf{S_B} = (\mathbf{m_2} - \mathbf{m_1})(\mathbf{m_2} - \mathbf{m_1})^{\mathrm{T}} \tag{28}$$

and $S_W$ is the within-class covariance matrix, expressed as

$$\mathbf{S_W} = \sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}} + \sum_{n \in C_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^{\mathrm{T}} \tag{29}$$

Differentiating Equation (27) with respect to $\mathbf{w}$, $J(\mathbf{w})$ is maximized when

$$(\mathbf{w}^T \mathbf{S_B} \mathbf{w}) \mathbf{S_W} \mathbf{w} = (\mathbf{w}^T \mathbf{S_W} \mathbf{w}) \mathbf{S_B} \mathbf{w} \tag{30}$$

$(\mathbf{w}^T \mathbf{S_B} \mathbf{w})$ and $(\mathbf{w}^T \mathbf{S_W} \mathbf{w})$ are scalar factors; thus, these factors do not provide information about the direction of $\mathbf{w}$ and can be dropped. Equation (30) can multiply both sides with $\mathbf{S_W^{-1}}$, so the result of the Fisher linear discriminant can expressed as

$$\mathbf{w} = \mathbf{S_W^{-1}} (\mathbf{m_2} - \mathbf{m_1}) \tag{31}$$

The right hand side of Equation (31) shows the variance of the mean value and the Fisher linear discriminant of an example separation problem. The right plot in Figure 7.1 shows a two class example (depicted in red and blue), along with the histogram resulting from a projection onto the line, joining the class means by using the Fisher linear discriminant [12]. The linear discriminant analysis is an easy and useful classification technique, when the classes can be linearly separated from each other. It is also useful for feature extraction when transforming a set of features $\mathbf{x}$ to a one-dimensional feature $y(\mathbf{x})$. One of its drawbacks is the creation of a weighted vector, which is difficult, because they need a linear dependency on each other, and that is not automatically given in real-world systems. But still it is possible to have such linear dependencies in the data from a data acquisition system.

### 7.1.2 Bayesian networks

A Bayesian network is a diagrammatic representation of probability distributions. It has several useful properties:

- Visualize the structure of a probabilistic model
- Conditional dependence properties can be obtained by inspection the model
- Complex computations can be expressed in terms of graphical manipulations

A Bayesian network can be represented as a directed acyclic graph, where every node in the graph represents a variable. Figure 7.2 shows an example graph representing a Bayesian network with six random variables. The probability distribution of the combination of these random variables is called joint distribution. The joint distribution of the example is given by:

$$p(x_1)p(x_2)p(x_3|x_1)p(x_4|x_3)p(x_5|x_1,x_2,x_3)p(x_6|x_2,x_4) \tag{32}$$

For a graph with $k$ nodes, the joint distribution can generally be expressed as

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\mathrm{pa_k}) \tag{33}$$

where $pa_k$ are the parent nodes of $x_k$. The probabilities $p(x_k|pa_b)$ of the possible combinations (see Equation (33)) can be listed in a conditional probability table.
There are two ways to learn about a Bayesian network. First, it exists as the structure of a network and so it is not necessary to mention the conditional independencies. It is only necessary to calculate the conditional probability on every node. The second way is to create a network structure with conditional independencies and a conditional probability on every Node. For failure type detection and predictive maintenance, such an

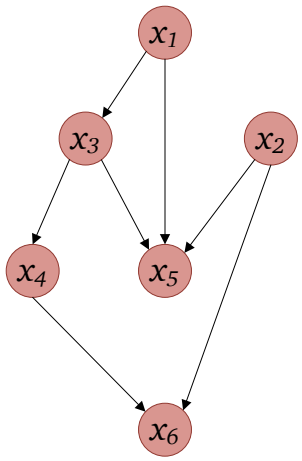



**Figure 7.2:** An example of a Bayesian network

approach is useful for state detection with the supervised learning technique (see Section 3.3.1) [109]. The labels of the historic data (see Section 6.1) contains the states that are represented in a Bayesian network by variables and nodes. The state with the highest probability for a given input vector is chosen. A residual lifetime estimation can be established when the duration in every state is considered and evaluated [60].

### 7.1.3 Hidden semi-Markov model

A hidden semi-Markov model is an extension of hidden Markov models [17]. A hidden Markov model [72] assumes that the real-world system being modeled is a Markov chain [33] with unobserved (hidden) states and transitions from one to another on a state space.
A hidden semi-Markov model allows the underlying process to be a semi-Markov chain with a duration for each state. A hidden semi-Markov model can use a parametric or nonparametric distribution family for the state duration. The choice of distribution family for the state duration is central to the use of the hidden semi-Markov model [113]. In failure type detection and predictive maintenance scenarios, the parametric hidden semi-Markov model is preferred, because the computational effort of a nonparametric hidden semi-Markov model is much higher than with the parametric approach [31].
In terms of failure type detection and predictive maintenance, a state represents a health state (see Section 6.1) of the real-world system. A health state duration is measured in time units, which can be represented as time of usage, working cycles, maintenance

intervals, etc. A time unit depends on observations of the real-world system, which represent a single state. A variable amount of single states are combined to a state in the hidden semi-Markov model. The nature of these models is a state-driven approach, an example is described in Figure 7.3.
A prognosis of the residual useful lifetime (see Section 6.2) is also possible with a hidden semi-Markov model, and is described in [31].

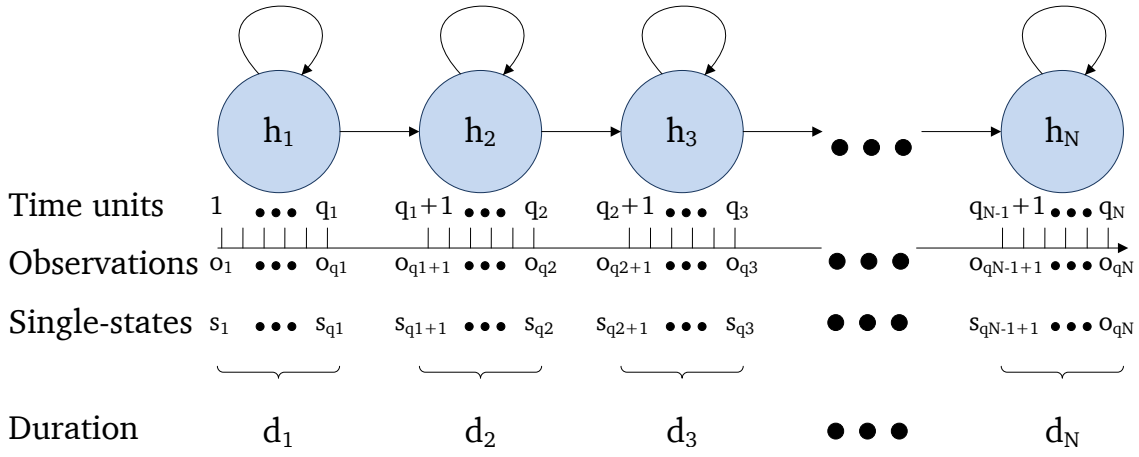



**Figure 7.3:** An example of a hidden semi-Markov model, where the blue cycles are states of the model. The black arrows are the transitions between the states. Every state has its own duration ($d_1$,...,$d_n$) of time units (1,...,$q_n$). These time units are observations of the real-world system ($o_1$,...,$o_{qN}$) and every observation represent a single state ($s_1$,...,$s_{qN}$).

## 7.2 Nonparametric

The previous section discussed the parametric machine learning approaches which estimates the satisfactory statistics of a process. Such an approach computes a decision faster, and if the estimated parameters are correct, the results of the parametric approach are good.
A process having no satisfying statistics estimated cannot be processed by a parametric machine learning technique with any good result. In this case, the nonparametric machine learning techniques are used. Theses techniques are based on the assumption that inputs that are close to one another must have the same results. The difference between such techniques relates to the definition of "close to another".
Some of the nonparametric machine learning techniques used in recent literature concerning failure type detection and predictive maintenance will be discussed in the following Sections.
A result of investigating recent literature indicates that nonparametric machine learning techniques have more stable results than the parametric machine learning techniques. A possible reason for this is discussed above, namely that the statistics satisfying a failure type detection and predictive maintenance process are too difficult to estimate exactly

because most real-world systems are non-linear and often associated with uncertainties, due to noises, uncertain physical parameters, or incomplete knowledge ([1], [73] and [46]).

### 7.2.1 Belief rule based

An extension of traditional IF-THEN-ELSE rule based systems are belief-rule-based systems, which are capable of capturing complicated non-linear causal relationships between antecedent attributes and consequents [21]. The belief-rule-based machine learning technique can use the information from a domain expert and historic data (see Section 3.4) to establish a non-linear relationship between attributes and an associated consequence. Because of the combined learning technique, such a model converges fast and the training time is short.
A belief rule based system in a predictive maintenance scenario captures one variable $x$ at time $t$ and make an assumption about $x$ at time $t+1$. A belief rule can be constructed as follows:

$$\begin{aligned} R_k \quad : \quad & \text{If } x(t) \text{ is } D_k \\ & \text{Then } x(t+1) \text{ is } \{(D_1, \beta_{1,k}), ..., (D_N, \beta_{N,k}), (D, \beta_{D,k})\} \end{aligned} \tag{34}$$

$R_k$ denotes the $k^{th}$ belief rule of the belief rule based system. $x(t)$ is the value of the variable $x$ at time $t$. $x(t+1)$ is the value of the variable $x$ at time $t+1$ which should be predicted from the belief rule. $D_n \in D$ and $D = \{D_1, ..., D_N\}$ is the $n^{th}$ consequent in the belief rule based system. $\beta_{j,k}$ $j = (1, ..., N), k = (1, ..., N)$ is the belief degree assessed to $D_j$ and $D_j \in D$. The last element of the Then expression are remaining belief degrees unassigned to any $D_j$, which reflects the uncertainty of assessment due to a limitation or lack of prior knowledge.
A rule, as described in Equation (34) for a failure type detection or predictive maintenance problem, can be created as follows:

1. Extracting belief rules from domain expert knowledge
2. Extracting belief rules by examining historical data
3. Using the previous belief-rule bases if available
4. Random rules without any prior knowledge

These options described above are mostly combined in a failure type detection and predictive maintenance scenario. A domain expert defines the first version of belief rules, which are refined by examining the historical data and the previous decisions.

### 7.2.2 Decision tree models

Decision tree models are highly related to the belief-rule-based system (see Section 7.2.1) as a decision tree can be viewed as a set of non-overlapping rules [3]. Figure 7.4 shows an example of a decision tree. There are three variables which are checked ($X_1$,

$X_2$ and $X_3$). One ellipse represent one rule of the example system. The result of the rule described on the arrows, and the leaves depict the result of the decision tree.
A decision tree can be learned as a belief rule based system from a combination of domain expert knowledge and historic data (see Section 3.4). But all decision-tree-based machine learning algorithms mentioned for failure type detection and predictive maintenance follow two steps in creating the decision tree:

- Building a decision tree by learning with the supervised learning technique (see Section 3.3.1)
- Pruning the decision tree

In some algorithms, the both steps are done concurrently. The most popular decision-tree-based learning algorithms are the C4.5 [71] and C5.0 [16]. C5.0 is an update of C4.5, and offers the advantages of high purity, high computing speed and excellent efficiency in memory usage. New features, such as boosting and variable misclassification costs, are implemented in C5.0.
Decision-tree-based machine learning techniques are frequently used for failure type detection and predictive maintenance, but only to classify the state of the real-world system (see Section 6.1), and not for regressing the residual useful lifetime (see Section 6.2). The reason is that a decision tree can only have a finite number of leafs, which represent the possible results. Because their number is finite it is not possible to estimate continuous values.
No approach combines failure type detection and predictive maintenance in one decision tree in current research. [42] has implemented two decision trees: a decision tree identifying the state of the system as well as any failure in the near future, and a tree identifying the type of upcoming failures.
A decision tree can also be used for feature selection (see Section 5.3.4). [42] Uses the resulting rules of one such a algorithm to identify the most used features, by which a neural network is learned. The result shows a significantly better performance in terms of accuracy and computational cost.

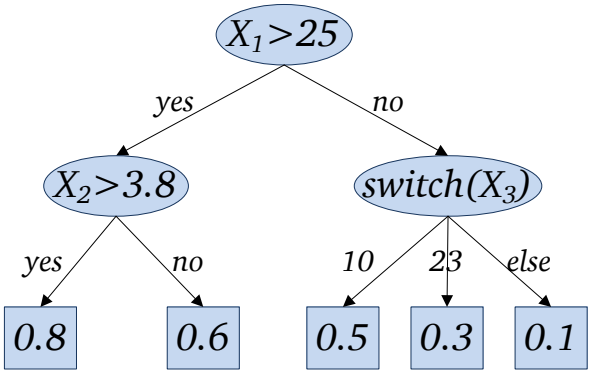



**Figure 7.4:** An example of a decision tree.

#### 7.2.2.1 Random forest

A random forest is an ensemble method, which trains multiple learners and then uses them in combinations. It aims at increasing the accuracy over that of a single learner [120].
A random forest operates by constructing a set of decision trees in the learning phase, and the output is the class chosen of most decision trees (classification), or the mean of the result (regression). The idea of ensemble learning was first mentioned by Ho [43], and that of the random forest by [13].
A decision tree of a random forest is created with a random subset of features and a random set of training data, with respect to the features from the historical data.
The advantages of a random forest are:

- Training time is reduced, because a decision tree learns from a subset of features, which implies less historic data to learn, and parallelization in learning the decision trees.
- The evaluation can be parallelizable with small trees. Thus, the evaluation time can be shorter than in a single decision tree and can used for large data analysis.

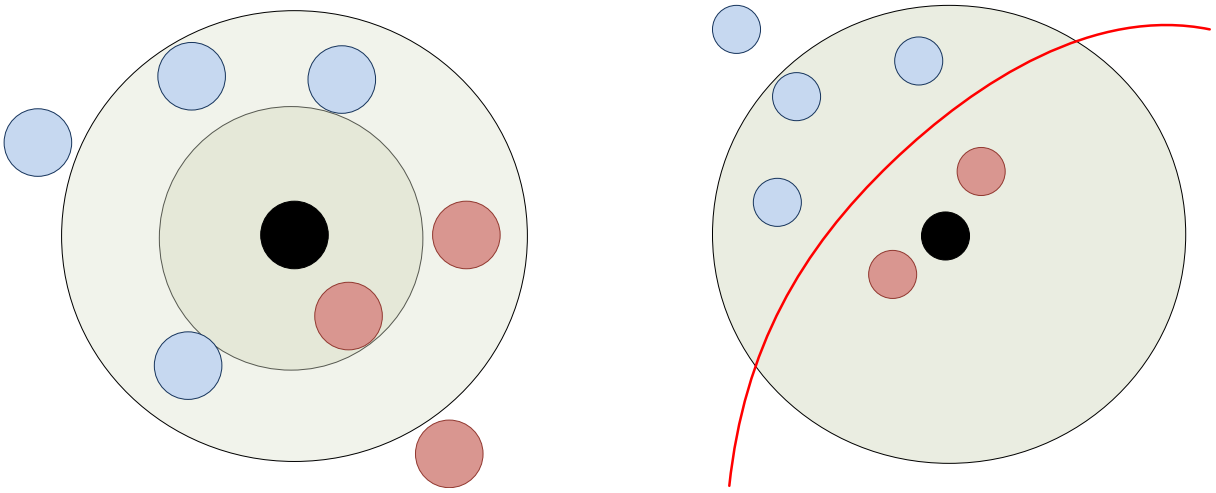
According to recent literature on failure type detection and predictive maintenance, the random forest machine learning approach is used due to its low computational cost with large data and stable results.

### 7.2.3 Nearest neighbor

The nearest neighbor machine learning technique is a nonparametric method used to estimate the probability density function [12]. It uses a distance metric like eukledian distance [27], hamming distance [49], or Manhattan-metric [4], to identify the neighborhood of a given input vector which consists of instances that have been input previously.
For achieving a stable result, $k$-nearest neighbors are examined Figure 5(a). The nearest neighborhood machine learning technique is a lazy learning system [118] where no model learned at all [5].
For failure type detection and predictive maintenance, a very simple nearest neighbor machine learning technique can be used. This technique has some drawbacks that need to be mentioned with its use for failure type detection and predictive maintenance:

- As mentioned in Section 6.1, training sets are sometimes not equally distributed. In this case correct classification can not be achieved. For example, see Figure 5(b), where few examples of a class (red points) lead to a misclassification.
- The distance metric has a significant influence on the result. To achieve a good result, the non-linear behavior of real-world system needs different distance metrics for the lifetime.

Despite the drawbacks, the nearest neighbor is used for failure type detection and predictive maintenance, due to the lazy learning technique and simple usage.



(a) Example of $k$-nearest neighbor classification with different $k$ ($k$=1 inner ring and $k$=5 outer ring). The black point is the instance to classify. The red points and the blue point visualize two different classes.

(b) Example of misclassification at a border between two classes, where the black point is the one which needs to be classified, the correct classification is red, but with $k$=5 the classification is blue

**Figure 7.5:** Example for nearest neighbor classification

### 7.2.4 Artificial neural networks

An artificial neural network is an abstraction of a network of nerve cells found in brains or spinal marrow. In the area of machine learning, an artificial neural network tries to adapt to the information flow of the biological neural network. The purpose is not to replace it, this is a part of computational neuroscience [97].
An artificial neural network can be of differing complexity. For example, such a network may contain multiple layers of neurons. There exists an input layer, a defined number of hidden layers and an output layer. Figure 7.6 depicts an example of an artificial neural network with two hidden layers and three output nodes at the output layer. The simplest structure is a two-layer neural network with an input and an output layer, where the number of input neurons equals the size of the input vector. The connections between the neurons have weights. Every neuron has an activation function [12]. The most common activation function is the sigmoid and radial base function [66]. The usual way of designing an artificial neural network is as a feed-forward neural network, which can be represented as a directed acyclic graph from the input to the output nodes. Theoretically, an artificial neural network can learn by the following operations:

- Creation of new neurons with new connections, removal of existing neurons and their connections
- Creation of new connections between existing neurons, removing existing connections between neurons

- Changing of the weights between neurons
- Adjustment of the neurons thresholds
- Changing the properties of the activation function

The basic learning principle of an artificial neural network using the gradient or steepest descent method [12], which uses the error between output of the artificial neural network and the labeled output. Afterwards, the weights of the neurons will adjust repeatedly. An artificial neural network with a radial basis function has the property that each basis function depends on the radial distance (typically Euclidean) from a center $\mu_j$ where $j$ is the number of neurons in the hidden layer. The center $\mu_j$ can change to have the best possible approximation of the original function. Accordingly, the learning improvement is propagated iteratively from the output to the input nodes in a technique called back propagation [12].

In the area of machine learning research, artificial neural networks have attracted

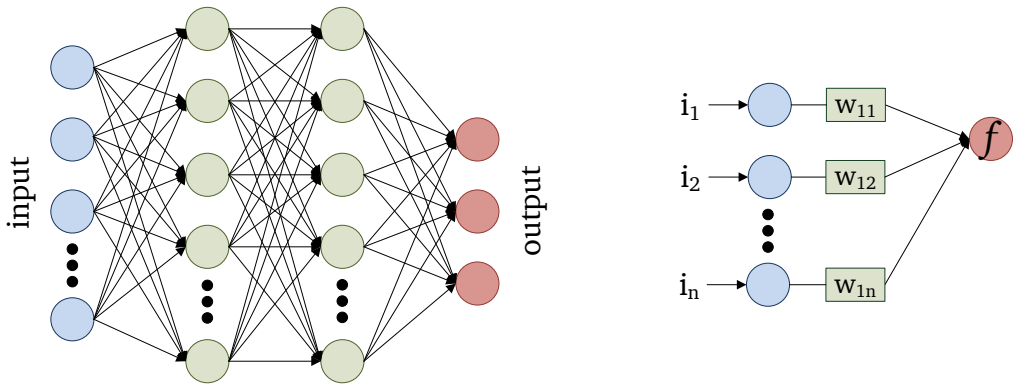



**Figure 7.6:** On the left, an example of an artificial neural network with two hidden layers. The right graphic shows the weights between neurons and an activation function.

enormous attention in recent years [53]. This extends to failure type detection and predictive maintenance data-driven systems, because artificial neural networks have useful properties for failure type detection and predictive maintenance [54]:

- They can handle nonlinear and undetermined processes where no model is available, and learn the diagnosis from the information of the learning data.
- They are noise tolerant and work well with noisy measurements.

Despite the benefits of a back propagation neural network, it also has certain drawbacks [108]:

- Existence of local minimum where the network is unable to achieve convergence to the global minimum.

- The need of many iterations to achieve a good learning rate.

In the following subsections, the most common artificial neural network types for recent failure type detection and predictive maintenance research will be discussed, They try to solve the drawbacks of implementing a standard back propagation neural network.

#### 7.2.4.1 Probabilistic neural networks

A probabilistic neural network [86] has a similar structure as the artificial back propagation neural network discussed above. The main difference is the replacement of the sigmoid activation function with an index function. Results of probabilistics neural networks can approximate the nonlinear decision boundary of the Bayes optimal decision surface. These networks do not need the iterative training procedure. They converge fast, even with a low number of training samples [102]. Therefore, the training time and classification time are also acceptable, even for high-dimension input vectors.
The probabilistic neural network is only a classifying machine learning technique. It is suitable for a state-driven (see Section 6.1), but not for a residual useful lifetime (see Section 6.2) estimation. A modified artificial neural network technique for such a continuous regression problem as the residual useful lifetime will be discussed in Section 7.2.4.2.
Improvements of the probabilistic neural network, over a standard back propagation neural network, have made the probabilistic neural network suitable for a failure type detection and predictive maintenance system, where a state-driven data labeling technique is used. Figure 7.7 shows the abstract structure of a probabilistic neural network with the input layer, the pattern layer with the activation function, the summation layer, and the output layer, which is the classification result.

#### 7.2.4.2 Generalized regression neural network

In contrast to the artificial neural network discussed in Section 7.2.4, a generalized regression neural network does not need an iterative training process nor weight calculations that can lead to a local minima problem [87].
As in standard regression techniques, the generalized regression neural network is used for the estimation of continuous variables. It is related to the kernel function network, and is based upon a standard statistical technique, called kernel regression. The regression of a dependent variable y (output layer) on an independent x (input vector) estimates the most probable value for y by a given x and a training data set. The generalized regression neural network method will produce the estimated value of y, which minimizes the mean-squared error, which is defined as:

$$\mathbf{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{35}$$

Where $n$ is the number of examples, $\hat{y}_i$ is the $i^{th}$ estimated value and $y_1$ is the $i^{th}$ real value.
As described in Section 7.2, estimating the probability density function of many real-world systems is either difficult or not feasible. The generalized regression neural network is, a nonparametric method that estimates the joint probability density function

of x and y, given only a training set with no preconceptions about its form. Figure 7.7 shows the abstract structure of a generalized regression neural network with the input layer, the pattern layer (with the kernel regression function), the summation layer, and the output layer as a continuous variable.

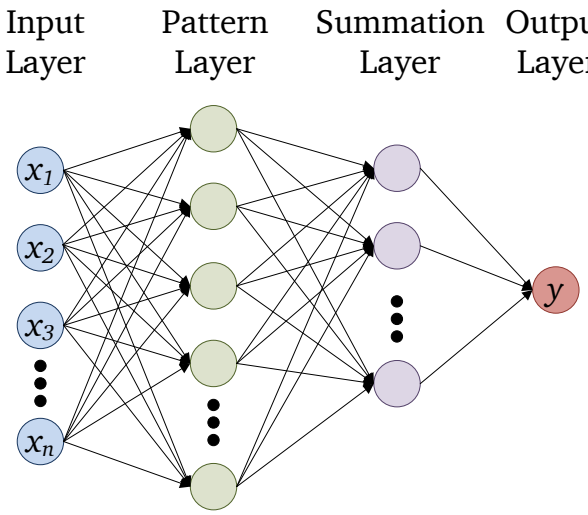



**Figure 7.7:** Schematic diagram of a probabilistic neural networks and a generalized regression neural network architecture. They have the same structure of the network, but use different kernel functions to archive a classification (probabilistic neural networks) or a regression (generalized regression neural network).

#### 7.2.4.3 Adaptive neuro-fuzzy inference system

The adaptive neuro-fuzzy inference system is a fuzzy Sugeno model of integration [92]. As described in Section 3.2.2, fuzzy sets are sets whose elements have degrees of membership. An adaptive neuro-fuzzy inference system optimizes a final fuzzy inference system, which uses fuzzy set theory [114] to map an input vector to an output, with the artificial neural network training discussed in Section 7.2.4. It maps the input vector by inputting membership functions and associated parameters, prior to mapping the normalized results to outputs by using output membership functions. An adaptive neuro-fuzzy inference system can be used for classification (see Section 6.1) and regression (see Section 6.2) problems.

The initial membership functions and rules for the fuzzy inference system can be designed by domain experts with knowledge of the real-world system (see Section 3.2.2). The adaptive neuro-fuzzy inference system refines the fuzzy IF-THEN-ELSE rules and membership functions, according to labeled historical data. [47] showed that even if domain expert knowledge is not available, it is possible to set up membership functions, and then employ the artificial neural networks training process on labeled historical

data, to generate a set of fuzzy IF-THEN ELSE rules. Two examples of fuzzy IF-THEN ELSE rules, based upon a first-order Sugeno model, show the adaptive neuro-fuzzy inference system architecture:

$$\text{Rule 1: IF } (x_1 \text{ is } A_1) \text{ and } (x_2 \text{ is } B_1) \text{ THEN } (y_1 = p_1 x_1 + q_1 x_2 + r_1),$$
$$\text{Rule 2: IF } (x_1 \text{ is } A_2) \text{ and } (x_2 \text{ is } B_2) \text{ THEN } (y_2 = p_2 x_1 + q_2 x_2 + r_2),$$

$x_1$ and $x_2$ represent the inputs feature vector and $A_i$ and $B_i$ are the fuzzy sets [114]. $y_1$ and $y_2$ are outputs, $p_i$, $q_i$ and $r_i$ are the parameters defined by the training process. The adaptive neuro-fuzzy inference system of these two rules is described in Figure 7.8, where the cycles are nodes with a fixed function, and the rectangle are adaptive nodes.

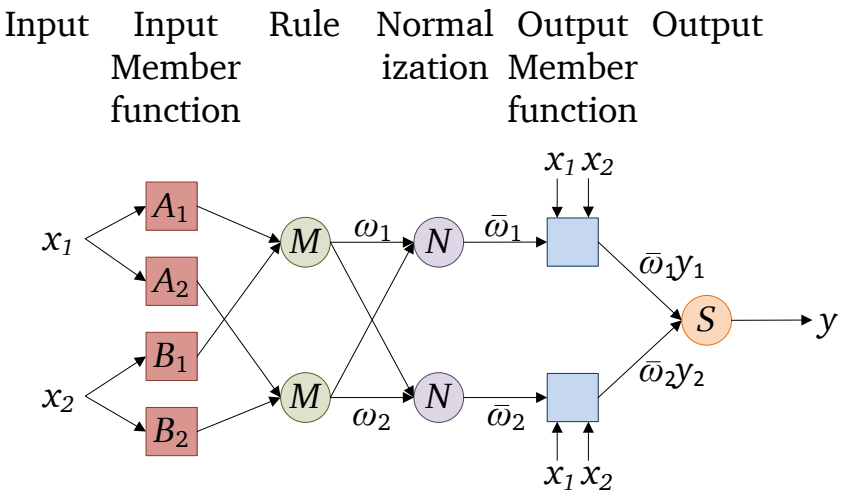



**Figure 7.8:** The adaptive neuro-fuzzy inference system for the given rules [57].

1. Input member function layer:
   The node in this layer is adaptive, and the layer performs fuzzification. This implies that the outputs of this layer represent the membership grade of the inputs.

2. Rule layer:
   These nodes are simple multipliers. The outputs of this layer represent the fuzzy strengths $\omega_i$.

3. Normalization layer:
   Normalize the weights $\omega_i$. The normalization factor can be calculated as the sum of all weight functions.

4. Output member function layer:
   The outputs of this layer are given by $\bar{\omega}_i y_i = p_i x_1 + q_i x_2 + r_i$, where $p_i$, $q_i$ and $r_i$ are the parameters of the output member function.

5. Output layer:
   This node performs the summation of all incoming signals $\bar{\omega}_i y_i$.

The adaptive neuro-fuzzy inference system can be useful for failure type detection and predictive maintenance, especially in combination with the knowledge of a domain expert. A good definition of the IF-THEN ELSE rules can significantly reduce the labeled historical data needed. The simple structure also allows the fast computation of results.

### 7.2.5 Support vector machine

A support vector machine is a nonparametric machine learning technique that is useful when the underlying process of the real-world system is not known, or the mathematical relation is too expensive to be obtained due to the increased influence by a number of interdependent factors. A support vector machine can be used for classification (see Section 6.1) and regression (see Section 6.2) problems.
Usually, the border between two classes is non-linear in the input vector space. In this situation, the multidimensional hyperplane has to be non-linear. The creation of a multidimensional and non-linear function which try to maximize the margin between different classes is a very hard task. A support vector machine uses a kernel function to map the input vector onto a higher dimensional space, where a linear hyperplane between different classes is possible, with a maximal margin between different classes to reduce misclassification. The location of the boundary is determined by a subset of historical data points, known as support vectors. A two-dimensional visualization is shown in Figure 7.9. In the same way as with classification, there is motivation to seek and optimize the bounds in the higher dimensional space given for regression [85].

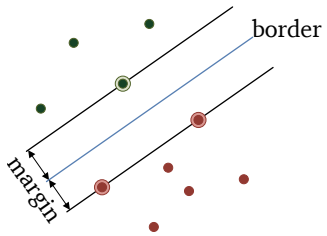



**Figure 7.9:** Visualization of a two-class support vector machine. The location of the boundary is determined by a subset of the data points, known as support vectors, which are indicated by circles.

A support vector machine cannot lead to a local minimum, because obtaining a support vector machine solution corresponds to dealing with a convex quadratic optimization problem [112]. Where the Karush-Kuhn-Tucker statements determine the necessary and sufficient conditions for a global optimum, and every local minimum is a global minimum [78], which can be an advantage when compared to approaches like back propagation neural networks (see Section 7.2.4). This is another difference to that of artificial neural networks, which are based on the empirical risk minimization that only minimizes the training errors. Support vector machines make use of the

structural risk minimization, seeking to minimize an upper bound on the error in the higher-dimensional space. Structural risk minimization approaches usually appear to be computationally less demanding, since the empirical risk minimization is accustomed to dealing with large sample sizes [100]. The artificial neural network approach does not look at the generalization performance leading to either underfitting or overfitting problems. The support vector machine with the maximal margin hyperplane focuses on a trade-off between model accuracy and model ability in predicting future values [28]. This characteristic tends to improve the SVM effectiveness to forecast upcoming outputs, which is a main requirement for a machine learning approach for failure type detection and predictive maintenance.

## 7.3 Summary

This chapter discussed the machine learning techniques found in recent literature relevant in the area of failure type detection and predictive maintenance. All the techniques described are distinguished into parametric and non parametric. Parametric machine learning algorithms make assumptions about the distribution of data. That is why parametric approaches are of low computational effort, and very fast. The accuracy of the distribution defines the accuracy of the result. This is the main problem for failure type detection and predictive maintenance scenarios, because these data are usually non-linear in the lifetime of a real-world systems, and therefore, it is hard to make assumptions about the distributions. Nonparametric approaches makes the assumption that inputs close to one another must have the same result. The difference between such techniques relates to the definition of "close to another".
Table 7.1 provides an overview of the approaches used in recent literature of failure type detection and predictive maintenance. These approaches are introduced, briefly discussing their advantage and the drawbacks.
Apart from other chapters, the summary does not contain a decision tree nor a guideline for selecting the right technique. In recent literature, empirical experiments forms the basis of decision making for a machine learning technique. Scenarios differ considerably from each other in terms of computational resources, memory resources, available data, and quality of data. As described in Chapter 5 the feature engineering, especially the feature extraction (see Section 5.2), and the data labeling (see Chapter 6), are not researched in terms of failure type detection and predictive maintenance scenarios. As long as this is not change, it is not possible, from an analytic point of view, to make a decision on a machine learning technique suitable for a given failure type detection and predictive maintenance problem. The reason is that the accuracy in these areas is too important and needs the correct analytic choice of a machine learning technique.

| Reference | Linear Discriminant Analysis | Bayesian Networks | Hidden semi-Markov Model | Belief Rule Base | Decision Tree Base Model | Random Forest | Nearest Neighbor | Back propagation neural network | Probabilistic neural network | Generalized regression neural network | Adaptive neuro-fuzzy inference system | Support Vector Machine |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| [20] | X | | | | | | | X | | | | X |
| [119] | | | | | | | | X | | | | |
| [116] | | | | | | | | X | | | | |
| [54] | | | | | | | | X | | | | |
| [103] | | | | | | | | X | | | | |
| [107] | | | | | | | | X | X | | | |
| [117] | | | | | | | | X | | | X | |
| [108] | | | | | | | | X | | X | | |
| [57] | | | | | | | | X | | | X | |
| [94] | | | | | | | | X | | | | |
| [111] | | | | | | | | X | | | | X |
| [102] | | | | | | | | | X | | | |
| [106] | | | | | | | | X | | X | | |
| [28] | | | | | | | | | | | | X |
| [41] | | | | | | | | | | | | X |
| [19] | | | | | X | | | | | | | X |
| [58] | | | | | X | | | | | | | X |
| [109] | | | | | | | | | | | | X |
| [30] | | | | | | | | | | | | X |
| [110] | | | | | | | | | | | | X |
| [9] | | | | | | | | | | | | X |
| [82] | | | | | | | | | | | | X |
| [67] | | | | X | X | X | X | | | | | |
| [96] | | | | | | | X | | | | | |
| [40] | | | | | | | X | | | | | |
| [56] | | X | | | | | | | | | | |
| [31] | | | X | | | | | | | | | |

| Reference | Linear Discriminant Analysis | Bayesian Networks | Hidden semi-Markov Model | Belief Rule Base | Decision Tree Base Model | Random Forest | Nearest Neighbor | Back propagation neural network | Probabilistic neural network | Generalized regression neural network | Adaptive neuro-fuzzy inference system | Support Vector Machine |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| [60] | | | X | | | | | | | | | |
| [95] | | | X | | | | | | | | | |
| [64] | | | | X | | | | | | | | |
| [115] | | | | X | X | | | | | | | |
| [42] | | | | | X | | | | | | | |
| **Sum** | **1** | **1** | **3** | **3** | **5** | **1** | **3** | **12** | **2** | **2** | **2** | **11** |

**Table 7.1:** Usage of machine learning techniques in recent literature

## 8 Evaluation Strategies

An evaluation strategy is needed for analyzing a failure type detection and predictive maintenance approach. There are many common ways to check the quality of a classification:

- True positive rate
- True negative rate
- False positive rate
- False negative rate
- Accuracy
- Precision
- Recall
- F-measure
- Mean square error
- Mean absolute percentage error

These evaluation techniques are about to true positives, false positives, true negatives and false negatives. These parameters are the result of tests with the historical data and the failure type detection and predictive maintenance approach. For example, accuracy is defined as:

$$accuracy = \frac{\sum tp + \sum tn}{\sum tp + \sum fp + \sum tn + \sum fn} \tag{36}$$

Where $tp$ are the true positives, $tn$ are the true negatives, $fp$ are the false positives and $fn$ are the false negatives.
In recent literature, the most commonly used metrics in failure type detection and predictive maintenance are accuracy, precision, mean square error and mean absolute percentage error [77]. This assertion is also valid for failure type detection and predictive maintenance applications.
In failure type detection and predictive maintenance scenarios, the requirements for an evaluation metric are different from those of a common classification problem. The main reasons are:

- Acausality:
  Some failures in a real-world system are difficult to predict, because the knowledge about the future operation modes of environmental conditions is not controllable.

- Failure data from real applications:
  Data on faults in a real-world system is very rare, because faults are generally expensive and a safety risk. To avoid this, usually a maintenance operation has already been conducted to avert any failure. Two other problems arise with maintenance operations, preventing the verification whether the prediction was correct. After a maintenance action, the state of the influenced component is changed.

- Off-line performance evaluation:
  For obtaining missing failure data, experiments are sometimes conducted in a controlled run-to-failure scenario. However, these experiments may not have the same conditions as the real execution, and therefore, the data may not be precise. Such an experiment can be also very expensive, particularly when many failure types are to be detected.

Another problem for all machine learning algorithms is the uncertainty in prognostics. A good machine learning technique, and especially a failure type detection and predictive maintenance system, should provide accurate and precise estimations. However, it is also important to obtain information from such a system about the reliability of the prediction.
Predictions are made early on, with less information about the dynamics of fault evolution, and are required to foresee further into time, which makes the prediction task more difficult, compared to assessing at a later stage. This characteristics usually not a problem, although the performance of failure type detection and predictive maintenance applications tends to be more critical as time passes by, and the system reaches End of Life. Approaches which try to predict the state of the real-world system n-states ahead (fixed horizon) have better performance of the metrics mentioned above than approaches with a residual useful life estimation (moving horizon) that say nothing about the accuracy close to the end of its life, which is most important.
[76] introduces a new metrics of prognostic performance for estimating residual useful lifetime. This metrics is more compliant to the requirements of a failure type detection and predictive maintenance system. The metrics has a hierarchical design Figure 8.1 and will be discussed below.
With the presented metrics, it is possible to identify the weak spots of a model, which are important for failure type detection and predictive maintenance approaches. Possible weak spots that can be detected are:

- Inadequate real-world system models:
  Real-world systems often exhibit inherent transient events in their life cycles. The prognosis model may not have recognized such transient events, which may change the behavior of the model.

- Operational transient:
  Another reason for poor results of a metric might be a change in operational conditions, which are not detected in historical data.

- Uncertainties in prognostic environments:
  In the historical learning data, usually not all conditions are detected, which can lead to an inexact prognosis.

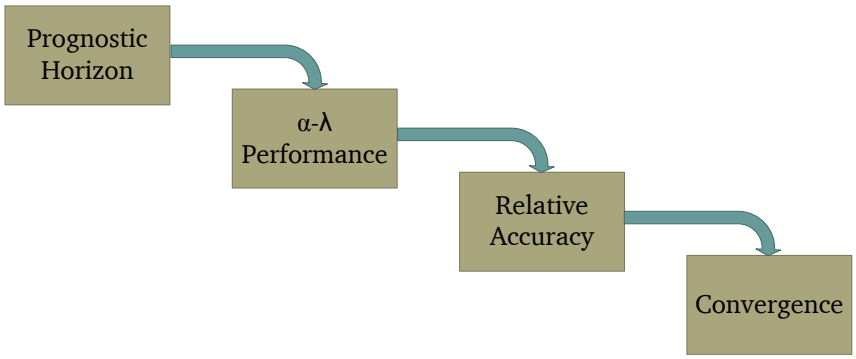



**Figure 8.1:** Hierarchical design of the prognostics metrics [76].

## 8.1 Prognostic Horizon

The prediction horizon is the difference between the time $i_{\alpha\lambda}$ when the predictions first meet a specified performance criteria, and the time index for the End of Life, $EoL$. The prediction horizon ($PH$) can expressed as:

$$PH = t_{EoL} - t_{i_{\alpha\lambda}} \tag{37}$$

Figure 8.2 depicts an example of two different prognostics and the prognostic horizon. The black line shows the residual useful lifetime, which is degrading and linear with increasing time. The gray lines represent the border of the performance criteria. The red and the blue lines represent the results of the predictions of two different failure type detection and predictive maintenance systems. The red and blue points represent new predictions on grounds of new data available from the real-world system. The prediction horizon is the time between the End of Life (EoL) and the time when the prediction of the failure type detection and predictive maintenance system for the first time fulfills the performance criteria (prediction is between the black and the grey line). That implies, that $PH^2$ is the prediction horizon of the blue failure type detection and predictive maintenance system and $PH^1$ is the prediction horizon of the red failure type detection and predictive maintenance system.

## 8.2 $\alpha$-$\lambda$ Performance

The $\alpha$-$\lambda$ accuracy is defined as a binary metric, which evaluates the accuracy and confidence at a time $t_\lambda$:

$$\alpha - \lambda\, accuracy = \begin{cases} 1 & \text{if} \quad \pi\left[r\left(i_\lambda\right)\right]_{-\alpha}^{+\alpha} \geq \beta \\ 0 & \text{otherwise} \end{cases} \tag{38}$$

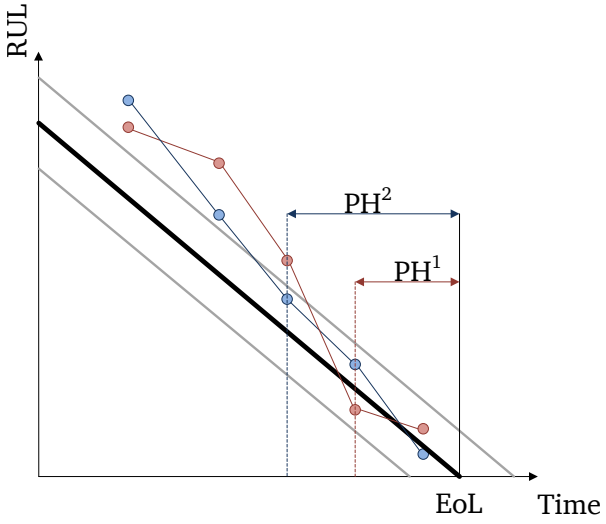



**Figure 8.2:** Example of prognostic horizon of two different failure type detection and predictive maintenance systems (red and blue lines). The black line is the component residual useful lifetime (marked on the axis as RUL). The gray lines represent the performance criteria [76].

where $\beta$ is a threshold and $\pi\left[r\left(i_\lambda\right)\right]_{-\alpha}^{+\alpha}$ is the probability mass of the prediction probability density function within the bounds from $+\alpha$ to $-\alpha$. In this case, the probability mass of the prediction probability density function ($\pi\left[r\left(i_\lambda\right)\right]$) represents the confidence of the prediction and the $\alpha$ bounds the error margins of the prediction. That means the smaller the $\alpha$ bounds the higher accuracy is achieved. The prediction needs to have greater or equal than $\beta$ of probability mass within the $\alpha$ bounds.

## 8.3 Relative Accuracy

The $\alpha$-$\lambda$ performance provides information when predictions fall within an accuracy level. The relative accuracy gives information about the accuracy at a given time, without the accuracy level. It is needed because in a failure type detection and predictive maintenance system, the accuracy of the prediction needs to increase with time to the End of Life of a component. Therefore, it is helpful to distinguish between estimations at the beginning or the ending of the lifetime of a component. The relative accuracy can be expressed as:

$$relative\ accuracy = 1 - \frac{|r_*(i_\lambda) - \langle r(i_\lambda)\rangle|}{r_*(i_\lambda)} \tag{39}$$

where $i_\lambda$ is the time index, $r_*(i_\lambda)$ are the ground truth residual useful lifetime at $i_\lambda$, and $\langle r(i_\lambda)\rangle$ is the predicted residual useful lifetime at $i_\lambda$.

## 8.4 Convergence

The convergence defines the distance between the origin and the prediction result, and can be expressed as

$$convergence = \sqrt{\left(x_p - x_o\right)^2 + \left(y_p - y_o\right)^2} \tag{40}$$

Where $\left(x_p,\ y_p\right)$ are the coordinates of the prediction and $(x_o,\ y_o)$ are the origin coordinates, where the $x$-axis is the lifetime and the $y$-axis is the residual useful lifetime. This implies that at the End of Life, the origin residual useful lifetime is $y_o = 0$.

## 8.5 Summary

The standard metrics for machine learning systems mentioned in Chapter 8 are not efficient enough to asses the performance of a machine learning systems for failure type detection and predictive maintenance. Unfortunately, most failure type detection and predictive maintenance systems mentioned in recent literature are measured in their performance with the standard metrics. This leads to difficulties comparing that systems to each other.
Figure 8.1 shows four metrics that can be used in a hierarchical manner. The Prognostic Horizon identifies when a failure type detection and predictive maintenance system first enters the specified error margin around the actual end-of-life. This gives information about how much time is allowed for any corrective action to be taken. The second metric $\alpha$-$\lambda$ performance (see Section 8.2) continues to identify whether the algorithm performs within desired error margins (specified by the parameter $\alpha$) of the actual residual useful lifetime for a given time instant (specified by the parameter $\lambda$). The third step calculates the accuracy level relative to the actual residual useful lifetime (see Section 8.3). This metric assumes that prognostic performance improves with a decreasing residual useful lifetime, because more information becomes available with time. A failure type detection and predictive maintenance model will satisfy these metrics criteria when it converges with the true residual useful lifetime. The fourth metric, Convergence (see Section 8.4), quantifies how fast the failure type detection and predictive maintenance system converges when it satisfied all previous metrics. By applying these four metrics, an assumption of the quality of a failure type detection and predictive maintenance system is much more descriptive than using a standard metric, and a comparison between different failure type detection and predictive maintenance systems using the four introduced metrics becomes more sensible.

# 9 Conclusion and Future Work

Different types and the elements of failure type detection and predictive maintenance approach are discussed in this thesis. Different models are introduced, focusing on the data driven models and their different learning techniques.
The integral parts of a data-driven model are presented in respect to recent research results in all areas. This thesis shows that the data collected from the real-world system by the data acquisition technique strongly influences further decisions of techniques to use for feature engineering, data labeling and machine learning techniques (see Chapter 4). One significant issue is that feature extraction and data labeling are currently not researched for failure type detection and predictive maintenance, and therefore, it is not possible to make assumptions about the quality of features and data labeling in failure type detection and predictive maintenance.Considering that, it is impossible to decide on an analytic process for choosing a machine learning technique. The same applies to the conditions, as long as these topics are not adequately researched.
The data gathered by data acquisition systems usually has to be modified to obtain more detailed information about the state of the real-world system; this area is called feature engineering. It is shown that, especially for sensor data, signal processing techniques are essential to convert information and, in some cases, extract more detailed information (see Section 5.1.3.3 and Section 5.1.3.4). The decision about the signal processing technique depends on the requirements of the information needed. Furthermore, the differences between feature extraction and feature selection are discussed.
Before a failure type detection and predictive maintenance approach can analyze current data, it is necessary to learn from historic data, which has to be labeled with the expected output. To achieve this, two different approaches are discussed, the State Driven Data Labeling (see Section 6.1) and the Residual Useful Lifetime labeling technique (see Section 6.2).
The decision regarding the selection of a data labeling technique is important for finding a suitable machine learning technique. Independent from the data labeling technique, Table 7.1 shows that in recent literature, nonparametric machine learning approaches are used. In particular, artificial neural networks and support vector machines have been used for approaches to recent failure type detection and predictive maintenance. All related papers choose a machine learning technique by empirical evaluation. Possible reasons for using this techniques are: the research community using these techniques has been active in recent years and the techniques that are propagated are very flexible for different problems.
The evaluation metrics to rate machine learning techniques used in recent literature are not suitable to quantify the performance of these techniques in the context of failure type detection and predictive maintenance. This thesis shows evaluation techniques that are better adapted to evaluate the results and indicate possible defects.

## Future Work

Some future work in different areas is preferable. Currently, there is no approach for the feature extraction of physical properties with regard to failure type detection. Such a technique must convert the data given into the physical properties of a component. There is also no known feature selection technique designed for failure type detection and predictive maintenance, in which the features, with the most physical relevance. With respect to upcoming failures, are selected. Investigations into the requirements of a feature selection technique, and how to implement them, could be another area of future work.
Surprisingly, recent literature only mentions two labeling techniques. As labeling is essential for supervised learning, it seems that there is a lot of potential for novel labeling techniques. In the area of data labeling, possible future work could thus involve exploring a labeling system that correlates with the physical degradation of a component. Such an approach could improve the prognostic performance, particularly in defining a prognostic horizon for confidential failure type detection. Moreover, with such an approach, it could also be helpful to realize a multi-failure type detection system, in which the failures are related to each other and distributed over multiple components. Together with an investigation into the area of feature engineering, an assumption about the quality of data and an analytic decision process for a machine learning technique is possible.
Components of a real-world system can usually be distributed. Therefore, designing a distributed machine learning approach could be interesting. Such an approach would offer the advantage that large amounts of data could be analyzed in parallel, which results a shorter time to reach a decision. Moreover, the influence of components upon one another could show the correlation of classes.
As discussed in Chapter 8, data is usually rare for faults in real-world systems. In the future, a learning algorithm can be combined with a supervised learning algorithm for determining an acceptable state. When the transition from an acceptable state occurs, a maintenance point has to be scheduled, whereby the type of failure should be detected and learned by the system, comparable to what a reinforcement learning strategy does. Accordingly, this could relieve the problem concerning the absence of historical failure data.

## References


[1] Jürgen Ackermann, Andrew Bartlett, D Kaesbauer, W Sienel, and R Steinhauser. *Robust control: the parameter space approach*. Springer Science & Business Media, 1993.

[2] Charu C. Aggarwal. *Managing and mining sensor data*. Springer Science & Business Media, 2013.

[3] Charu C. Aggarwal. *Data Classification: Algorithms and Applications*. CRC Press, 2014.

[4] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *Database Theory - ICDT 2001*, volume 1973 of *Lecture Notes in Computer Science*, pages 420–434. Springer Berlin Heidelberg, 2001.

[5] David W. Aha. Lazy learning. In *Lazy Learning*, pages 7–10. Kluwer Academic Publishers, 1997.

[6] Ethem Alpaydin. *Maschinelles Lernen*. Oldenbourg Verlag, 2008.

[7] Howard Austerlitz. *Data acquisition techniques using PCs*. Academic press, 2002.

[8] Roberto Baldoni, Mariangela Contenti, and Antonino Virgillito. The evolution of publish/subscribe communication systems. In *Future Directions in Distributed Computing*, volume 2584 of *Lecture Notes in Computer Science*, pages 137–141. Springer Berlin Heidelberg, 2003.

[9] Tribeni Prasad Banerjee and Swagatam Das. Multi-sensor data fusion using support vector machine for motor fault detection. *Information Sciences*, 217:96–107, 2012.

[10] Eric Bechhoefer and Michael Kingsley. A review of time synchronous average algorithms. In *Annual conference of the prognostics and health management society*, volume 23, page 24, 2009.

[11] Tatiana Biagetti and Enrico Sciubba. Automatic diagnostics and prognostics of energy conversion processes via knowledge-based systems. *Energy*, 29(12):2553–2572, 2004.

[12] Christopher M. Bishop. *Pattern recognition and machine learning*, volume 4. Springer Science & Business Media, 2006.

[13] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[14] Tom Brotherton, Gsry Jahns, Jerry Jacobs, and Dariusz Wroblewski. Prognosis of faults in gas turbine engines. In *Aerospace Conference Proceedings*, volume 6, pages 163–171, 2000.

[15] Alejandro Buchmann and Boris Koldehofe. Complex event processing. *it-Information Technology Methoden und innovative Anwendungen der Informatik und Informationstechnik*, 51(5):241–242, 2009.

[16] Tomasz Bujlow, Tahir Riaz, and Jens Myrup Pedersen. A method for classification of network traffic based on c5.0 machine learning algorithm. In *Computing, Networking and Communications (ICNC)*, pages 237–241, 2012.

[17] Carey Bunks, Dan McCarthy, and Tarik Al-Ani. Condition-based maintenance of machines using hidden markov models. *Mechanical Systems and Signal Processing*, 14(4):597–612, 2000.

[18] Karen L. Butler. An expert system based framework for an incipient failure detection and predictive maintenance system. In *International Conference on Intelligent Systems Applications to Power Systems (ISAP)*, pages 321–326, 1996.

[19] Matthew Butler and Vlado Kešelj. Data mining techniques for proactive fault diagnostics of electronic gaming machines. In *Advances in Artificial Intelligence*, volume 6085 of *Lecture Notes in Computer Science*, pages 366–369. Springer Berlin Heidelberg, 2010.

[20] Kai-Ying Chen, Long-Sheng Chen, Mu-Chen Chen, and Chia-Lung Lee. Using svm based method for equipment fault detection in a thermal power plant. *Computers in Industry*, 62(1):42 – 50, 2011.

[21] Yu-Wang Chen, Jian-Bo Yang, and Dong-Ling Xu. Uncertain nonlinear system modeling and identification using belief rule-based systems. In *Integrated Uncertainty in Knowledge Modelling and Decision Making*, volume 8032 of *Lecture Notes in Computer Science*, pages 209–218. Springer Berlin Heidelberg, 2013.

[22] Seong Soo Choi, Ki Sig Kang, Han Gon Kim, and Soon Heung Chang. Development of an on-line fuzzy expert system for integrated alarm processing in nuclear power plants. *IEEE Transactions on Nuclear Science*, 42(4):1406–1418, 1995.

[23] Alice Chuang. Time series analysis: Univariate and multivariate methods. *Technometrics*, 33(1):108–109, 1991.

[24] Leon Cohen. Time-frequency distributions-a review. *Proceedings of the IEEE*, 77(7):941–981, 1989.

[25] Madalena Costa, Ary L Goldberger, and C-K Peng. Multiscale entropy analysis of biological signals. *Physical review E*, 71(2):021906, 2005.

[26] Michele M. Covell and John M. Richardson. A new, efficient structure for the short-time fourier transform, with an application in code-division sonar imaging. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 3, pages 2041–2044, 1991.

[27] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and image processing*, 14(3):227–248, 1980.

[28] Márcio das Chagas Moura, Enrico Zio, Isis Didier Lins, and Enrique Droguett. Failure and reliability prediction by support vector machines regression of time series data. *Reliability Engineering & System Safety*, 96(11):1527–1534, 2011.

[29] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 41(7):909–996, 1988.

[30] Feng Ding, Zhengjia He, Yanyang Zi, Xuefeng Chen, Jiyong Tan, Hongrui Cao, and Huaxin Chen. Application of support vector machine for equipment reliability forecasting. In *IEEE International Conference on Industrial Informatics*, pages 526–530, 2008.

[31] Ming Dong and David He. Hidden semi-markov model-based methodology for multi-sensor equipment health diagnosis and prognosis. *European Journal of Operational Research*, 178(3):858–878, 2007.

[32] Pierre Duhamel and Martin Vetterli. Fast fourier transforms: a tutorial review and a state of the art. *Signal processing*, 19(4):259–299, 1990.

[33] Rick Durrett. *Probability: theory and examples*. Cambridge university press, 2010.

[34] Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The many faces of publish/subscribe. *ACM Computing Surveys (CSUR)*, 35(2):114–131, 2003.

[35] Otto Föllinger and Mathias Kluwe. *Laplace-und Fourier-Transformation*. Elitera Berlin, 1977.

[36] Carl Frelicot. A fuzzy-based pronostic adaptive system. *Journal européen des systèmes automatisés*, 30(2-3):281–299, 1996.

[37] Peng Guo and Nan Bai. Wind turbine gearbox condition monitoring with aakr and moving window statistic methods. *Energies*, 4(11):2077–2093, 2011.

[38] Joseph F. Hair, William C. Black, Barry J. Babin, Rolph E. Anderson, and Ronald L. Tatham. *Multivariate data analysis*, volume 6. Pearson Prentice Hall Upper Saddle River, 2006.

[39] Hash M. Hashemian and Wendell C. Bean. State-of-the-art predictive maintenance techniques. *Instrumentation and Measurement*, 60(10):3480–3492, 2011.

[40] David He, Ruoyu Li, and Eric Bechhoefer. Split torque type gearbox fault detection using acoustic emission and vibration sensors. In *International Conference on Networking, Sensing and Control (ICNSC)*, pages 62–66, 2010.

[41] Kuanfang He and Xuejun Li. A quantitative estimation technique for welding quality using local mean decomposition and support vector machine. *Journal of Intelligent Manufacturing*, pages 1–9, 2014.

[42] Shu-Guang He, Zhen He, and GangA. Wang. Online monitoring and fault identification of mean shifts in bivariate processes using decision tree learning techniques. *Journal of Intelligent Manufacturing*, 24(1):25–34, 2013.

[43] Tin Kam Ho. Random decision forests. In *International Conference on Document Analysis and Recognition*, volume 1.1, pages 278–282, 1995.

[44] Bin Hu, Chee Khiang Pang, Ming Luo, Xiang Li, and Hian Leng Chan. A two-stage equipment predictive maintenance framework for high-performance manufacturing systems. In *Industrial Electronics and Applications (ICIEA)*, pages 1343–1348, 2012.

[45] Norden E Huang, Zheng Shen, Steven R Long, Manli C Wu, Hsing H Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 454, pages 903–995, 1998.

[46] Rolf Isermann and Marco Münchhof. *Identification of dynamic systems: an introduction with applications*. Springer Science & Business Media, 2010.

[47] J.-S.R. Jang. Anfis: adaptive-network-based fuzzy inference system. *Systems, Man and Cybernetics, IEEE Transactions on*, 23(3):665–685, May 1993.

[48] Andrew KS Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical systems and signal processing*, 20(7):1483–1510, 2006.

[49] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *European Conference on Computer Vision (ECCV)*, volume 5302 of *Lecture Notes in Computer Science*, pages 304–317. Springer Berlin Heidelberg, 2008.

[50] Shubha Kadambe and G. Faye Boudreaux-Bartels. A comparison of the existence of 'cross terms' in the wigner distribution and the squared magnitude of the wavelet transform and the short-time fourier transform. *IEEE Transactions on Signal Processing*, 40(10):2498–2517, 1992.

[51] Kevin A. Kaiser and Nagi Z. Gebraeel. Predictive maintenance management using sensor-based degradation models. *Systems, Man and Cybernetics, Part A: Systems and Humans*, 39(4):840–849, 2009.

[52] Karl-Dirk Kammeyer and Kristian Kroschel. *Digitale Signalverarbeitung*, volume 5. BG Teubner Verlag, 1998.

[53] Panagiotis Karampelas, Vassiliki Vita, Christos Pavlatos, Valeri M. Mladenov, and Lambros Ekonomou. Design of artificial neural network models for the prediction of the hellenic energy consumption. In *Symposium on Neural Network Applications in Electrical Engineering (NEUREL)*, pages 41–44, 2010.

[54] Pooria Karimi and Hooshang Jazayeri-Rad. Comparing the fault diagnosis performances of single neural networks and two ensemble neural networks based on the boosting methods. *Journal of Automation and Control*, 2(1):21–32, 2014.

[55] Uwe Kiencke, Michael Schwarz, and Thomas Weickert. *Signalverarbeitung: Zeitfrequenz-analyse und Schätzverfahren*. Oldenbourg Verlag, 2008.

[56] Renata Klein, Eduard Rudyk, and Eyal Masad. Decision and fusion for diagnostics of mechanical components. In *Annual Conference of the Prognostics and Health Management Society*, 2011.

[57] Yaguo Lei, Zhengjia He, and Yanyang Zi. A new approach to intelligent fault diagnosis of rotating machinery. *Expert Systems with Applications*, 35(4):1593–1600, 2008.

[58] Hongfei Li, Dhaivat Parikh, Qing He, Buyue Qian, Zhiguo Li, Dongping Fang, and Arun Hampapur. Improving rail network velocity: A machine learning approach to predictive maintenance. *Transportation Research Part C: Emerging Technologies*, 45:17–26, 2014.

[59] Ming Li. Fractal time series-a tutorial review. *Mathematical Problems in Engineering*, 2010, 2009.

[60] Qinming Liu and Ming Dong. Online health management for complex nonlinear systems based on hidden semi-markov model using sequential monte carlo methods. *Mathematical Problems in Engineering*, 2012.

[61] Tao Liu, Zheng Chen, Benyu Zhang, Wei-Ying Ma, and Gongyi Wu. Improving text classification using local latent semantic indexing. In *IEEE International Conference on Data Mining (ICDM).*, pages 162–169, 2004.

[62] Igor Loboda, Sergiy Yepifanov, and Yakov Feldshteyn. A more realistic scheme of deviation error representation for gas turbine diagnostics. *International Journal of Turbo & Jet-Engines*, 30(2):179–189, 2013.

[63] Stéphane G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, Jul 1989.

[64] Florent Martin, Nicolas Meger, Sylvie Galichet, and Nicolas Becourt. Data-driven prognosis applied to complex vacuum pumping systems. In *Trends in Applied Intelligent Systems*, volume 6096 of *Lecture Notes in Computer Science*, pages 468–477. Springer Berlin Heidelberg, 2010.

[65] Parikshit Mehta, Andrew Werner, and Laine Mears. Condition based maintenance-systems integration and intelligence using bayesian classification and sensor fusion. *Journal of Intelligent Manufacturing*, pages 1–16, 2013.

[66] Hrushikesh N. Mhaskar and Charles A. Micchelli. Approximation by superposition of sigmoidal and radial basis functions. *Advances in Applied mathematics*, 13(3):350–373, 1992.

[67] Sławomir Nowaczyk, Rune Prytz, Thorsteinn Rögnvaldsson, and Stefan Byttner. Towards a machine learning algorithm for predicting truck compressor failures using logged vehicle data. In *Scandinavian Conference on Artificial Intelligence*, pages 205–214, 2013.

[68] Benjamin E. Olivares, Matias A. Cerda Munoz, Marcos E. Orchard, and Jorge F. Silva. Particle-filtering-based prognosis framework for energy storage devices with a statistical characterization of state-of-health regeneration phenomena. *IEEE Transactions on Instrumentation and Measurement*, 62(2):364–376, 2013.

[69] Charles H. Oppenheimer and Kenneth A. Loparo. Physically based diagnosis and prognosis of cracked rotor shafts. In *AeroSense 2002*, pages 122–132. International Society for Optics and Photonics, 2002.

[70] Ying Peng, Ming Dong, and MingJian Zuo. Current status of machine prognostics in condition-based maintenance: a review. *The International Journal of Advanced Manufacturing Technology*, 50(1-4):297–313, 2010.

[71] J. Ross Quinlan. Improved use of continuous attributes in c4. 5. *Journal of artificial intelligence research*, pages 77–90, 1996.

[72] Lawerence Rabiner and Biing-hwang Hwang Juang. An introduction to hidden markov models. *IEEE Acoustics, Speech, and Signal Processing (ASSP) Magazine*, 3(1):4–16, 1986.

[73] Wolfgang Reinelt and Lennart Ljung. Robust control of identified models with mixed parametric and non-parametric uncertainties. *European journal of control*, 9(4):373–380, 2003.

[74] Joshua S. Richman and J. Randall Moorman. Physiological time-series analysis using approximate entropy and sample entropy. *American Journal of Physiology-Heart and Circulatory Physiology*, 278(6):H2039–H2049, 2000.

[75] Said E. Said and David A. Dickey. Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, 71(3):599–607, 1984.

[76] Abhinav Saxena, Jose Celaya, Bhaskar Saha, Sankalita Saha, and Kai Goebel. Metrics for Offline Evaluation of Prognostic Performance. *International Journal of Prognostics and Health Management (IJPHM)*, 1:1–20, 2010.

[77] Abhinav Saxena, José R. Celaya, Bhaskar Saha, Sankalita Saha, and Kai Goebel. Evaluating algorithm performance metrics tailored for prognostics. In *IEEE Aerospace conference*, pages 1–13, 2009.

[78] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2002.

[79] Mark Schwabacher and Kai Goebel. A survey of artificial intelligence for prognostics. In *Aaai Fall Symposium*, pages 107–114, 2007.

[80] Claude E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[81] Robert C. Sharpley and Vesselin Vatchev. Analysis of the intrinsic mode functions. *Constructive Approximation*, 24(1):17–47, 2006.

[82] Nasar Aldian Ambark Shashoa, Goran Kvaščev, Aleksandra Marjanović, and Željko Djurović. Sensor fault detection and isolation in a thermal power plant steam separator. *Control Engineering Practice*, 21(7):908–916, 2013.

[83] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, Mao-Yin Chen, and Dong-Hua Zhou. A wiener-process-based degradation model with a recursive filter algorithm for remaining useful life estimation. *Mechanical Systems and Signal Processing*, 35(1):219–237, 2013.

[84] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation–a review on the statistical data driven approaches. *European Journal of Operational Research*, 213(1):1–14, 2011.

[85] Alexander J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[86] Donald F. Specht. Probabilistic neural networks. *Neural networks*, 3(1):109–118, 1990.

[87] Donald F. Specht. A general regression neural network. *IEEE Transactions on Neural Networks*, 2(6):568–576, 1991.

[88] Richard S. Sutton and Andrew G. Barto. *Introduction to reinforcement learning*. MIT Press, 1998.

[89] Roman W. Swiniarski and Larry Hargis. Rough sets as a front end of neural-networks texture classifiers. *Neurocomputing*, 36(1):85–102, 2001.

[90] Roman W Swiniarski and Andrzej Skowron. Rough set methods in feature selection and recognition. *Pattern recognition letters*, 24(6):833–849, 2003.

[91] Li Tan and Jean Jiang. *Digital signal processing: fundamentals and applications*. Academic Press, 2013.

[92] Kazuo Tanaka and Michio Sugeno. Stability analysis and design of fuzzy control systems. *Fuzzy sets and systems*, 45(2):135–156, 1992.

[93] Gerald Tesauro. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural computation*, 6(2):215–219, 1994.

[94] Zhigang Tian. An artificial neural network method for remaining useful life prediction of equipment subject to condition monitoring. *Journal of Intelligent Manufacturing*, 23(2):227–237, 2012.

[95] Diego Alejandro Tobon-Mejia, Kamal Medjaher, Noureddine Zerhouni, and Gerard Tripot. A data-driven failure prognostics method based on mixture of gaussians hidden markov models. *IEEE Transactions on Reliability*, 61(2):491–503, 2012.

[96] Houari Toubakh, Moamar Sayed-Mouchaweh, and Eric Duviella. Advanced pattern recognition approach for fault diagnosis of wind turbines. In *International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 368–373, 2013.

[97] Thomas Trappenberg. *Fundamentals of computational neuroscience*. Oxford University Press, 2009.

[98] Paul Y. L. Tu, Richard Yam, Peter W. Tse, and A. O. W. Sun. An integrated maintenance management system for an advanced manufacturing company. *The International Journal of Advanced Manufacturing Technology*, 17(9):692–703, 2001.

[99] Chris Van Hoof, Kris Baert, and Ann Witvrouw. The best materials for tiny, clever sensors. *Science*, 306(5698):986–987, 2004.

[100] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2000.

[101] Saeed V. Vaseghi. *Advanced digital signal processing and noise reduction*. John Wiley & Sons, 2008.

[102] Chengdong Wang, Youyun Zhang, and Zhenyuan Zhong. Fault diagnosis for diesel valve trains based on time–frequency images. *Mechanical Systems and Signal Processing*, 22(8):1981–1993, 2008.

[103] Chun-Chieh Wang, Yuan Kang, Ping-Chen Shen, Yeon-Pun Chang, and Yu-Liang Chung. Applications of fault diagnosis in rotating machinery by using time series analysis with neural network. *Expert Systems with Applications*, 37(2):1696–1702, 2010.

[104] H. Joseph Weaver. *Applications of discrete and continuous Fourier analysis*. Wiley-Interscience, 1983.

[105] Peter Welch. The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. *IEEE Transactions on audio and electroacoustics*, pages 70–73, 1967.

[106] Jian-Da Wu and Cheng-Kai Huang. An engine fault diagnosis system using intake manifold pressure signal and wigner–ville distribution technique. *Expert Systems with Applications*, 38(1):536–544, 2011.

[107] Jian-Da Wu and Shu-Yi Liao. A self-adaptive data analysis for fault diagnosis of an automotive air-conditioner blower. *Expert Systems with Applications*, 38(1):545–552, 2011.

[108] Jian-Da Wu and Chiu-Hong Liu. An expert system for fault diagnosis in internal combustion engines using wavelet packet transform and neural network. *Expert systems with applications*, 36(3):4278–4286, 2009.

[109] Tian Yau Wu, Jiuping Chen, and Charles Xiaoxue Wang. Characterization of gear faults in variable rotating speed using hilbert-huang transform and instantaneous dimensionless frequency normalization. *Mechanical Systems and Signal Processing*, 30:103–122, 2012.

[110] Wen-An Yang. Simultaneous monitoring of mean vector and covariance matrix shifts in bivariate manufacturing processes using hybrid ensemble learning-based model. *Journal of Intelligent Manufacturing*, pages 1–30, 2014.

[111] Wen-An Yang and Wei Zhou. Autoregressive coefficient-invariant control chart pattern recognition in autocorrelated manufacturing processes using neural network ensemble. *Journal of Intelligent Manufacturing*, pages 1–20, 2013.

[112] Yinyu Ye. Approximating global quadratic optimization with convex quadratic constraints. *Journal of Global Optimization*, 15(1):1–17, 1999.

[113] Shun-Zheng Yu. Hidden semi-markov models. *Artificial Intelligence*, 174(2):215–243, 2010.

[114] Lotfi A. Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

[115] Marvin Zaluski, Sylvain Létourneau, Jeff Bird, and Chunsheng Yang. Developing data mining-based prognostic models for cf-18 aircraft. *Journal of Engineering for Gas Turbines and Power*, 133(10):101601, 2011.

[116] Jafar Zarei, Mohammad Amin Tajeddini, and Hamid Reza Karimi. Vibration analysis for bearing fault detection and classification using an intelligent filter. *Mechatronics*, 24(2):151–157, 2014.

[117] Long Zhang, Guoliang Xiong, Hesheng Liu, Huijun Zou, and Weizhong Guo. Bearing fault diagnosis using multi-scale entropy and adaptive neuro-fuzzy inference. *Expert Systems with Applications*, 37(8):6077–6085, 2010.

[118] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007.

[119] Zhenyou Zhang, Yi Wang, and Kesheng Wang. Fault diagnosis and prognosis using wavelet packet decomposition, fourier transform and artificial neural network. *Journal of Intelligent Manufacturing*, 24(6):1213–1227, 2013.

[120] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC Press, 2012.

[121] Zhi-Jie Zhou, Chang-Hua Hu, Bang-Cheng Zhang, Dong ling Xu, and Yu wang Chen. Hidden behavior prediction of complex systems based on hybrid information. *IEEE Transactions on Cybernetics*, 43(2):402–411, 2013.