People's Democratic Republic of Algeria Ministry of Higher Education and Scientific Research

Ammar Telidji University – Laghouat – Faculty of Technology

Department of Mechanics

Graduation Thesis:

Master's Degree in Industrial Maintenance

Option:

Industrial Maintenance

Presented by:
Dali Larbi
Ziani Abdelkader

Thesis:

Machine Learning and Predictive maintenance: vibration monitoring of rotating machinery

Defended in front of the jury composed of:

Name and Surname	Grade	Quality
k.Rayane	Professor	Supervisor
A.hammou	Professor	President
a.rezig	Professor	Examiner
I.ameur	Professor	Examiner

Academic Year: 2023/2024

Acknowledgement

I am filled with gratitude as I reflect on my graduation. This accomplishment would not have been possible without the unwavering support of so many wonderful people in my life.

First, I would like to express my heartfelt thanks to my supervisor, Karim Rayane. Their guidance, expertise, and mentorship have been invaluable throughout my studies. They challenged me to grow, pushed me to reach my full potential, and always believed in me, even when I doubted myself. I am truly grateful for their dedication to helping me succeed.

I am also immensely thankful to my parents. Their unconditional love, encouragement, and willingness to make countless sacrifices have been the bedrock upon which I have been able to build my achievements. They have been my constant source of strength and inspiration, and I owe so much of my success to them.

Finally, I am grateful to my classmates, who have been on this journey alongside me. The camaraderie, collaboration, and support we have shared have made the challenges of our studies more manageable and the triumphs all the sweeter. I am honored to have had the opportunity to learn and grow alongside such a remarkable group of individuals.

As I embark on the next chapter of my life, I carry with me a deep sense of appreciation for all those who have contributed to my success. This graduation is as much theirs as it is mine, and I am truly thankful

Summary

Table of content

List of figures

List of tables

List of abbreviations

ML	**Machine Learning**
AI	**Artificial intelligence**
IM	**Industrial Maintenance**
PM	**Predictive maintenance**
SM	**Systematic maintenance**
CM	**Corrective Maintenance**
CBM	**Condition-based maintenance**
SVM	**Support Vector Machine**
TSD	**Time series data**
JVM	**Java virtual machine**
T	**period**
f	**frequency**
RMS	**Root mean square**
CF	**Crest factor**

General Introduction

In the industrial field, controlling production costs while ensuring a desired level of quality is a key challenge in this field. One of the major objectives of manufacturers is to minimize as much as possible the high costs induced by unplanned downtime and equipment breakdowns that can generate considerable losses. Predictive maintenance is a proactive maintenance method that uses data and analytics techniques to predict potential equipment failures before they occur. It allows manufacturers to schedule maintenance interventions at the right time, before equipment fails, reducing unplanned downtime and high maintenance costs. Predictive maintenance uses a variety of technologies such as vibration monitoring, thermal analysis, oil analysis, equipment condition monitoring, energy consumption monitoring, maintenance data analysis, and other methods to monitor the health status of equipment. This data is then analyzed to predict potential failures and plan necessary maintenance interventions. By using predictive maintenance, manufacturers can also avoid unnecessary costs related to replacing non-defective spare parts, reducing overall maintenance costs. In addition, this method also helps reduce the chances of unforeseen property and human damage, as it helps identify problems before they cause significant damage. In short, predictive maintenance is a key element in improving the reliability, safety and availability of production equipment while reducing maintenance costs. It also allows manufacturers to focus on efficient and sustainable production strategies, allowing them to remain competitive in an increasingly demanding global market. Machine learning is a branch of artificial intelligence that allows computer systems to learn and improve their performance from data, without being explicitly programmed. In the field of predictive maintenance, machine learning can be used to improve the accuracy of failure predictions and to identify complex failure patterns that are not easily detectable with traditional methods.

Machine learning is used in predictive maintenance to develop failure prediction models using sensor data and other relevant data sources. These models can be trained from historical data to predict future failures with increased accuracy. For example, a machine learning model can be trained to predict machine failures by analyzing sensor data such as vibration levels, temperatures, oil levels, and more. The model can be trained to recognize complex failure patterns and to alert maintenance teams before failure occurs. Machine learning can also be used to optimize maintenance strategies by analyzing maintenance data, such as repair and part replacement times, and using this data to improve maintenance scheduling. By using machine learning in predictive maintenance, manufacturers can improve the efficiency of their equipment and reduce maintenance costs. In addition, it can also improve worker safety by reducing the chance of unplanned equipment failures.

Traditional maintenance, based on post-failure diagnosis, has been replaced by more predictive approaches such as preventive maintenance, condition-based maintenance and predictive maintenance. Predictive maintenance focuses on real-time monitoring of equipment, collecting data, and analyzing that data to predict failures before they happen. Machine learning, as a branch of Artificial intelligence, offers a method for data analysis and pattern identification that can be used to improve predictive maintenance. Machine learning algorithms can be trained from historical data to identify complex failure patterns and to accurately predict future failures. By using predictive maintenance and machine learning, companies can optimize their maintenance strategies, reduce downtime and reduce downtime.

The first chapter is devoted to maintenance in industry. with a focus on predictive maintenance. We address the different challenges of maintenance in industry, which have evolved over time and are reflected in the different forms of maintenance used, from traditional maintenance to predictive maintenance. We also present the link between predictive maintenance

In Chapter 2 we will provide an overview of artificial intelligence in general, with a particular focus on machine learning. We will discuss the most commonly used techniques in recent literature, defining machine learning and presenting its most common types, such as regression and classification. We will also focus on algorithms related to failure prediction and maintenance. The goal of this chapter is to introduce a machine learning approach that will be used for modeling and implementation in subsequent chapters.

In Chapter 3 we present a case study using and we will make our contribution in order to solve the problem in question. After the collection and description of the dataset used by the sensor and phone wave , we proceed with the pre-processing. Selecting an appropriate machine learning approach is crucial analytically. Then, we present a number of algorithms adopted to model the initial objective, as well as the techniques to measure the performance of the built models. Explaining the implementation of the approach, starting with an overview of the tools and development environment. Then, we will present some interfaces illustrating the results obtained by applying the chosen algorithms on the prepared dataset. Finally, we validate this approach with performance metrics. In the general conclusion, we summarize all the developments in relation to the initial objective of the study, we summarize the main results obtained.

CHAPTER I
vibration of rotating machines

1.1 Introduction

Maintenance techniques are most times reliant on experienced-based information and are subsequently formalized into a system discipline with all the necessary technical particulars that prepare their evaluation method for the occurrence that was experienced in the experimental or operational phases. Maintenance methods adequately give product design methods and information to enable it to do better than before and most importantly predict the future new equipment behavior.

1.2 Industrial maintenance

A first normative definition of maintenance was given by AFNOR in 1994 (NFX 60-010 standard), namely: "all actions to maintain or restore an asset in a specified condition or capable of providing a specific service". Since 2001, it has been replaced by a new definition, now European (NF EN 13306 X 60-319): "All technical, administrative and management actions during the life cycle of an asset, intended to maintain or restore it to a state in which it can perform the required function". The European Federation of National Maintenance Societies (EFNMS) offers a similar definition: "All actions which have the objective of retaining or restoring an item in or to a state in which it can perform its required function. The actions include the combination of all technical and corresponding administrative, managerial and supervisory actions", i.e.: All actions which have as their objective the maintenance or restoration of a thing to fulfil the function required of it.

Maintenance activities involve interventions on multi-technology equipment. These interventions require scientific and technical knowledge of the systems, products, processes, hardware and software used, as well as how they work and the principles that govern their interactions [1].

1.3 Types of maintenance

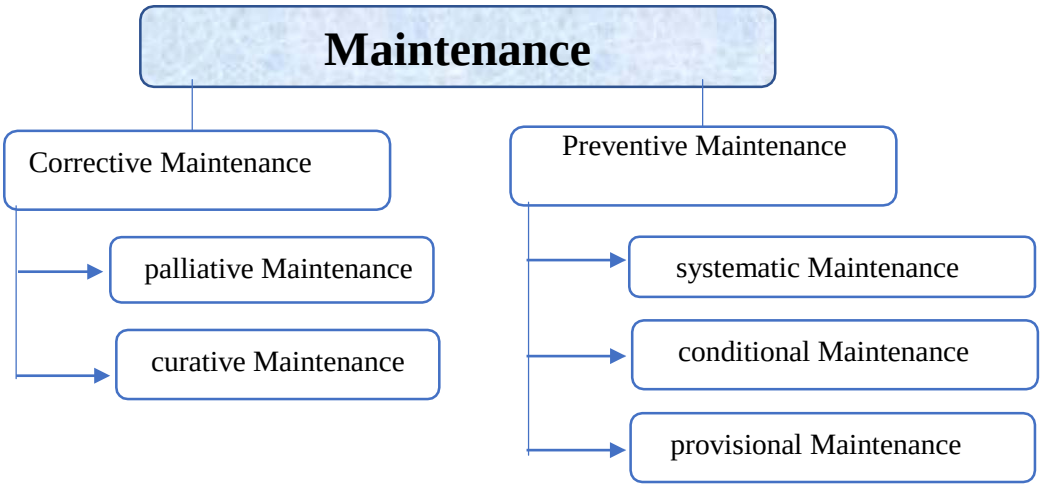


Figure1-1: Forms of maintenance according to standard NF EN 13306 (2010). [4]

1.3.1 Corrective or accidental maintenance

This is why corrective maintenance is defined as maintenance carried out after failure, the NF EN 13306 (2010) standard defines it as maintenance carried out after the detection of a breakdown and intended to restore an asset to a condition in which it can perform a required function [1].

Corrective maintenance is characterized by its random nature and requires competent human resources and material resources, namely: spare parts and tools, available on site. This type of maintenance is generally suitable for equipment where:

- The consequences of the outage are not critical;
- Repair is easy and doesn't require much time;
- Investment costs are low.
- Two forms of corrective maintenance can be distinguished

1.3.2 Palliative corrective maintenance

It is based on the troubleshooting action that allows the equipment to be temporarily restored to an acceptable level of performance that may be lower than the optimal level, the intervention is therefore of a provisional nature, the AFNOR standard [2] describes it as: "Corrective maintenance action intended to allow an asset to temporarily perform all or part of a required function, commonly referred to as troubleshooting". Palliative maintenance is mainly made up of temporary actions that must be followed by curative action

1.3.3 Curative corrective maintenance

In contrast to what is called palliative corrective maintenance, interventions in this type of corrective maintenance are of a definitive nature, the intervention that follows the failure allows the restoration of the optimal level of performance of the equipment, the AFNOR X60-319/NF EN 13306 2010 AFNOR Maintenance Terminology standard defines it as: "a corrective maintenance action the purpose of which is to restore an asset to a specified condition to enable it to perform a required function. The results of the actions carried out must be of a permanent nature."

1.3.4 Preventive maintenance

Unlike the corrective maintenance, which waits for the occurrence of the breakdown to intervene by causing the increase in indirect costs related to the interruption of production, preventive maintenance consists of intervening on a piece of equipment before it fails, so interventions are triggered before failures according to one or more parameters determined after monitoring the behavior of the machine. The aim is to achieve a zero-failure rate by maintaining the required level of performance before the defect occurs, the definition given by AFNOR [2] is as follows: "Maintenance carried out at predetermined intervals or according to prescribed criteria and intended to reduce the probability of failure or the deterioration of the operation of an asset", Its objectives are to:

- Increase the lifespan of equipment;
- Decrease the likelihood of in-service failures;
- Reduce downtime in the event of an overhaul or breakdown;

• Prevent and also anticipate costly corrective maintenance interventions;

• Enable corrective maintenance to be decided in good conditions

• Decrease the maintenance budget;

• Avoid abnormal consumption of energy, lubricant, etc.;

• Eliminate the causes of serious accidents.

Three forms of preventive maintenance can be distinguished:

1.3.4.1 Systematic preventive maintenance

When the maintenance work is carried out at fixed and predefined intervals, it is referred to as systematic preventive maintenance. This type of maintenance is triggered according to a schedule that can be: hours of work, kilometers travelled, etc. and results in the periodic replacement of parts, without prior control and regardless of the state of deterioration of the assets, the definition given by the European standard [2] is: "Preventive maintenance carried out at pre-established time intervals or according to a defined number of units of use but without prior control of the condition of the asset".

The periodicity of replacements is determined by two methods: the first is block and the second is age. The age-type replacement policy suggests replacing equipment after The units of uptime. The block type policy suggests replacing the equipment after a predetermined period of time T, 2T, etc. regardless of the age and condition of the component.

Systematic maintenance therefore requires knowledge of the behavior of the equipment; wear and tear; modes of deterioration; Mean time between failures (MTBF) to determine intervention periods.

1.3.4.2 Condition-based preventive maintenance

Systematic preventive maintenance can lead to an excess of unnecessary interventions, and therefore to financial waste for the company. To overcome this, other forms of preventive maintenance, based on monitoring the actual condition of assets, have appeared: condition-based and predictive maintenance.

Condition-based maintenance is defined as: "Preventive maintenance based on monitoring the operation of the asset and/or the significant parameters of this functioning, including the

actions resulting from it" [2]. It is a maintenance that is subordinated to a predetermined type of event. Various tools, such as vibration analysis and oil analysis, can detect signs of wear or degradation of the equipment. This is done by measuring, at each inspection, the value of a control parameter such as the amplitude of displacement, velocity or acceleration of vibrations, the degree of acidity, or the content of solid particles in the oil. In some cases where measuring equipment or sensors are integrated into the monitored system, the inspection is only done after a signal has been obtained. Generally speaking, the action is only triggered when the control parameter exceeds an empirically determined threshold set by the manufacturer or by occupational health and safety standards.

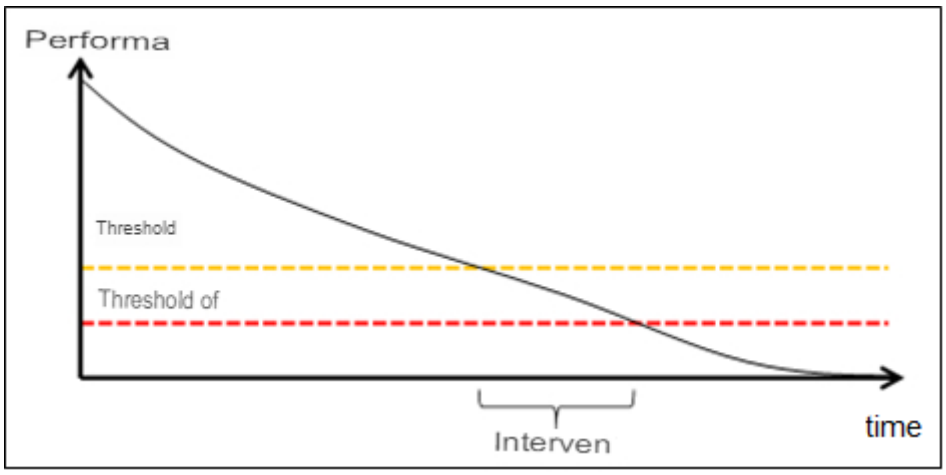


Figure 1-2 Principle of monitoring in condition-based maintenance

1.3.5 Predictive maintenance

Like preventive maintenance, predictive maintenance has many definitions. For some workers, predictive maintenance monitors the vibration of rotating machinery with the goal of detecting emerging problems and avoiding catastrophic failure. For others, it monitors the infrared image of switchgear, motors, and other equipment

to detect developmental problems. The common principle of predictive maintenance is that regular monitoring of the actual mechanical condition, operating efficiency, and other indicators of the operating conditions of machine trains and process systems will provide the necessary data to ensure the maximum interval between repairs of unplanned failures [3].

Preventive maintenance (PM) is an effective way to improve reliability [4]. When it comes to advanced predictive maintenance, the main challenge is accurate failure prediction that could prevent significant loss of operation at the initial stage [5].

1.4 Predictive maintenance techniques

Various technologies can and should be used as part of an overall predictive maintenance program. Because mechanical systems or machines make up most of the equipment in the plant, vibration monitoring is usually the key component of most predictive maintenance programs.

Therefore, a comprehensive predictive maintenance program should include other monitoring and diagnostic techniques. These techniques include vibration monitoring,

thermography, tribology, process parameters, visual inspection, ultrasound, and other non-destructive testing techniques [5].

1.5 Vibration monitoring

Because most installations consist of electromechanical systems, vibration monitoring is the primary tool for predictive maintenance. Over the past 20 years, most of these programs have adopted single-channel microprocessor-based data collectors and Windows® software to acquire, manage, track, and evaluate the vibrational energy created by these electromechanical systems. While this approach is a valuable predictive maintenance methodology, the limitations of these systems may limit the potential benefits [6].

Monitoring the facilities helps to limit the level of preventive maintenance. Vibration analysis is a tool for detecting and diagnosing faults in installations.

1.5.1 Thermography

AFNOR definition: "Thermography is the technique used to obtain, by means of appropriate equipment, the thermal image of a scene observed in an infrared spectral range".

Thermography means "writing with heat" just as photography means "writing with light". This image is called a thermogram or thermal image.

The approach therefore consists of producing images from invisible thermal radiation. As a result, infrared thermography is an instantaneous means of detecting problem areas, highlighting defects that more conventional methods cannot detect.

Thermography is a predictive maintenance technique that can be used to monitor the condition of machinery, structures, and systems in the plant, not just electrical equipment. It uses instrumentation designed to monitor infrared energy emission (i.e., surface temperature) to determine operating conditions. By detecting thermal anomalies (i.e., areas that are hotter or colder than they should be), an experienced technician can locate and define a multitude of emerging problems in the facility.

1.5.2 Tribology

Tribology is the general term that refers to the design and operation dynamics of the bearing-lubrication-rotor support structure of machines. Two main techniques are used for predictive maintenance: lubricating oil analysis and wear particle analysis.

1.5.3 Visual inspections

Visual inspection was the first method used for predictive maintenance. Almost since the beginning of the Industrial Revolution, maintenance technicians have been performing daily "walkdowns" of critical production and manufacturing systems to identify potential failures or maintenance issues that could affect reliability, product quality, and production costs. A visual inspection is always a viable predictive maintenance tool and should be included in all maintenance management programs for the total facility [6].

Other techniques that can support predictive maintenance include acoustic emissions, eddy currents, magnetic particles, residual stresses, and most traditional non-destructive methods

1.6 Vibration analysis monitoring

All mechanical equipment in motion generates a vibration profile, or signature, that reflects its operating state. This is true regardless of speed or whether the mode of operation is rotation, reciprocating motion, or linear motion. Vibration analysis is applicable to all equipment

Mechanical, although a common, but not valid, assumption is that it is limited to simple rotating machines with operating speeds exceeding 600 revolutions per minute (rpm). Vibration profile analysis is a useful tool for predictive maintenance, diagnostics, and many other uses.

Predictive maintenance has become synonymous with monitoring the vibration characteristics of rotating machinery to detect budding problems and prevent catastrophic breakdowns. Vibration monitoring has been found to be an essential part of any good predictive maintenance program for any machine in the plant [7].

However, vibration analysis does not provide the data needed to analyze electrical equipment, heat loss areas, lubricating oil condition, or other parameters typically evaluated in a maintenance management program. Therefore, a predictive maintenance program for the total facility should include several techniques, each designed to provide specific information about the plant's equipment.

1.7 Definition of a vibration

A mechanical system is said to be in vibration when it is animated by a back-and-forth movement around an average position, known as the equilibrium position. A vibration is usually translated as:

• Displacement: the position of the mass varies on either side of the equilibrium point;

• Speed: variation of displacement with respect to time;

• Acceleration: variation in speed in relation to time.

1.8 Characteristics of a vibration

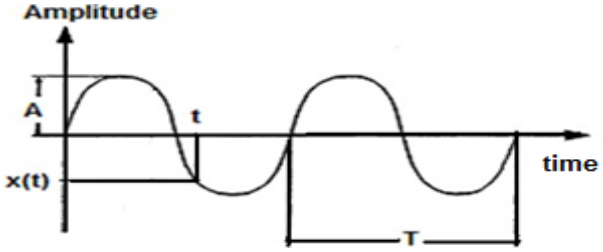


Figure1- 3 Notable Quantities [11]

x (t): this is the instantaneous value of the quantity under consideration.

A: This is the largest value that the variable x (t) can take.

Period T: This is the time interval at the end of which the variable x (t) returns to the same value in the same direction (unit: second [s]).

Frequency f: This is the number of periods per unit of time. Frequency is the inverse of period. (Unit: [HZ]).

Pulsation ω: also known as angular velocity. (Unit: [rad/s])

$$\omega = 2\pi f \quad (1)$$

1.8.1 Amplitude

The amplitude of a vibratory movement is the value of its deviations from its equilibrium position. From this general definition, the complexity of a real vibration signal leads to the definition of several amplitude quantities:

Peak Amplitude (Ac): This represents the maximum amplitude of the signal per relative to its equilibrium value.

Peak-to-Peak Amplitude (Acc): This represents the deviation between the extreme amplitudes of the signal for a given observation time. In the case of a sinusoidal

vibration, it is sometimes

referred to as a double amplitude. It is noted that:

$$Acc = 2Ac \quad (2)$$

RMS (Root Mean Square): This indicates the energy given by the vibratory movement.

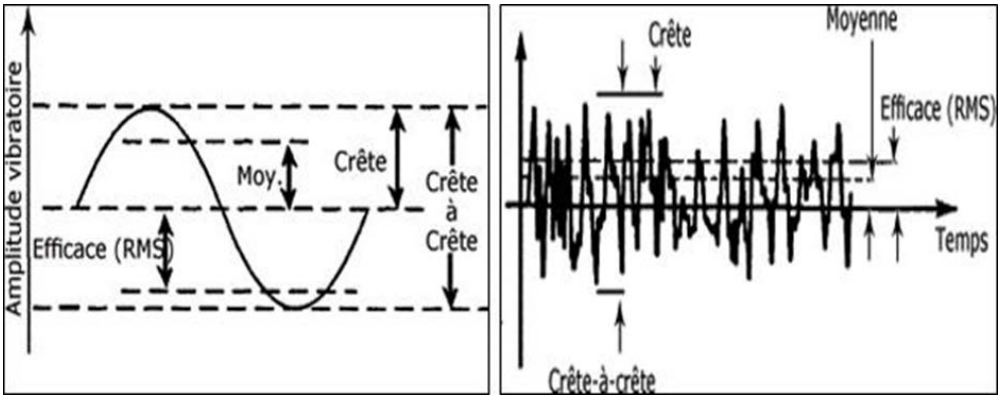


Figure 1-4 Characteristics of a Sinusoidal or Any Vibration

1.8.2 Frequency

Frequency represents the rate at which a phenomenon is repeated or the number of times it occurs in a given amount of time. When the chosen unit of time is the second, the frequency is expressed in Hertz (Hz). A vibration that occurs 50 times/second will therefore have a frequency of 50 Hz. The frequency f is the inverse of the period T, which is the duration of a cycle.

$$f = \frac{1}{T} = \text{hz} \quad (3)$$

1.8.2.1 Natural frequency

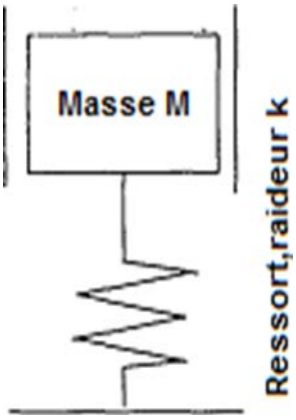


Figure 1-5 Spring Mass System

A shock on a suspended mass generates a sinusoidal vibratory movement as illustrated by Fig (1-5), the frequency of this movement is independent of the shock, it is written:

$$f_P = \frac{1}{2\pi}\sqrt{\frac{K}{M}} \qquad (4)$$

We call f_P , the natural frequency of the spring mass system [43].

1.8.3 Resonance

Suppose that we apply to the above system a sinusoidal force of frequency f of constant amplitude "A". As f approaches f_P, the amplitude of the system's vibration naturally amplifies to a maximum value, this is called system resonance.

A resonance is therefore a phenomenon of amplification of the vibrational level that occurs when the excitation frequency coincides with the natural frequency of the system [12].

1.9 The different forms of vibration

Vibrations encountered in rotating machinery are generally classified by:

•Harmonics

•Periodicals

• Aperiodic [11].

1.9.1 Harmonic vibrations

A harmonic vibration is a vibration whose amplitude-time diagram is represented by a sinusoid (Figure 6).

A harmonic vibration can be generated by the unbalance of a moving rotor.[11]

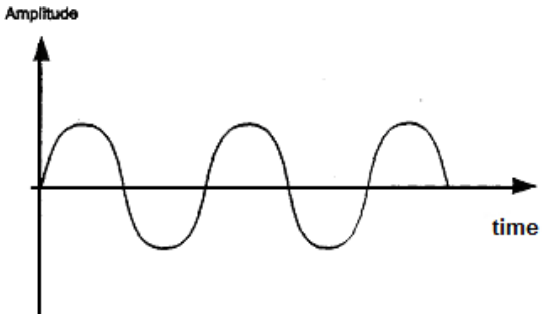


Figure1- 6 Harmonic Vibration [11]

1.9.2 Periodic vibrations

A periodic vibration is such that it recurs exactly after a certain amount of time called the period (Figure 7). Such a vibration is generated by a periodic excitation. This is the most common case encountered on machines. A periodic vibration is the compound of

several harmonic vibrations.[11]

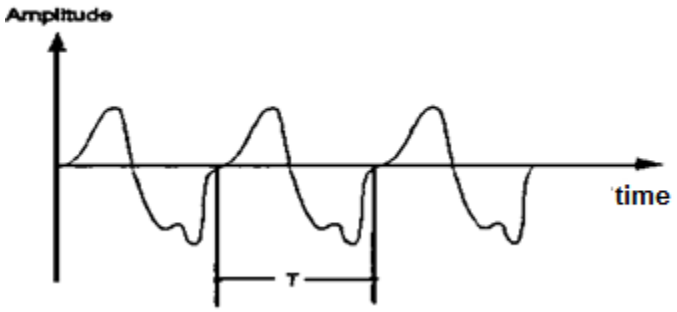


Figure 1-7 periodic vibration [11]

1.9.3 Aperiodic vibrations

An aperiodic vibration is such that its temporal behavior is nondetermined, i.e., reproducibility over time is never observed (Figure 8). This is the case for recorded shocks on a crusher [42].

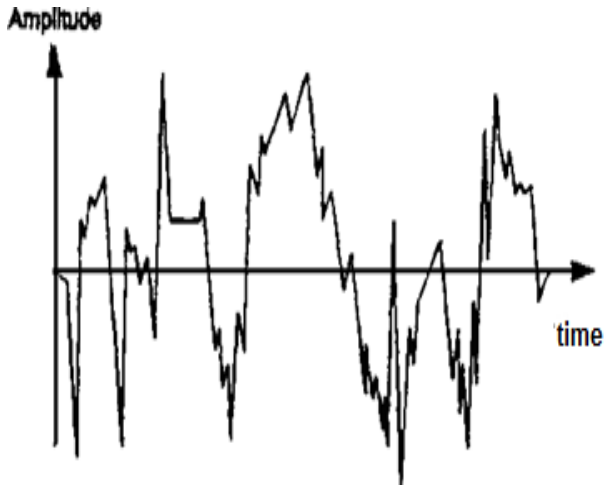


Figure 1-8: aperiodic vibration [42]

1.10 Source of data exploitation

The vibration analysis process requires gathering complex machine data and deciphering it. In contrast to simple theoretical vibration curves, the profile of a piece of equipment is extremely complex because there are usually many sources of vibration. Each source generates its own curve, but these are essentially summed and displayed as a composite profile. These profiles can be displayed in two formats: time domain and frequency domain [4].

1.10.1 The time domain

Vibration data plotted as a function of amplitude and time is called a time-domain data profile (Figure 9). Time diagrams should be used for all linear and reciprocating machines. They are useful in the overall analysis of machine trains to study changes in operating conditions; However, time-domain data is difficult to use. Because all vibration data in this type of plot is summed to represent the total displacement at a given point in time, it is difficult to directly see the contribution of a particular vibration source.

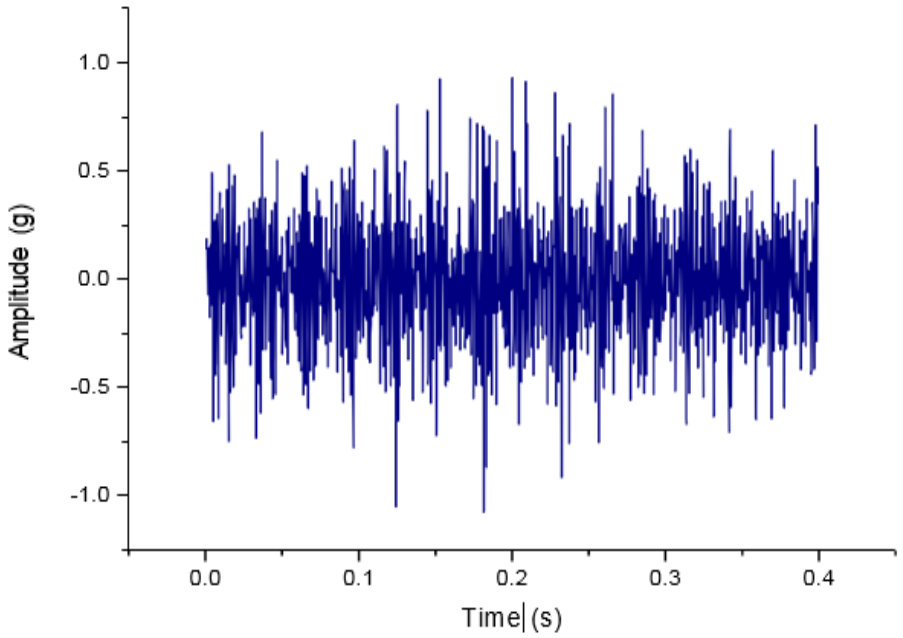


Figure 1-9: Time Spectrum

1.10.2 Frequency domain

From a practical point of view, the simple harmonic vibration functions are related to the circular frequencies of rotating or moving components. Therefore, these frequencies represent a multiple of the basic operating speed of the driveline (Figure 10). Determining these frequencies is the first fundamental step in analyzing the machine's operating condition.

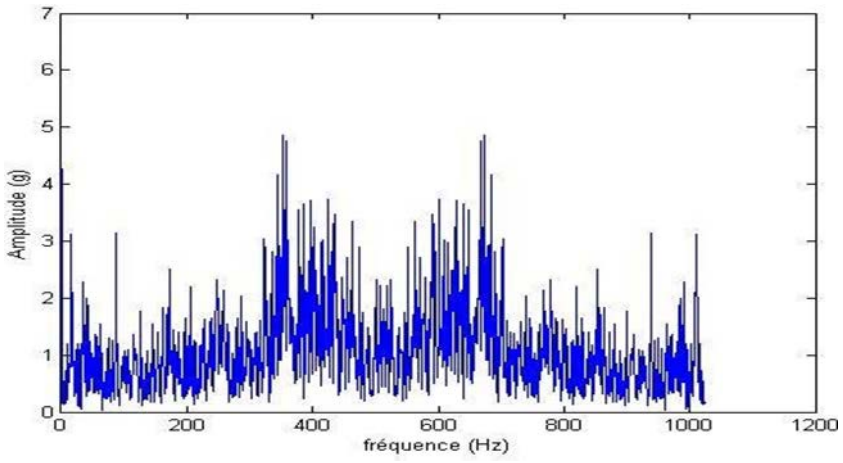


Figure 1-10: Frequency Spectrum

Frequency domain data is obtained by converting time domain data using a mathematical technique called Fast Fourier Transform (FFT). FFT allows each vibration component of a complex machine-train spectrum to be represented as a discrete frequency peak. The amplitude of the frequency domain can be the displacement per unit of time related to a particular frequency, which is represented by the Y-axis as a function of the frequency as the X-axis. This is in contrast to time-domain spectra that sum the velocities of all frequencies and plot the sum as the Y-axis versus time as the X-axis.

Formulas (5) and (6) represent the Fourrier transform and its discrete variant, respectively.

$$X(f) = \int_{-\infty}^{+\infty} x(t)e\text{^} - i2\pi ftdt \qquad (5)$$

$$X(f) = \sum_{k=0}^{Ne-1} x.(k)e^{-2i\pi f\frac{k}{N}} \qquad (6)$$

PARCEVAL's theorem states that the energy contained in the time signal is equal to that in its frequency representation. From this we can also construct power spectra (DSP power spectral density) on the finite power signals, representative of the square of the modulus of the Fourier transform, relative to the observation time [8].

1.10.3 The spectrum

Spectral analysis is a complementary analysis technique, developed on several variants, the most widely used of which are the complex constrain is defined as the inverse Fourier transform of the decimal logarithm of the Fourier transform,

is expressed in terms of a time-uniform variable, and is represented by formula (7) [9].

$$C = TF^{-1} [Ln| X(f) |] \quad (7)$$

The power cube is defined as the inverse Fourier transform of the decimal logarithm of the modulus of the Fourier transform of the signal, represented in equation (8)

$$C = TF^{-1} [Ln |X(f)|]^{2} \quad (8)$$

1.10.4 Wavelets

Unlike the FTTS, the wavelet transform is a signal processing method with a resolution that is adaptive to the size of the object or detail being analyzed.

1.10.5 Envelope analysis

Envelope analysis is a technique for the early detection of shock-type defects. To do this, the vibration signal is measured in a wide frequency band, and filtered around a resonant frequency. The signal is then rectified "by putting all negative values in the positive", and the Hilbert transform is applied in order to raise its envelope and thus dissociate the modulated signal (resonant frequencies) from the modulating signal corresponding to the desired defect. The final diagnosis can then be made after spectral analysis of the envelope [10].

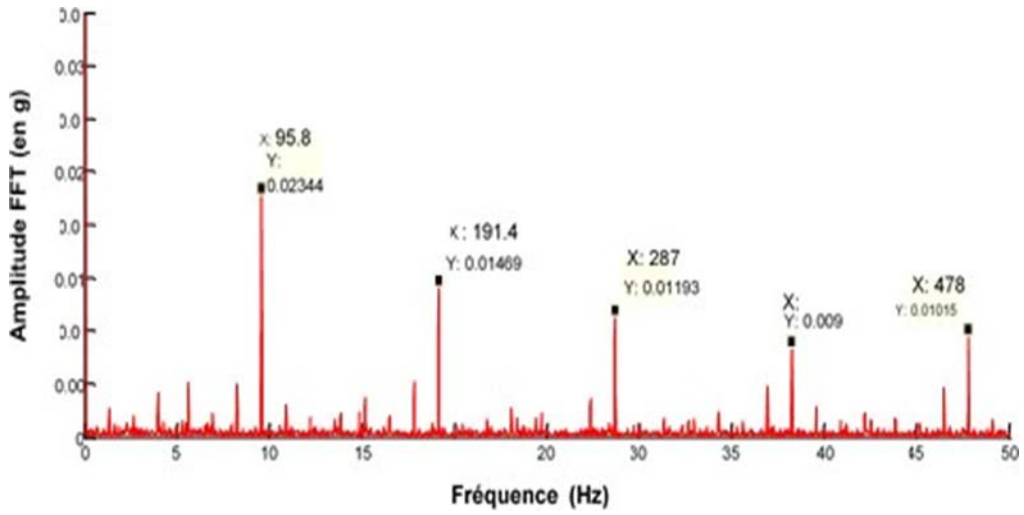


Figure 1- 11: FFT of Time Signal Envelope (with Filter + Hanning) (g)

1.10.6 Scalar indicators

A raw signal or a signal that has been previously processed (filtering, demodulation, etc.) is characterized by the following quantities:

• Amplitude RMS., peak amplitude, modulation rate, etc.);

• Amplitude distribution (crest factor, Kurtosis);

• Spectral composition (amplitude of a spectral component, RMS value of a family of components, harmonic rate, etc.) [10]

1.10.7 Rms

The RMS, also known as the RMS or RMS value of a signal, is the square root of the moment of order two and is calculated as shown by equation (9).

$$\text{RMS} = \sqrt{\frac{1}{t}\int_0^t f(t)^2\, dt} \qquad (9)$$

The RMS is one of the first indicators used in industry. This is due in part to its simplicity and speed of execution. An excessive variation in the RMS level usually means a change in the operating state and therefore a failure. One of the major disadvantages of using RMS is that it usually gives a rather late alarm, especially in the case of bearing faults, where the signal variation due to the appearance of the fault is masked by other components of higher amplitudes.

1.10.8 Crest factor

The crest factor FC is a more specific indicator, which allows you to observe the vibration signal more closely. Crest factor monitoring allows for earlier detection of defects by measuring the ratio of the maximum signal modulus value (peak value) to the RMS value, as shown by equation (10)

$$\text{CF} = \frac{peak\ value}{RMS} \qquad (10)$$

1.10.9 Kurtosis

More specific to the detection of bearing defects, the Kurtosis is a statistical quantity used to analyze the "sharp" or "flat" character of a distribution, and therefore to observe the shape of the signal. Derived from the four-order statistical moment, it is defined as the ratio of the average value of the high signal to the power of 4 to the square of its energy. It is given by formula (11)

$$\text{Kurtosis} = \frac{\sum(xi-\bar{x})^4}{nS^4} \qquad (11)$$

$\bar{x}$ = mean of the given data

S = standard deviation of the data

n = total number of the observations

1.11 The different defects of a rotating machine

Examples of defects that can lead to machine failure include mass imbalance, rotor friction, shaft misalignment, gear failures, and bearing defects

In addition to detecting the early onset and severity of a fault, there are systems that can also be designed to identify components that are deteriorating and estimate the time interval that the monitored equipment can still operate before failure. These systems continuously measure and interpret signals (e.g., vibration, acoustic emission, infrared thermography, etc.), which provide useful information for identifying the presence of defective symptoms

1.11.1 The unbalance

Unbalance in a rotor is when the mass is not uniformly distributed, causing the rotor to vibrate.

This vibration is caused by the vibration excited by the interaction of an unbalanced mass element with the radial acceleration involving rotation generating a centrifugal force. The force produced is rotating since the mass component revolves; it also revolves and attempting to move the rotor along the line of action of force.

The vibration will reach the rotor's bearings and every point along the bearing will encounter the force one's revolution.

1.11.1.1 Static unbalance

It is the eccentricity of the center of gravity of a rotor due to a point mass at a certain radius from the center of rotation as shown in Fig. 1-12. A mass of equal magnitude and at an angle of 180° to the unbalanced mass along the same radius is required to restore the center of gravity to the center of the rotation.

Static Balancing Involves resolving primary forces into one plane and the addition of a correction mass in that plane only. A lot of rotating objects, with their mass mostly centered in or close to one plane, like flywheels, grindstones and car wheels, etc., can be seen as static balancing issues. When a rotor has a diameter more than 7 to 10 times its width its generally treated as a single-plane rotor.[19]

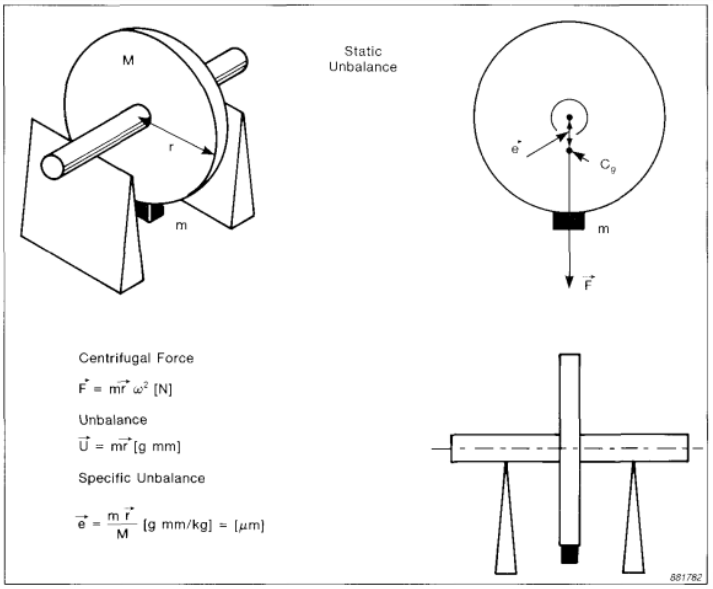


Figure 1- 12: Static unbalance

1.11.1.2 Couple (moment) unbalance

Can be found in rotors whose diameter is 7 to 10 times smaller than their width. For a cylinder, as shown in **Fig 1-13 ,** there can be two equal masses arranged symmetrically about the center of gravity, but at an angle of 180° to each other. The rotor is in static balance, which is There is no eccentricity of the center of gravity, but if the rotor turns When rotating, the two masses cause the inertial axis to move so that it no longer corresponds to the axis of rotation, causing the bearing to vibrate strongly. This unbalance can only be corrected by measuring the vibrations of the rotor as it rotates and adding balancing masses in both planes.

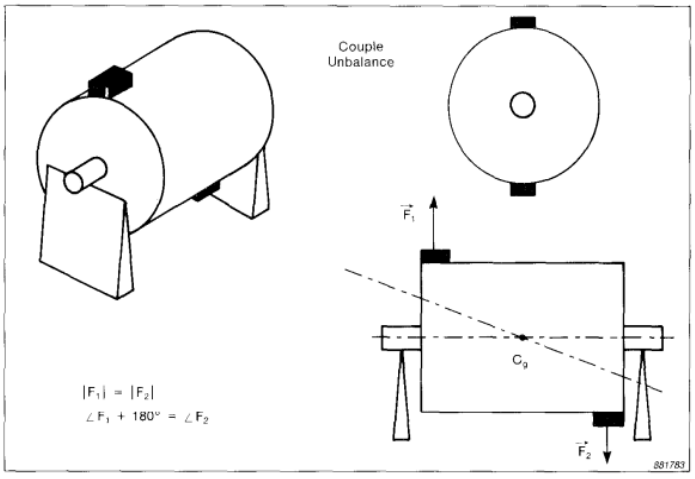


Figure 1-13: Couple unbalance

The difference between static balance and couple balance is shown in Figure 1-13. It can be seen that when the rotor is stationary, the end masses balance each other. But there is

a strong sense of unbalance when rotating.[19]

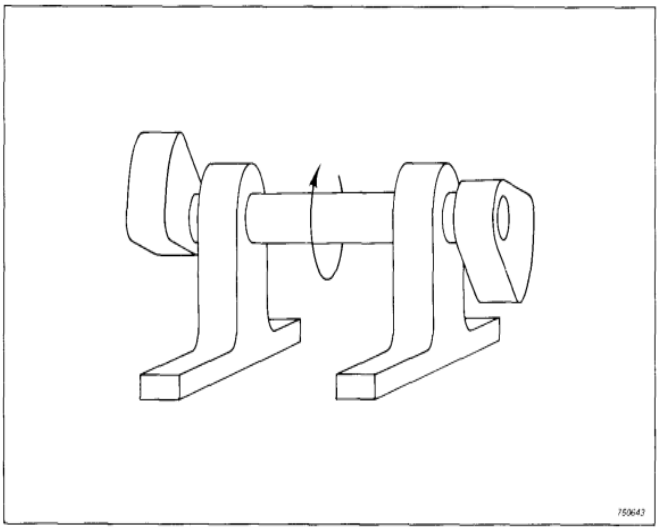


Figure 1- 14: Static balance, couple unbalance

1.11.1.3 Dynamic unbalance

The imbalance shown in Fig 1-15 is a combination of static and coupled unbalance and is the most common type of unbalance in rotors. To correct the dynamic unbalance, vibration measurements are a must while the machine is running and compensating masses are added in two planes.

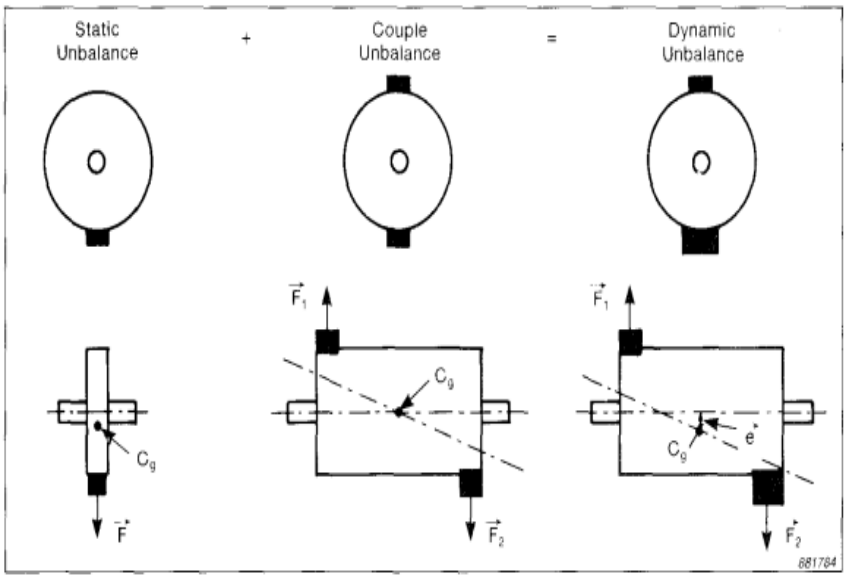


Figure 1-15: Dynamic unbalance

Rotors are divided into rigid rotors and flexible rotors. This application note only covers rigid rotors. A rigid rotor is a rotor that operates at a speed less than 50% of its first critical speed. Above this speed, the rotor is considered flexible. A rigid rotor can be balanced by making corrections in any selected two planes. The balancing process of a flexible rotor is more complicated due to the elastic deformation of the rotor.[19]

1.11.2 Misalignment

A shaft is an essential part of the rotating machine; it is used to transmit power and motion [13].

Shaft misalignment is a common problem in rotating machines that causes more than 70% of vibration problems [14]. It occurs when the axes of rotation of two (or more) machine shafts are not aligned. This increases the axial and radial forces on the bearings, seals and couplings, thereby inducing wear of these components and deflection of the shaft reducing the amount of power transmitted [15].

Even if initially, or after adjustment, the shaft is aligned, during operation various factors such as thermal growth, piping pressure and foundation movements will change the alignment [16]. The misalignment generally manifests itself in the vibration spectra by a spike which has a frequency equal to 2 times the rotation frequency (2F0).

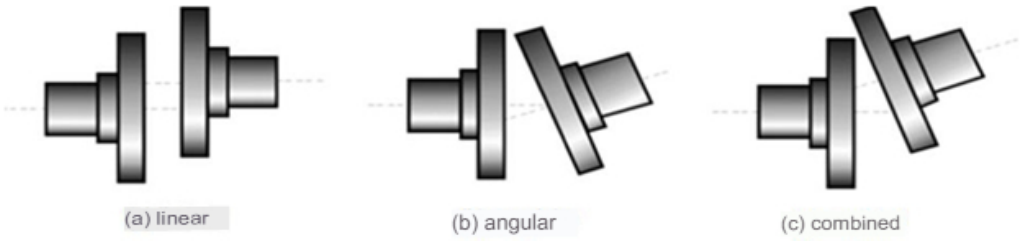


Figure 1-16: Types of Misalignments

1.11.3 Bearing defects

Bearings are critical components of rotating machinery and monitoring their condition is important to prevent catastrophic failures and reduce machine downtime [17].

Twelve main damages can be considered and come from four families of causes:

- **Damage related to the load and the speed applied:** chipping, seizing, staining and deterioration of the cages.
- **Lubrication-related damage:** galling, breakage of rings due to thermal stress, coloring, deterioration of cages and chipping,
- **Damage related to assembly:** imprints of rolling bodies due to plastic deformation, marks of blows, certain deterioration of the cages, corrosion by contact, certain chipping

due to misalignment and roundness.

- **Damage related to the environment:** wear, fingerprints, corrosion, craters, grooves created by the passage of an electric current.

Figure 1-17: Bearing defects

1.11.4 Gear defects

Gears are important components of almost all machines used in the industrial environment. Therefore, the detection of a defect in these organs must be detected in advance to avoid catastrophic failure [18].

Figure 1-18: Gear defects

The gear defect is usually manifested in the vibrational spectra by a spike that has a frequency equal to: $f = z.f_0$ (7)

1.12 Conclusion

In this chapter we presented in the first part a definition and a general overview on industrial maintenance. We then discussed the different types of maintenance, and in the second part, we also talked about vibrations and types of vibrations, their characteristics and the faults that cause them in general

CHAPTER II
Machine Learning

2.1 Introduction

Machine learning, is a branch of artificial intelligence (AI) and computer science that involves learning from data and algorithms to mimic the way human beings learn, gradually improving their accuracy. Machine learning is a key component of data science and plays an important role in many fields. Using statistical methods and algorithms, they are trained to perform classifications or predictions, which can uncover valuable insights in data mining projects. This information can be used to make decisions across applications and businesses, and ideally impacts key growth metrics. With the continued growth of data and the digitization of many industries, the demand for competent data scientists will only increase in the coming years. These professionals will play a critical role in identifying the most relevant business questions and collecting the data needed to answer them. [16] In this chapter, , we introduce the most important concepts and techniques used in machine learning. Finally, we will establish a state-of-the-art on the application of machine learning to predictive maintenance.

2.2 Definition of Artificial Intelligence

In other words, artificial intelligence is a technology that allows machines to perform tasks that normally require human intelligence, such as voice recognition, decision-making, and complex problem-solving. AI systems can learn from data, through machine learning algorithms, and gradually improve based on experience. Artificial intelligence can take different forms, such as chatbots, recommendation systems, image recognition, and fraud detection. It can be used in many industries, such as finance, healthcare, manufacturing, education, and transportation, to improve process efficiency and accuracy. Although there are concerns that AI will replace human jobs, it is actually designed to work collaboratively with human beings, allowing them to focus on higher-value tasks. Artificial intelligence is therefore a valuable asset for companies looking to improve their performance and competitiveness [20].

2.3 Definition of machine learning

Machine Learning is a sub-branch of Artificial Intelligence that allows machines to learn from data without being explicitly programmed for each specific task. AI aims to replicate certain aspects of human behavior such as the ability to understand natural language, reason, perceive, learn, and make decisions. AI can be built in a variety of ways, including using symbolic or neural approaches. Currently, neural AI is the most popular method Commonly used because of its ability to adapt and learn from data. Machine learning often requires large amounts of data to train and improve, which raises ethical questions related to

privacy and data security. [21] Predictive Plant Maintenance Machine Learning is an AI application that helps detect potential failures of industrial equipment by analyzing performance data and predicting failures before they occur. This allows companies to reduce unplanned downtime, improve worker safety, and reduce maintenance costs. [22] In recent years, machine learning has become increasingly important in computing because data can be collected and stored much more easily.

2.3 Machine Learning Use fields

With the democratization of machine learning algorithms and the availability of the data necessary for their proper functioning, it is clear that this technology is increasingly used in different fields, Machine learning concerns all sectors of activity, including industry, Legal, luxury transport, accounting, commerce, health...

In the industrial world:

• Predictive maintenance and equipment monitoring.

• Data analysis can also be used to optimize the complex production cycles of certain factories.

• Inventory forecasting.

• Market/competition analysis.

Marketing:

• Improved ad campaigns by analyzing previous successful ads.

• Increase sales by personally targeting the most fortunate customers to make the purchase.

Social media:

• Trend studies.

• Brand or product image studies.

• Analysis of reactions to current events (opinions).

Product placements, such as choosing posts in your Facebook News Feed. Based on a few criteria such as the user's tastes, the algorithm will determine the chances that you will like the post and then decide whether or not it will appear in your news feed. [23]

2.4 Approaches to Machine Learning

There are 3 main approaches in machine learning: supervised learning and unsupervised and reinforcement learning, each proceeding by its own methods as shown in Figure 2-1 Supervised learning is what we're interested in for prediction. It is made up of several algorithms, both for regression and classification. To choose an algorithm, you need to understand the foundations of existing algorithms and what distinguishes them, which allows you to create the models that would best address a particular problem.

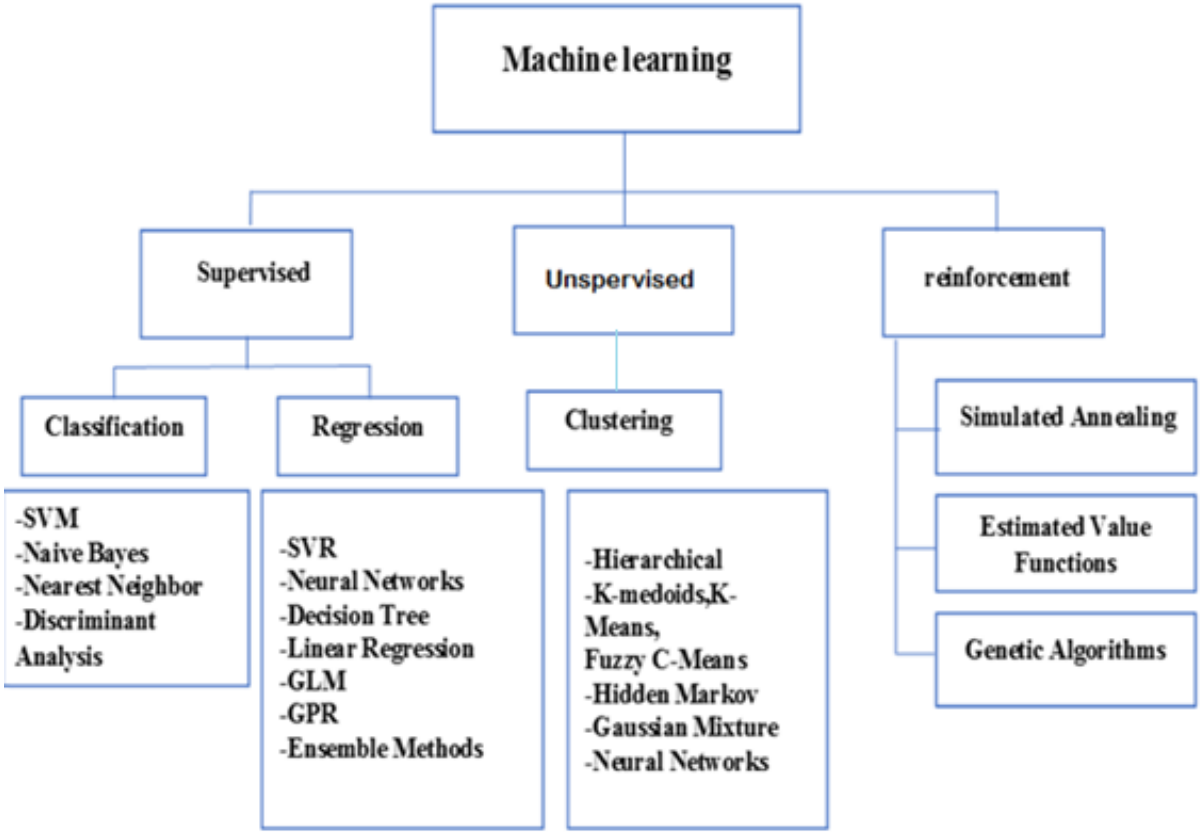


Figure 2-1: Classifications in machine learning [20]

Indeed, there are different ways for a machine to learn, and machine learning approaches are generally distinguished by the types of problems they attempt to solve and the type of data used to train the model. Here are the three main sub-areas of machine learning

2.4.1 Supervised Learning

This method is based on a supervised learning algorithm, i.e. the machine is interested in input data and responses to data that is already known. This processing of the known data will allow the algorithm to train and develop a forecasting model adapted to the data to be processed subsequently. To develop these predictive models, supervised learning will be based on classification techniques. The practical application of classification techniques implies that data

can be identified and categorized according to their characteristics. The application of classification techniques can be found in several fields: digital, medical, banking. One of the most common examples of classification is the automatic determination of the nature of an email: genuine or spam. Supervised learning is also found in regression techniques. [25] This approach consists of training a model from input examples and output (labels) known. In this way, the model learns to predict the correct outputs for the new inputs presented to it. Common examples include classification, regression, and anomaly detection. [26] Supervised learning is a very rich method, but it requires quality data and a good understanding of the problem to be solved. It is important to note that the algorithm will not be able to generalize beyond the training data, which can lead to erroneous predictions if the training data is not representative of the actual data. It is therefore important to select the data carefully and ensure that the model is regularly updated to account for changes in the actual data.

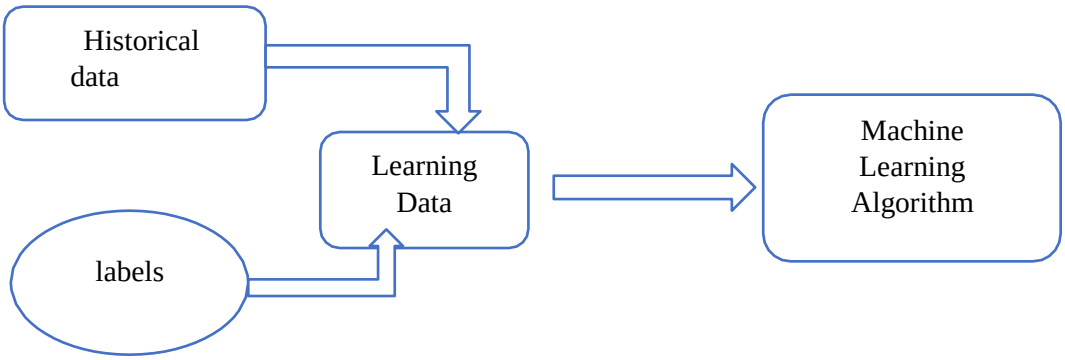


Figure 2-2: Supervised Machine Learning Step 1. [20]

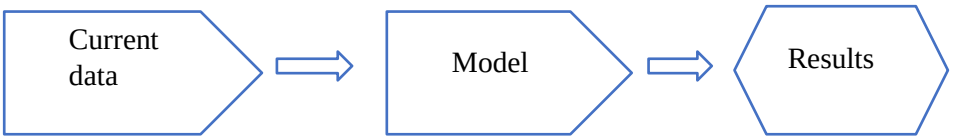


Figure 2-3: Supervised Machine Learning Step 2. [20]

Supervised machine learning can be divided into two classes:

• Classification: The output variable is a category.

• Regression: The output variable is a specific value. [27]

What are the applications of supervised learning:

There are many applications of supervised learning:

- Natural language processing.

- Voice recognition.
- Computer vision.
- Bioinformatics.

Supervised learning is also used for spam detection in emails, the management of chatbots and voice bots, as well as for robotics. This method also enables the development of embedded technologies dedicated to autonomous vehicles. [28]

What are the supervised learning algorithms:

To create a supervised learning model, different algorithms can be used:

• Linear regression: y = c + b * x.

• Logistic regression: h(x) = 1 / (1 + e^-x).

• The decision tree with different output variables.

• The Support Vector Machine (SVM).

We can also mention the k-NN algorithm for performing tests and naïve Bayesian classification for categorizing large volumes of data. [26]

2.4.2 Unsupervised learning

The opposite of supervised learning, unsupervised learning will rely on input data whose answers are not identified. This type of technical objective is to highlight patterns intrinsic to the processed data. Among the unsupervised learning is the clustering technique. This unsupervised learning model is the most common and allows for the identification of commonalities between certain data and the clear grouping of them into groups. [23] This approach aims to uncover hidden structures or patterns in data with no known training labels. Common examples include data segmentation, anomaly detection, dimensionality, and clustering. [24] Unsupervised learning is a very useful method for discovering knowledge from raw data, but it can be more difficult to implement than supervised learning because it often requires pre-processing of the data to make it actionable.

Unsupervised machine learning can be divided into two classes:

• Clustering: The goal is to find clusters in the data.

• Association: The goal is to identify the rules that will be used to define large groups of data.

The main algorithms in unsupervised machine learning are: K-Means, clustering/hierarchical grouping, and dimensionality reduction. [25]

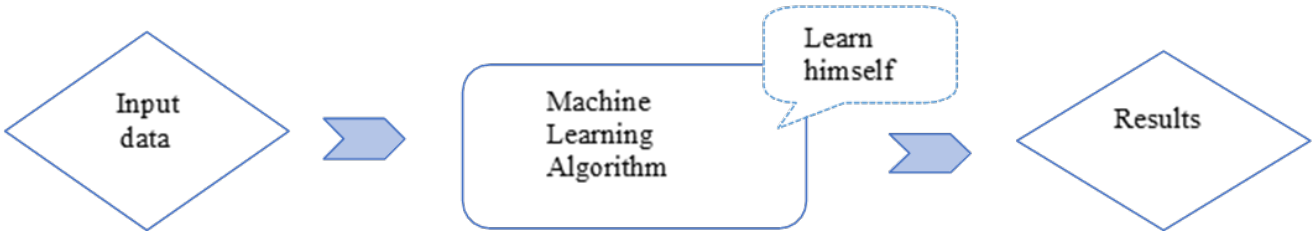


Figure 2-4 unsupervised Machine Learning. [15]

	Supervised learning	Unsupervised learning
Input data	Known input data	Unknown input data
Computer complexity	Complex	Less complex
Areas of activity	Classification and regression	Exploitation of rules of clustering and association
Accuracy	Produces accurate results	Generates moderate results

Table 1: Differences: supervised learning and unsupervised learning. [27]

2.4.3 Reinforcement Learning

This approach involves an agent system that interacts with its environment to maximize a long-term reward. In this way, the model learns to make decisions based on its condition and the reward it receives. Common examples include games, robotics, and trajectory planning. [26] In 2013, it was already a reinforcement machine learning (Q-learning) algorithm that made itself famous by learning how to win in six Atari video games without any intervention from a programmer.

Reinforcement machine learning can be divided into two classes:

• Monte Carlo: The program receives its rewards at the end of the "terminal" state.

•Time Difference Machine Learning (TD): Rewards are evaluated and awarded at each stage.

The main algorithms for reinforcement machine learning are: Q-learning, Deep Q Network (DQN), and SARSA (State-Action-Reward-State-Action). [27]

2.5 Open-source artificial intelligence technologies (importing libraries and dataset)

Machine learning and artificial intelligence (AI) technologies are changing rapidly all areas of our society. It seems that AI is becoming more and more important in our daily lives. As a result of these rapid advancements, many talents and resources are being dedicated to accelerating the growth of technologies.

Python libraries make it very easy for us to handle the data and perform typical and complex tasks with a single line of code.

Here's a list of the best Open-source artificial intelligence technologies and libraries. [30]

2.5.1 TensorFlow

Originally released in 2015, TensorFlow is an easy-to-use and easy-to-deploy open-source machine learning framework. It is one of the best maintained and most used machine learning frameworks

- Produce by Google to support its search and search objectives

  TensorFlow is now widely used by several companies, including Dropbox, eBay, Intel, and Uber. TensorFlow can be used for a variety of machine learning tasks, including classification, regression, image segmentation, speech understanding, machine translation, character recognition, text analysis, and many more. It is also possible to combine multiple models to create complex neural network architectures.

- TensorFlow also provides tools for data preprocessing, model visualization, model validation, and result production, as well as functionality for model export and import. It is also compatible with a variety of programming languages, including Python, C++, Java, Rust, and most recently JavaScript, making it accessible to a wide audience of developers.

- Concretely, the Framework allows you to develop neural networks (and even other computational models) using flow graphs.

In summary, TensorFlow is a flexible and powerful machine learning model development framework that can be used for a wide variety of machine learning tasks. [30]

2.5.2 Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

It is free software released under the three-clause BSD license.

The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals, as well as a play on the phrase "Python data analysis".

Wes McKinney started building what would become Pandas at AQR Capital while he was a researcher there from 2007 to 2010.[5] wiki

Pandas allows you to create tables containing objects of different types, and to perform operations very similar to those that can be performed with Excel.

2.5.3 Keras

Released in 2015, Keras is a very popular open-source software library for machine learning. It has been designed to allow developers to create machine learning models in a simple and fast way by offering

A user-friendly and intuitive programming. Keras is coded in Python and can be deployed on other AI technologies such as TensorFlow, Microsoft Cognitive Toolkit (CNTK), and Theano. Keras is particularly appreciated for its user-friendliness, modularity and ease of scalability. It's perfect if you need a tool for machine learning that allows for quick and easy prototyping, supports convolutional and recurring networks, and runs from that point of view to the machine.

This ease of use does not impact flexibility: Keras integrates with deep learning tools (especially Tensorflow) at a low level, allowing you to implement all the features you would like to use. For example, Keras integrates into your workflow under TensorFlow, as tf.keras. Keras is widely used by the industrial world as well as the research community. [28

Keras makes it easy to convert models into finished products:

Models can be easily deployed on a wider range of platforms than other deep learning tools:

- On iOS, via Apple CoreML (Keras support is officially provided by Apple).
- On Android, with the TensorFlow runtime for android (example: The Not Hotdog app)
- In the browser, with GPU-supporting libraries like Keras.js and WebDNN.
- On Google Cloud, with TensorFlow.
- In the Python web application engine (e.g. with the Flask framework).
- On JVM, with the DL4J model import functionality provided by SkyMind.
- on Raspberry PI. [29]

2.5.4 numpy

NumPy is a very popular Python library that is mainly used to perform mathematical and scientific calculations. It offers many features and tools that can be useful for Data Science projects. Becoming familiar with NumPy is an essential step in a Data Science training project.

- The term NumPy is an abbreviation for “Numerical Python “. It is an open-source library in the Python language. It is used for scientific programming in Python, and in particular for programming in Data Science, engineering, mathematics, or science.
- Numpy allows you to create multidimensional arrays containing numbers, and to easily transform them using many mathematical formulas, it has the advantage of speeding up calculations compared to Python

2.5.5 Scikit Learn

Created in 2007, Scikit-learn is widely used in industry and research for various machine learning applications, such as image classification, anomaly detection, value prediction, text analysis, etc. The library is built on top of three other NumPy open source projects,

SciPy and Matplotlib, which are basic scientific libraries for Python. It offers a simple and unified interface for a wide range of templates and provides tools for data preparation and manipulation, model optimization, model evaluation, cross-validation, and more.

Scikit-learn is also known for its detailed documentation, active community, and user-friendly features. The library is constantly evolving with new features and improvements added regularly. Plus, being open source, it's easily accessible for users who want to contribute or customize the library to suit their specific needs.

In summary, scikit-learn is a must-have library for Python developers working in the field of machine learning and data analytics. [30]

2.5.6 Torch

Torch is indeed an open-source machine learning library that was originally released in 2002. It was developed by the New York University research team under the direction of Ronan Collobert, and since then it has been maintained and developed by a community of developers.

The open-source framework gives you maximum flexibility and speed when processing machine learning projects, without unnecessary complexities. Torch is written using the Lua scripting language and uses an underlying C implementation to improve performance. It offers a wide range of algorithms for deep learning, including convolutional neural networks, recurrent neural networks, and multilayer perceptron-like neural networks.

Torch is also known for its flexibility and speed. It offers features such as N-dimensional arrays, linear algebra routines, and numerical optimization routines, as well as efficient GPU support to speed up the training of deep learning models. Torch is also available on different platforms, including iOS and Android.

However, it should be noted that Torch is no longer actively developed and maintained as of 2018 [30].

2.6 Time series data

2.6.1 Definition

Is a series of data points recorded or collected at regular intervals. It is a data type that tracks changes in variables over time, such as sales, stock prices, temperatures, etc. Periodic time intervals can be daily, weekly, monthly, quarterly, or yearly, and the data is typically represented as a line graph or a time series plot. Time series data are commonly used in fields such as economics, finance, weather forecasting, and operations management to analyze trends and patterns and make predictions or forecasts. [32]

Examples Here are some examples in which time series arise:

- Economics and Finance
- Environmental Modelling
- Meteorology and Hydrology
- Demographics
- Medicine
- Engineering

2.6.2 theoretical process of time series

The theoretical process of time series analysis typically involves the following steps:

Data Collection and Preprocessing:

Gather the relevant time series data, ensuring it is accurate and complete.

Clean and preprocess the data, handling any missing values, outliers, or other data quality issues.

Exploratory Data Analysis:

Visualize the time series data to identify any obvious patterns, trends, seasonality, or irregularities.

Calculate descriptive statistics, such as mean, variance, and autocorrelation, to gain insights into the data.

Stationarity Testing: Determine if the time series is stationary, meaning its statistical properties (mean, variance, autocorrelation) do not change over time.

If the time series is not stationary, it needs to be transformed or differenced to achieve stationarity.

Model Identification: Based on the characteristics of the stationary time series, select an appropriate model from the class of linear time series models, such as:

Autoregressive (AR) models

Moving Average (MA) models

Autoregressive Moving Average (ARMA) models

Autoregressive Integrated Moving Average (ARIMA) models

Model Estimation: Estimate the parameters of the selected time series model using techniques like maximum likelihood estimation or least squares.

Model Diagnostics: Assess the adequacy of the fitted model by checking the residuals for white noise (no serial correlation).

If the model is not adequate, revisit the model identification step and try a different model.

Forecasting: Use the fitted model to generate forecasts for future time points.

Evaluate the forecast accuracy using metrics like Mean Squared Error (MSE) or Mean Absolute Percentage Error (MAPE).

Model Refinement and Evaluation: Optimize the model by fine-tuning the parameters or considering additional variables.

Assess the overall performance of the model and its suitability for the specific problem at hand.

2.6.3 Components of time series

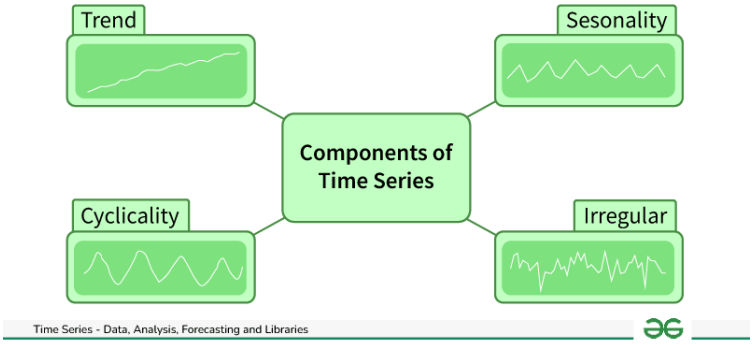


Figure 2-6: components of time series [32]

2.6.3.1 trend

Trend in time series data refers to the long-term upward or downward movement of the data, indicating an overall increase or decrease over time. Trends represent the underlying structure of data and capture the direction and magnitude of change over time. In time series analysis, trends from the data are often modeled and removed to better understand underlying patterns and make more accurate predictions. There are different types of trends in time series data:

- **Upward Trend**: A trend that shows an overall increase over time, where the values of the data tends to rise over time.

- **Downward Trend**: A trend that shows an overall decrease over time, where the values of the data tends to drop over time.

- **Horizontal Trend**: A trend that shows no significant change over time, where the values of the data remain the same (constant) over time.

- **Non-linear Trend**: A trend that shows a more complex pattern of change over time, including upward or downward trends that change direction or magnitude over time.

- **Damped Trend**: A trend that shows a gradual decline in the magnitude of change over time, where the rate of change slows down over time.

It is important to note that time series data can exhibit a combination of these types of trends or multiple trends simultaneously. Accurately identifying and

modeling trends is a critical step in time series analysis as it can significantly impact the forecasts accuracy and interpretation of data patterns.[33]

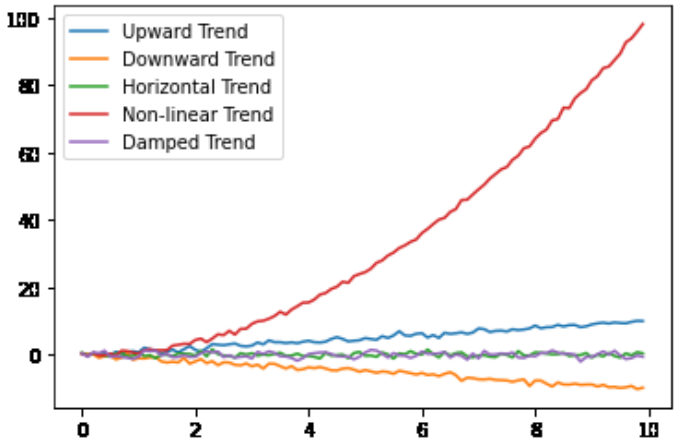


Figure 2-7 Various Trends in Time Series Data generated using python

2.6.3.2 Seasonality

Seasonality in time series data refers to patterns that repeat over a fixed period of time, such as a day, a week, a month, or a year. These patterns arise due to regular events such as holidays, weekends, or seasonal changes, and can be presented in different types of time series data, such as sales, weather, or stock prices.

There are several types of seasonality in time series data, including:

- **Weekly Seasonality**: A type of seasonality that repeats over a 7-day period and is commonly seen in time series data such as sales, energy usage, or transportation patterns.

- **Monthly Seasonality**: A type of seasonality that repeats over a 30- or 31-day period and is commonly seen in time series data such as sales or weather patterns.

- **Annual Seasonality**: A type of seasonality that repeats over a 365- or 366-day period and is commonly seen in time series data such as sales, agriculture, or tourism patterns.

- **Holiday Seasonality**: A type of seasonality that is caused by special events such as holidays, festivals, or sporting events and is commonly seen in time series data such as sales, traffic, or entertainment patterns. [33]

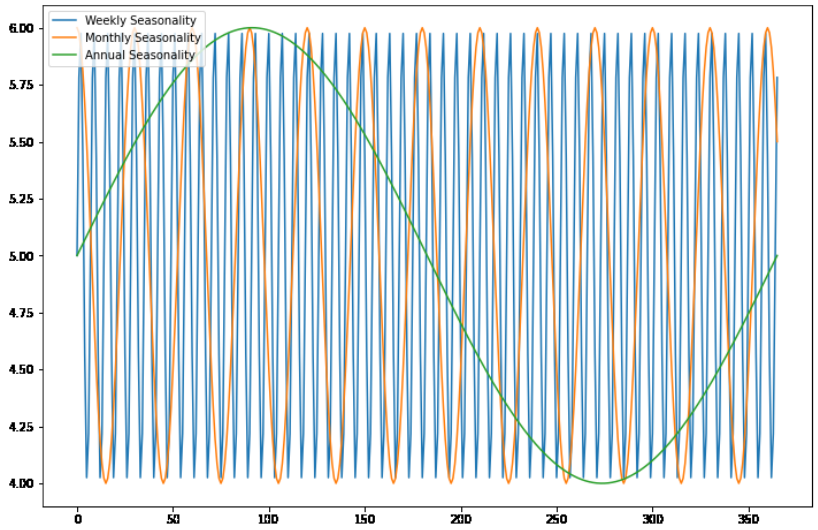


Figure 2-8 Seasonality in Time Series Data generated using python

2.6.3.3 cyclicity

Cyclicity in time series data refers to the repeated patterns or periodic fluctuations that occur in the data over a specific time interval. It can be due to various factors such as seasonality (daily, weekly, monthly, yearly), trends, and other underlying patterns.

- **Difference between Seasonality and Cyclicity:**

Seasonality: refers to a repeating pattern in the data that occurs over a fixed time interval, such as daily, weekly, monthly, or yearly. Seasonality is a predictable and repeating pattern that can be due to various factors such as weather, holidays, and human behavior.

Cyclicity: on the other hand, refers to the repeated patterns or fluctuations that occur in the data over an unspecified time interval. These patterns can be due to various factors such as economic cycles, trends, and other underlying patterns. Cyclicity is not limited to a fixed time interval and can be of different frequencies, making it harder to identify and model.

Putting together, seasonality refers to a repeating pattern in the data that occurs over a fixed time interval, while cyclicity refers to a repeating pattern that occurs over

an unspecified time interval.[33]

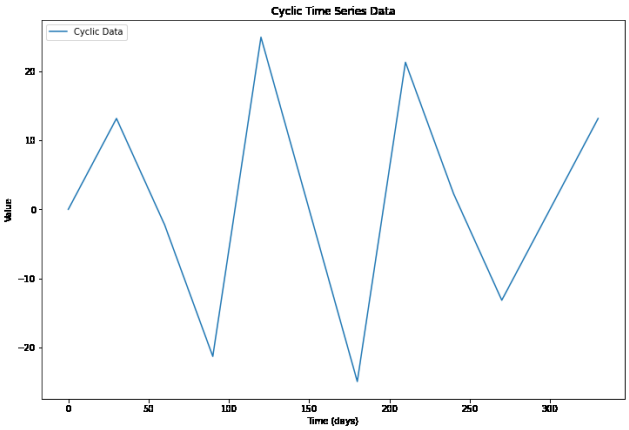


Figure 2-9 Cyclicity in Time Series Data generated using python

2.6.3.4 Irregularities (Residuals)

Irregularities in time series data refer to unexpected or unusual fluctuations in the data that do not follow the general pattern of the data. These fluctuations can occur for various reasons, such as measurement errors, unexpected events, or other sources of noise. Irregularities can have a significant impact on the accuracy of time series models and forecasting, as they can obscure underlying trends and seasonality patterns in the data.

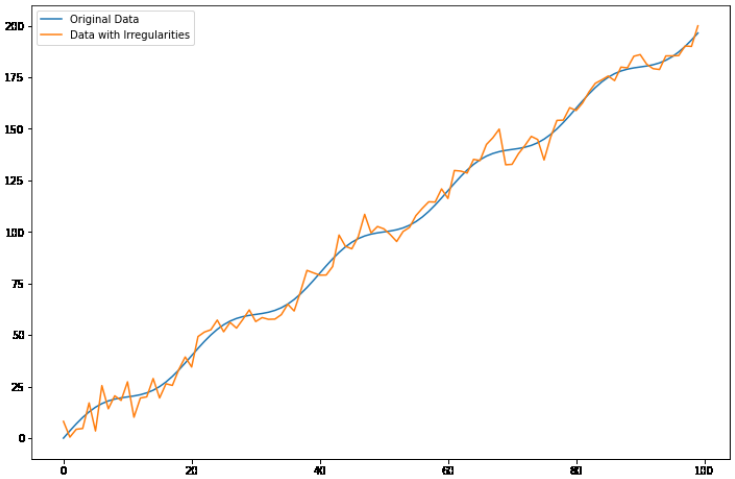


Figure 2-10 Irregularities in Time Series Data generated using python

The plot shows that the irregularities can significantly affect the appearance of the time series data, making it more difficult to identify the underlying patterns.[33]

2.6.3.5 Noise

Random fluctuations or variations that are not the result of an underlying pattern or trend are referred to be noise in time series data. Generally speaking, it is understood to be any erratic and random variance in the data. inaccuracies in data processing or recording, random fluctuations in the underlying process, and measurement inaccuracies are some of the possible causes of these oscillations. The existence of noise might hinder the identification of the fundamental trend or pattern in the data, therefore necessitating the removal or reduction of noise prior to any subsequent analysis.[33]

2.7 Algorithms Used in Time Series

2.7.1 Autoregressive processes

Predicts future values based on linear regression of past observations. Order (p) determines the number of lagged observations used.

In other meaning auto Regressive (AR) model is a specific type of regression model where, the dependent variable depends on past values of itself. This necessarily means that the current values are correlated with values in the previous time-steps. And more specifically, the type of correlation here is partial auto-correlation. The equation for the AR model is shown below

$$Y_t = \beta_1 + \Phi_1 Y_{t-1} + \Phi_2 Y_{t-2} + \ldots + \Phi_p Y_{t-p}$$

AR equation

The respective weights (Φ1, Φ2 …Φp) of the corresponding lagged observations are decided by the correlation between… that lagged observation and the current observation. If the correlation is more, the weight corresponding to that lagged observation is high (and vice-versa).

Notice the **(p)** in the equation…

This **(p)** is called the lag order. It represents the number of prior lag observations we include in the model i.e. the number of lags which have a significant correlation with the current observation.[34]

2.7.2 Moving average processes

Forecasts using weighted sum of past forecast errors. Order (q) represents the number of past errors considered

Which means Moving Average (MA) model works by analyzing how wrong you were in predicting values for the previous time-periods to make a better estimate for the current time-period. Basically, this model factors in errors from the lagged observations. The effects of these previous(lagged) observation errors, on the current observation, depends on the auto-correlation between them. This is similar in sense as the AR model which considers the partial auto-correlations.

$$Y_t = \beta_2 + \omega_1 \varepsilon_{t-1} + \omega_2 \varepsilon_{t-2} + \ldots + \omega_q \varepsilon_{t-q} + \varepsilon_t$$

The ε terms represent the errors observed at respective lags and the weights (ω1, ω2 …ωq) are calculated statistically depending on the correlations. Notice the (q) in the equation… (q) represents the size of the moving window i.e. the number of lag observation errors which have a significant impact on the current observation. It's similar to the lag order(p), but it considers errors instead of the observations themselves.[34]

MA model supplements the AR model by taking into considerations, the errors from the previous time-periods thereby helping to get a better estimate.

When we combine the AR and MA equation, we get

- **ARMA (Autoregressive Moving Average):** Combines autoregressive and moving average components for stationary data

$$Y_t = (\beta_1 + \beta_2) + (\Phi_1 Y_{t-1} + \ldots + \Phi_p Y_{t-p}) + (\omega_1 \varepsilon_{t-1} + \ldots + \omega_q \varepsilon_{t-q} + \varepsilon_t)$$

Combined equation for ARMA

2.7.2.1 Stationarity

The models we've discussed so far (AR and MA) assume that the series is stationary. This is also means that **"stationarity"** is a necessary condition to take advantage of these models for any time series.

Basically, for a time series to be *stationary*, it should satisfy the following 3 conditions

- Mean (μ) is constant
- Standard Deviation (σ) is constant
- Seasonality doesn't exist

In most cases, these conditions can be visually analyzed by studying the plot against time. You will come across a lot of series which are clearly not stationary. Does this mean that forecasting cannot be applied in these cases? Well…this is where the sub-acronym I comes in…

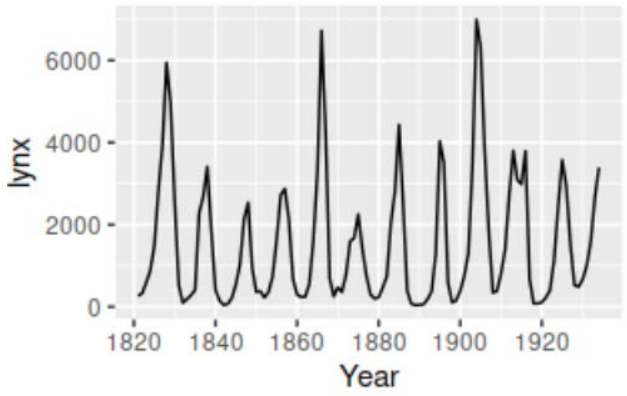


Figure 2-11 Graph describing the population of Lynx relating to a non-periodic model [34]

2.7.2.2 Integrated (I)

Let's say, for example, you come across a series with a **"non-constant"** mean. It's clearly observable that the mean increases over time i.e. the series is not stationary.

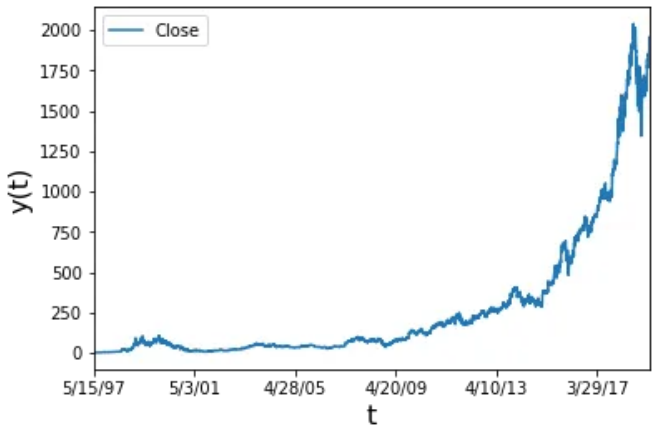


Figure 2-12 Amazon stock value over time

If we could somehow eliminate this upward trend, we'll be good to go.

One way to do this is to consider the differences between consecutive timesteps. This is equivalent to performing a transformation of the form…

$$Z_t = Y_{t+1} - Y_t$$

Transformation

Applying this transformation, we get the following trend with an observable constant mean. The standard deviation is constant as well and seasonality doesn't exist i.e. the series is now stationary.[34]

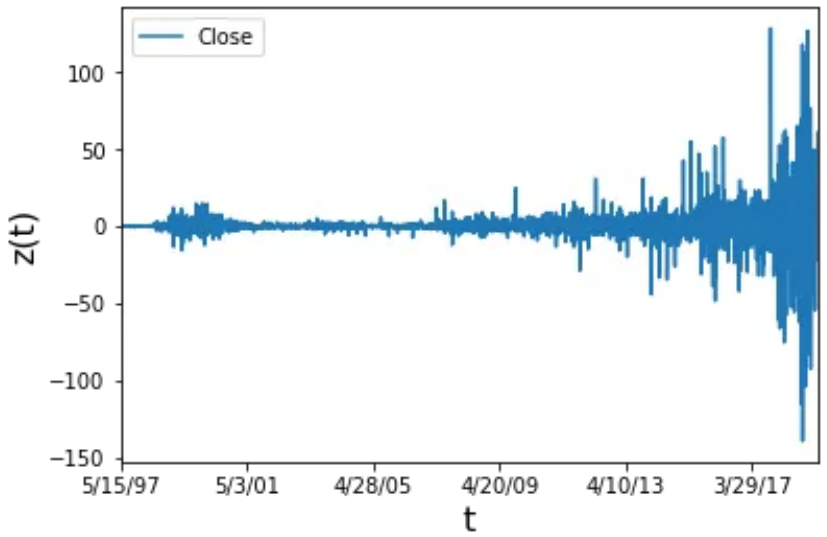


Figure 2-13 Transformation applied

I stand for **Integrated** (though it has nothing to do with integration). It just means that, instead of predicting the time series itself we will predict the differences of the series from one-time step to the next time step.

Notice that here we have taken the first order difference i.e. a single phase of differencing of consecutive terms. This is could do multiple times to make the series stationary.

$$Z_t = Y_{t+1} - Y_t \quad \dots d = 1$$
$$Q_t = Z_{t+1} - Z_t \quad \dots d = 2$$
$$\dots$$

This order of differencing (**d**) is an important parameter of ARIMA and determines the success of the model.[34]

2.7.3 ARIMA (Autoregressive Integrated Moving Average)

Extends ARMA by incorporating differencing to achieve stationarity. Parameters: p (autoregression), d (differencing), q (moving average).

So, to revise, the final ARIMA model will take the following form.[34]

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q}$$

ARIMA model

ARIMA(p, d, q) where,
p => lag order
d => order of differencing
q => size of moving average window
α => represents some value of noise

2.8 Anomaly detection

2.8.1 anomaly detection in time series data

anomaly detection is the process of identifying data points or patterns in a dataset that deviate significantly from the norm. A time series is a collection of data points gathered over some time. Anomaly detection in time series data may be helpful in various industries, including manufacturing, healthcare, and finance. Anomaly detection in time series data may be accomplished using unsupervised learning approaches like clustering, PCA (Principal Component Analysis), and autoencoders [35]

2.8.2 What is an Anomaly Detection Algorithm

Anomaly detection is the process of identifying data points that deviate from the expected patterns in a dataset. Many applications, including fraud detection, intrusion detection, and failure detection, often use anomaly detection techniques. Finding uncommon or very infrequent events that could point to a possible hazard, issue, or opportunity is the aim of anomaly detection.

The autoencoder algorithm is an unsupervised deep learning algorithm that can be used for anomaly detection in time series data. The autoencoder is a neural network that learns to reconstruct its input data By first compressing input data into a lower-dimensional representation and then extending it back to its original dimensions. An autoencoder may be trained on typical time series data to learn a compressed version of the data for anomaly identification. The anomaly score may then be calculated using the reconstruction error between the original and reconstructed data. Anomalies are data points with considerable reconstruction errors [35]

2.8.3 Time Series Data and Anomaly Detection

In the case of time series data, anomaly detection algorithms are especially important since they help us spot odd patterns in the data that would not be obvious from just looking at the raw data. Anomalies in time series data might appear as abrupt increases or decrease

in values, odd patterns, or unexpected seasonality. Time series data is a collection of observations across time.

- Time series data may be used to teach anomaly detection algorithms, such as the autoencoder, how to represent typical patterns. These algorithms can then utilize this representation to find anomalies. The approach can learn a compressed version of the data by training an autoencoder on regular time series data. The anomaly score may then be calculated using the reconstruction error between the original and reconstructed data. Anomalies are data points with considerable reconstruction errors.
- Anomaly detection algorithms may be applied to time series data to find odd patterns that could point to a hazard, issue, or opportunity. For instance, in the context of predictive maintenance, a time series anomaly may point to a prospective equipment failure that may be fixed before it results in a large amount of downtime or safety concerns. Anomalies in time series data may reveal market movements or patterns in financial forecasts that may be capitalized on. [35]

2.9 SVM with kernel methods

2.9.1 **Support Vector Machine** (SVM)

is a powerful supervised machine learning algorithm used for classification and regression tasks. It aims to find the optimal hyperplane that separates data points into different classes with the maximum margin, thereby improving generalization performance. [36]

2.9.1.1 Support Vectors

support vectors are the data points that lie nearest to the hyperplane, or the decision boundary, and have the most effect over the hyperplane's orientation and position. These are the essential data points that establish the ideal margin.[36]

2.9.1.2 Margin types

there are two margin types:

- **hard margin**: The goal of the hard margin support vector machine (SVM) technique is to locate a hyperplane that precisely divides the data points into distinct classes, preventing any potential misclassifications. Hard margin SVM, however, is sensitive to noise and outliers.

- **Soft margin:** presents a penalty parameter C that balances maximizing the margin and minimizing the classification error, allowing for certain misclassifications. Soft margin SVM is more resistant to noise and outliers.

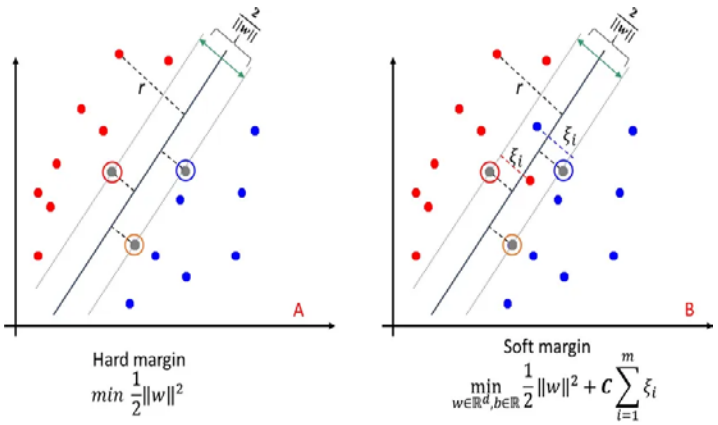


Figure 2-14: hard margin and soft margin [36]

2.9.1.3 Support Vector Classifier (SVC)

- SVC is a specific type of SVM that is used for classification tasks, where the goal is to assign an input to one of multiple discrete classes.
- SVC learns the decision boundaries between the classes by finding the support vectors, which are the data points closest to the decision boundary.

2.9.2 Kernel methods

Kernel methods are popular machine learning algorithms allowing for non-linear transformations or decision functions, among which **support vector machines** are probably the most famous ones and have been successfully used in numerous applications. Kernel methods rely on a kernel function measuring similarity between any pair of inputs.

In a higher-dimensional space, a kernel function K takes two input vectors, x and y, and computes their inner product or similarity. [36]

2.9.2.1 Kernel Functions

- A kernel is a function which determines the similarity between two data points in a high-dimensional space in the context of machine learning, especially in algorithms like Support Vector Machines (SVM).

- Finding a linear separation between classes that may not be linearly separable in the original space is made easier by this technique, which enables SVMs to implicitly transfer input data into a higher-dimensional feature space.[36]

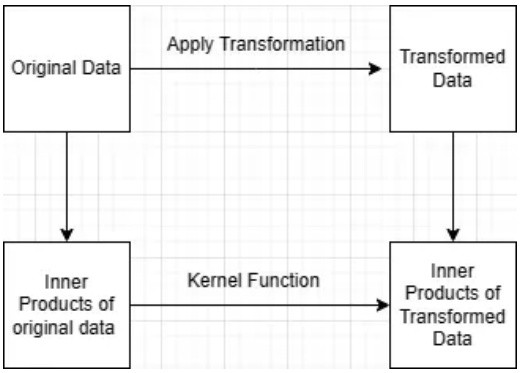


Figure 2-15: process of kernel functions [36]

2.9.2.2 Types of Kernel Functions

- Linear kernel
- Polynomial kernel
- Radial basis function (rbf) kernel
- Sigmoid kernel [37]

2.9.2.3 Comparison of kernel types

Factor	Linear Kernel	Polynomial Kernel	RBF Kernel	Sigmoid Kernel
Speed	Fastest	Moderate	Moderate to Slow	Moderate
Memory Usage	Low	Moderate	Moderate to High	Moderate
Type of Decision Boundary	Linear	Non-linear	Non-linear	Non-linear
Complexity	Low	Moderate to High	High	Moderate to High
Flexibility	Limited	Moderate	High	Moderate
Overfitting Risk	Low	High	Moderate to High	Moderate
Examples	Text classification	Image recognition	Pattern recognition	Time series prediction

Table 2: comparison table of kernel types [37]

2.10 Conclusion

In this chapter we presented in the second part a definition and a view machine learning and algorithms. Before we start building our program of machine learning, choosing one of the many options availabilities can be a difficult task. Therefore, it is important to evaluate several options before making a final decision. In addition, the most important thing is to learn how different machine learning technologies work.

CHAPTER III
Experimental and Numerical Study

3.1 Introduction

Data exploration is indeed a crucial step in the knowledge discovery process. It involves analyzing raw data to find significant patterns and relationships between variables, and we present the results of implementing two learning algorithms (SVM, RNN) and a comparison between them; numerical experiments are performed in two different environment Python and MATLAB. Dataset used are in two forms, first we used sensor data obtained by vibration test bench and secondly, we used surrounding sound of machine in operation captured with a cell phone microphone. In the first section, we describe the development environment used, and we detail its different steps, the algorithms chosen for modeling our objectives, as well as the various tools for evaluating the performance of the produced models. The rest of the chapter is devoted to evaluating the predictive capabilities of the models on the datasets and to comparing their results using appropriate metrics. Finally, the performance obtained is evaluated using the selected models.

3.2 Development environment

3.2.1 Hardware Environment

3.2.1.1 The Machinery Fault Simulator Spectra Quest’s

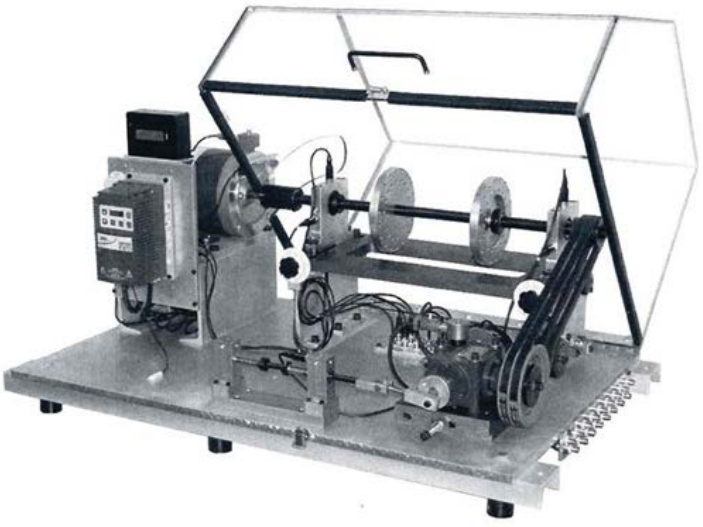


Figure 3-1 The Machinery Fault Simulator

Machinery Fault Simulator is an innovative tool for studying signatures of common machinery faults without compromising factory production or profits. The system, which fits on a desktop and weighs about 150 pounds, reflects a modular design that provides versatility, operational simplicity and robustness. Each component of the simulator is

machined to high tolerances so it can be operated without any significant conflicting vibration. Then, depending on the situation you want to analyze, you can introduce various faults either individually or jointly in a totally controlled environment. The most comprehensive device of its kind on the market, the Machinery Fault Simulator meets the needs of a broad range of vibration analysts, from new to experienced. It is an effective tool for introducing the concepts and methodology of predictive maintenance to engineering students. At the same time, the simulator can be used to train maintenance personnel in the area of predictive maintenance, offering experienced technicians a way to upgrade their job skills and performance.

- **The Benefits of Using the Machinery Fault Simulator** Using the Machinery Fault Simulator offers you a wide range of benefits as you develop your understanding of predictive maintenance and learn to recognize the signatures of various machine faults. The following list indicates many of the ways the simulator can increase your understanding of vibration analysis:

– **Study the vibration spectra of different faults and learn how to recognize them:** Misalignment, Unbalancing, Different types of bearing faults.

– **Learn the concepts of machine condition monitoring and predictive maintenance:** Resonance and critical speed analysis techniques, Phase measurement techniques, Low speed machinery - monitoring procedures.

– **Develop methodologies and protocols for a variety of applications:** Machinery foundation design, Correlation between vibration, motor current and noise spectra, Trouble - shooting machinery problems

3.2.1.2 Laptop computer

Our experiments were carried out on a Desktop computer whose characteristics of the material environment are reported in the table 3

	CHARACTERISTICS
Processor	**Intel(R) Core (TM) i7-10700F CPU @ 2.90GHz 2.90 GHz**
RAM	**64.0 GB**
Operating System	**64-bit operating system, x64-based processor**
Storage	**4 TB**

3.2.2 Software environment

3.2.2.1 Pytorch

PyTorch is a software Type: Software library

Developer: Ronan Colbert and then Facebook research teams.

Environment(s): Linux, Microsoft Windows and MacOS.

Development language: C++, Python, C, and Compute United Device Architecture

Python:

Python is an open-source platform, has become the most widely used programming language in the field of data science and machine learning. Its popularity is due in part to its simple and intuitive syntax, which allows programmers to write code that is clear and easy to understand. Python is also a high-level language, which means that it is more abstract than low-level languages like the C language, making it easier to program. Finally, the version of Python used in this work is version 3.9, which is a stable and widely used version of Python. Python 3.9 was released in Oct.5,2020 and includes many new and improved features compared to previous versions of Python.

Python is a versatile and popular programming language that is used in many applications, such as:

- Application programming.
- The creation of web services.
- Code generation.
- Metaprogramming.

3.3 Description of the data

Here is an experimental experiment on how to obtain data from a vibration simulator in the case of an imbalance: the objective of the experiment: To study the vibration data generated by a simulator when an imbalance is introduced into the system.

- **Used materials:**

- **The principal machine:**

Figure 3-2 Machinery fault simulator Lite (MFS-LT).

- **Vibration sensors (accelerometers)**

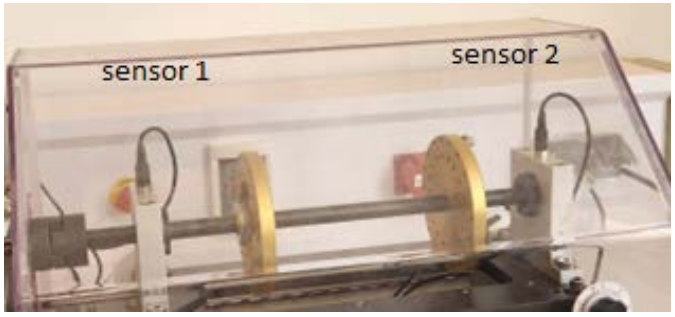


Figure 3-3 Accelerometer sensor.

- **Data Translation DT9837**

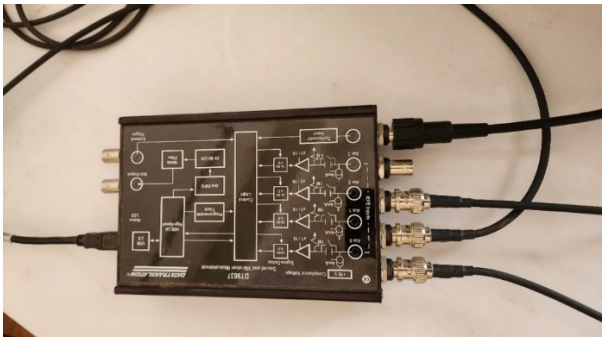


Figure 3-4 Analogue TTL output

- **Data acquisition system**

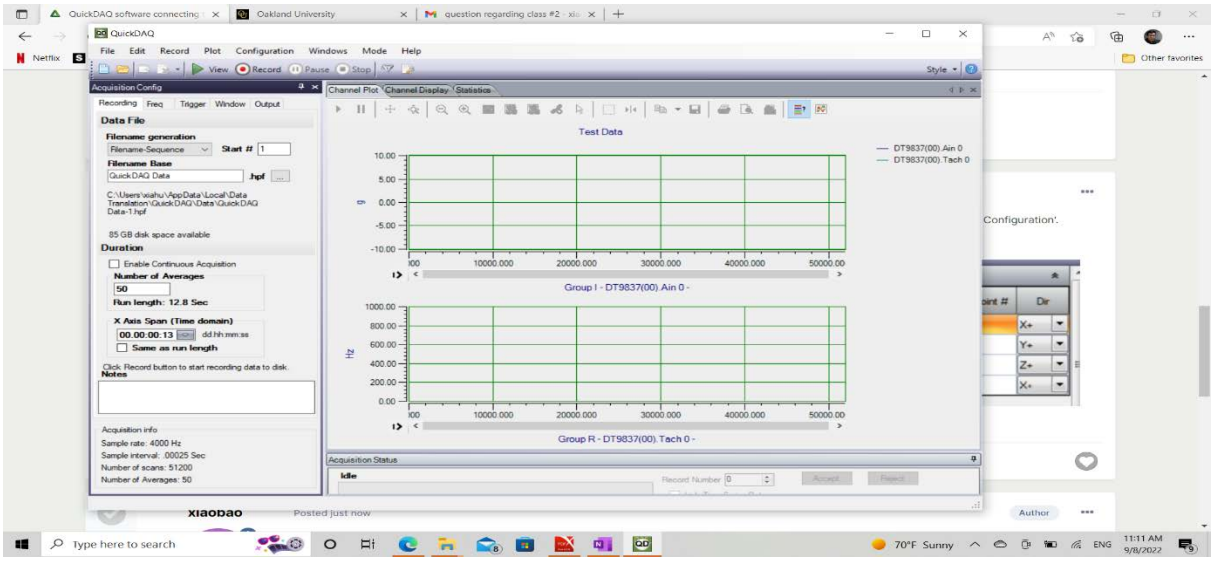


Figure 3-5 Data acquisition system(quickdaq)

- **Tachometer**

Figure 3-6 Built-in tachometer.

The experiment setup:

- **Setting up the vibration simulator:**

Install the vibration simulator and make sure it is working properly.

Place vibration sensors in strategic locations in the simulator to measure vibration.

Connect the sensors to the data acquisition system.

- **Steady-state vibration data collection:**

Start the simulator and let it run for a while in a steady state.

Record vibration data using the data acquisition system.

Analyze the recorded data to establish a baseline of steady-state vibration.

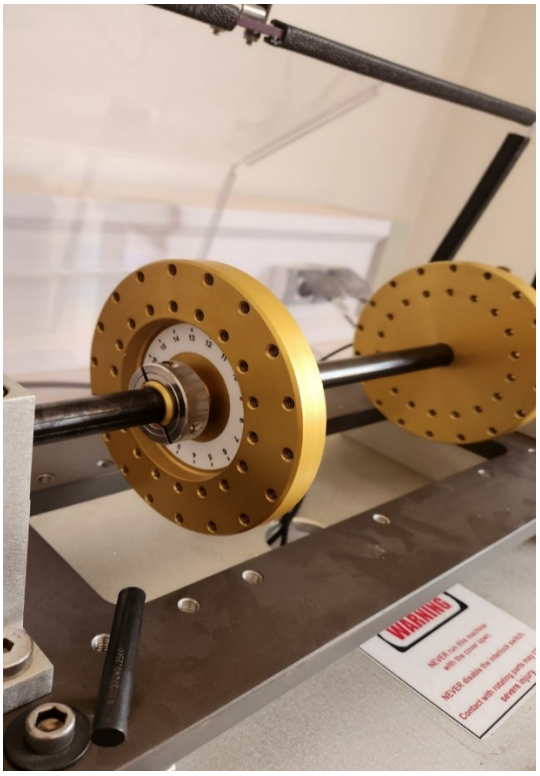


Figure 3-7 Shaft in its healthy state without unbalance

- **Introduction of an unbalance in the system:**

Modify the simulator to introduce an unbalance (for example, by adding extra mass to a specific point). Restart the simulator and let it run for a while with the unbalance.

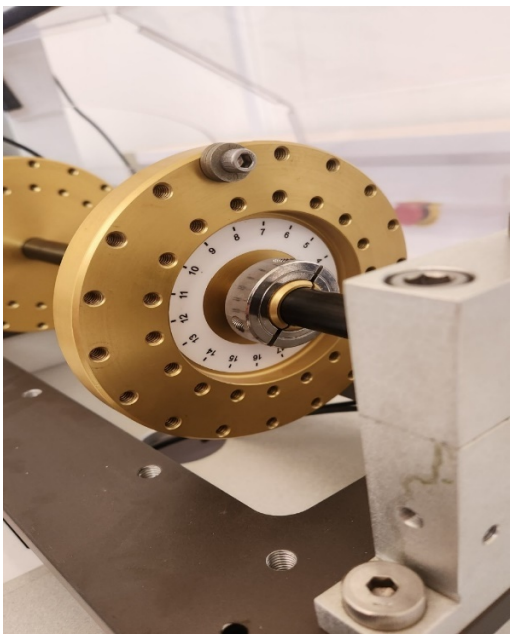


Figure 3-8 Shaft in the state with unbalance

- **Collection of vibration data in the presence of unbalance:**

  Record vibration data again using the data acquisition system.

  Compare vibration data with that recorded at steady state

- **Data analysis:**

  Analyze the vibration data recorded in both cases (balance and unbalance).

  Identify differences in vibration characteristics, such as amplitude, frequency, etc.

  Interpret the results to understand the impact of the unbalance on system vibration.

```
QuickDAQ Data
01/01/0001 00:00:00
Notes:
Sample Rate:    1000,11596       Hz
Measurement Type        Time Waveform   Time Waveform   Tachometer RPM Data
Channel Name    DT9837(00).Ain 0        DT9837(00).Ain 1        DT9837(00).Tach 0
X Axis Units    Sec     Sec     Sec
Y Axis Units    g       g       RPM
Seconds Real    Real    Real
 0,00000E+000   -0,0067993447    0,0028340322   1116,0299096016
 9,99884E-004   -0,0398357709    0,0217838107   1116,0299096016
 0,0019997681    0,0119540426    0,0095929740   1116,0299096016
 0,0029996522    0,0174067639    0,0195120865   1116,0299096016
 0,0039995362   -0,0126604681   -0,0157446231   1116,0299096016
 0,0049994203    0,0405532342    0,0185786553   1116,0299096016
 0,0059993043   -0,0082121955   -0,0094355277   1116,0299096016
 0,0069991884    0,0059825403    0,0361788948   1116,0299096016
 0,0079990725    0,0362926059    0,0113473748   1116,0299096016
 0,0089989565    0,0397033162    0,0140801916   1116,0299096016
 0,0099988406    0,0390962318    0,0290263374   1116,0299096016
 0,0109987246   -0,0090400378    0,0370898337   1116,0299096016
 0,0119986087    0,0728390835    0,0376633878   1116,0299096016
 0,0129984928    0,0460942586    0,0198044867   1116,0299096016
 0,0139983768    0,0089738104    0,0292062759   1116,0299096016
 0,0149982609    0,0405753100    0,0171953777   1116,0299096016
 0,0159981449    0,0740863659    0,0175440087   1116,0299096016
 0,0169980290    0,0449573552    0,0307245075   1116,0299096016
 0,0179979130   -0,0016556846    0,0698498960   1116,0299096016
 0,0189977971    0,0793183291    0,0298360609   1116,0299096016
 0,0199976812    0,0286323053    0,0044759714   1116,0299096016
 0,0209975652    0,0443392330    0,0194108711   1116,0299096016
 0.0219974493    0.0391624592    0.0311743538   1116.0299096016
```

Figure 3-9 Typical data from the acquisition system

3.4 Data preprocessing

Alright, now that we've collected and analyzed the vibration data from the simulator in the

presence of an unbalance, we can move on to the next step of developing a machine learning model to detect and predict unbalances. Here are the steps to follow:

Data preparation

Organize the collected vibration data in an appropriate format (text file).

Identify key characteristics of vibration data, such as amplitude, frequency, time domain, frequency domain, etc.

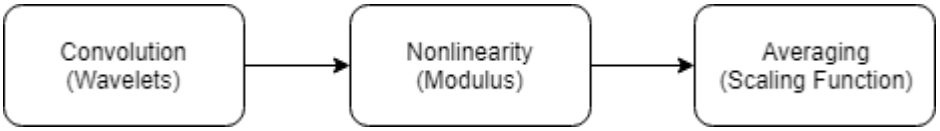
Label data according to the state of the system (unbalance) after that, we put our data into filtering tool, and the tool that we used in this case is the wavelets scattering and well known in the field of programming by (Kymatio): is an implementation of the wavelet scattering transform in the Python programming language, suitable for large-scale numerical experiments in signal processing and machine learning. Scattering transforms are translation-invariant signal representations implemented as convolutional networks whose filters are not learned, but fixed (as wavelet filters).

In summary, the wavelet scattering transform plays a crucial role in data analysis and machine learning by providing a stable, invariant, and informative feature representation of the input data. Its ability to capture local and global patterns, as well as its interpretability and potential for multimodal integration, make it a valuable tool in a wide range of applications

Wavelet scattering

Wavelet scattering transform is a powerful mathematical tool designed for signal processing and feature extraction, particularly in contexts where traditional methods like Fourier transforms fall short. It combines wavelet transforms with deep learning principles to capture multi-scale, hierarchical structures in data, making it highly effective for analyzing time-series, audio, and image data. The transform operates by repeatedly applying wavelet convolutions followed by modulus and averaging operations, which helps in preserving the essential structure of the signal while being invariant to translations and stable to deformations. Unlike deep neural networks, wavelet scattering transforms are non-parametric and do not require training, thus offering a robust and interpretable framework. This method has been successfully applied in various domains, including texture classification, signal denoising, and anomaly detection, showcasing its versatility and efficiency. The illustrations below depict the multi-scale decomposition of an image using wavelet filters and the hierarchical representation of features extracted through scattering layer

wavelet scattering transform processes data in stages. The output of one stage becomes input for the next stage. Each stage consists of three operations.



The zeroth-order scattering coefficients are computed by simple averaging of the input. Here is a tree view of the algorithm:

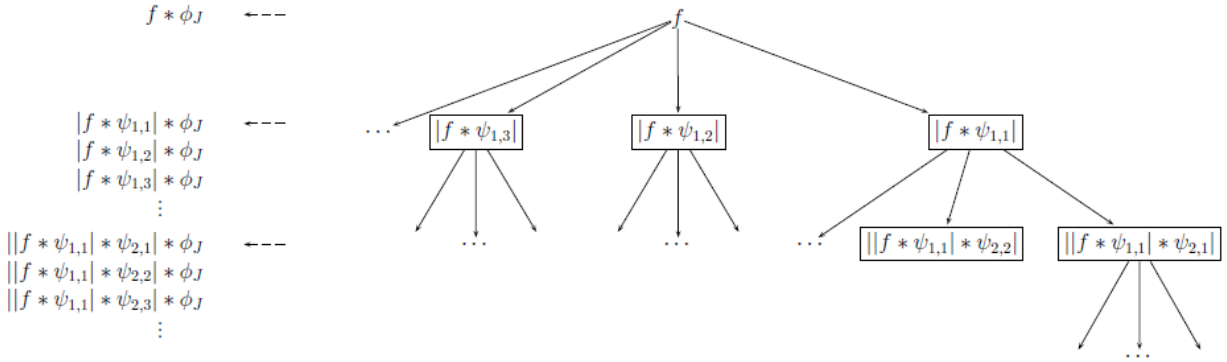


Figure 3.10 Scattering transform of signal f by iterating the {ψj,k} operator. The output nodes are shown as squares.

The {ψj,k} are wavelets, ϕJ is the scaling function, and f is the input data. In the case of image data, for each ψj,k, there are a number of user-specified rotations of the wavelet. A sequence of edges from the root to a node is referred to as a *path*. The tree nodes are the *scalogram coefficients*. The *scattering coefficients* are the scalogram coefficients convolved with the scaling function ϕJ. The set of scattering coefficients are the low-variance features derived from the data. Convolution with the scaling function is lowpass filtering and information is lost. However, the information is recovered when computing the coefficients in the next stage

The scattering transform generates features in an iterative fashion. First, you convolve the data with the scaling function, f∗ϕJ to obtain S[0], the zeroth-order scattering coefficients. Next, proceed as follows:

1. Take the wavelet transform of the input data with each wavelet filter in the first filter bank.
2. Take the modulus of each of the filtered outputs. The nodes are the scalogram, U[1].
3. Average each of the moduli with the scaling filter. The results are the first-order scattering coefficients, S[1].

Repeat the process at every node

Invariance Scale

The scaling filter plays a crucial role in the wavelet scattering network. When you create a wavelet scattering network, you specify the invariance scale. The network is invariant to

translations up to the invariance scale. The support of the scaling function determines the size of the invariant in time or space.

Time Invariance

For time series data, the invariance scale is a duration. The time support of the scaling function does not exceed the size of the invariant. This plot shows the support of the scaling function in a network with an invariance scale of two seconds and a sampling frequency of 100 Hz. Also shown are the real and imaginary parts of the coarsest-scale wavelet from the first filter bank. Observe the time supports of the functions do not exceed two seconds.

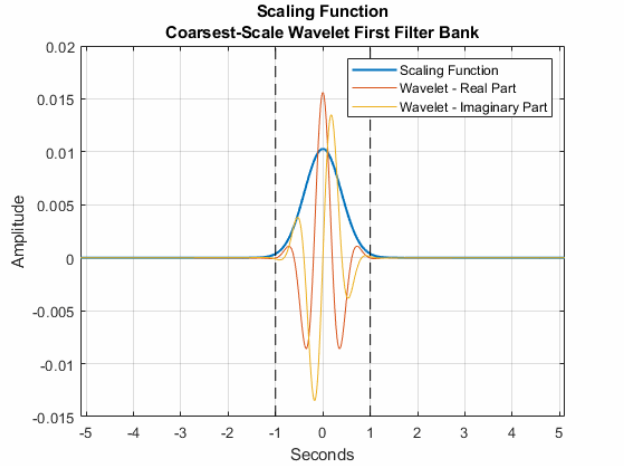


Figure 3.11 First filter bank of a wavelet scattering transform (WST) [39].

The invariance scale also affects the spacings of the center frequencies of the wavelets in the filter banks. In a filter bank, the bandpass center frequencies are logarithmically spaced and the bandwidths of the wavelets decrease with center frequency.

In a scattering network, however, the time support of a wavelet cannot exceed the invariance scale. This property is illustrated in the coarsest-scale wavelet plot. Frequencies lower than the invariant scale are linearly spaced with scale held constant so that the size of the invariant is not exceeded. The next plot shows the center frequencies of the wavelets in the first filter bank in the scattering network. The center frequencies are plotted on linear and logarithmic scales. Note the logarithmic spacing of the higher center frequencies and the linear spacing of the lower center frequencies

3.5 Proposed approach

- **Machine Learning Model Selection:**

  Choose a machine learning algorithm that is appropriate for your problem, such as

regression, classification, or reinforcement learning.

Consider models such as recurrent neural networks (RNN), support vector machines (SVMs), or decision trees, depending on the complexity of our problem and the expected performance.

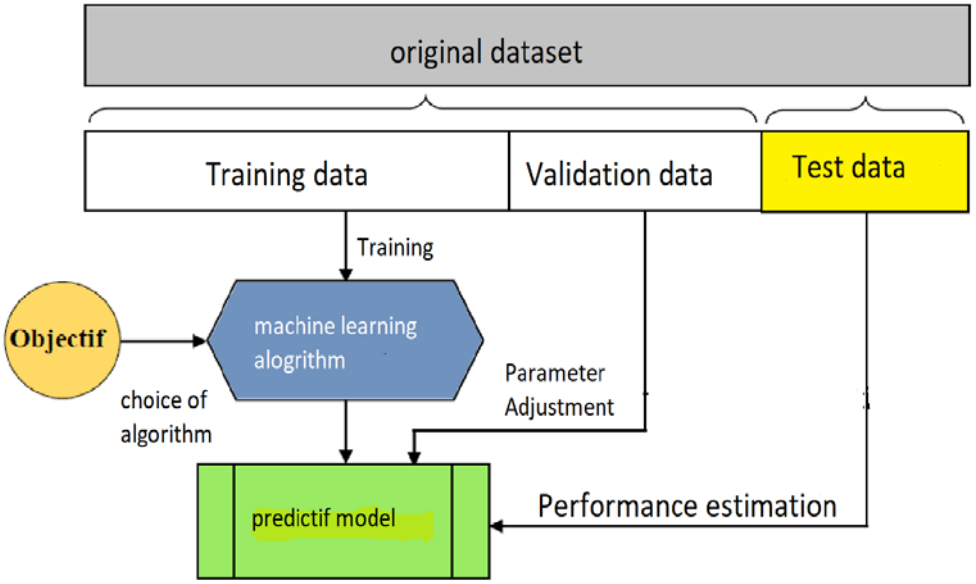


Figure 3-12 P*rocesses for model creation*

- **Model Evaluation:**

Evaluate the model's performance on the test set.

Calculate relevant evaluation metrics, such as accuracy, recall, F1-score

Identify the strengths and weaknesses of the model.

We can evaluate the logistic regression model using the following metrics:

• **Accuracy:** provides the proportion of correctly classified instances.

$$Accuracy = \frac{True\ Positives + true\ negative}{total}$$

• **Precision:** focuses on the accuracy of positive predictions.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

• **Recall (Sensitivity or True Positive Rate):** measures the proportion of correctly predicted positive instances among all actual positive instances.

$$\text{Recall} = \frac{True\ Positives}{True\ Positives + False\ \text{Negatives}}$$

F1Score: is the harmonic mean of precision and recall.

$$\mathrm{F1-score} = 2 * \frac{\mathrm{Precision} * \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

- **Model Improvement:**

  If the model's performance is not satisfactory, consider enhancement techniques, such as adding new features, modifying the model architecture, or using data augmentation techniques.

  Iterate on the training and evaluation process until performance is satisfactory.

- **Model deployment:**

  Once the model has reached acceptable performance, it can be deployed to detect and predict unbalances in the vibration simulator.

  Integrate the model into a real-time monitoring system or decision support tool for simulator operators.

3.6 Machine Learning Algorithms

The object is uses SVM and LRC [logistic regression classifier] model to classify vibration data into equilibrium or disequilibrium by classifying each vibration on her own class

3.6.1 Support vector machine

Support Vector Machine (SVM) is a powerful machine learning algorithm used for linear or nonlinear classification, regression, and even outlier detection tasks. SVMs can be used for a variety of tasks, such as text classification, image classification, spam detection, handwriting identification, gene expression analysis, face detection, and anomaly detection. SVMs are adaptable and efficient in a variety of applications because they can manage high-dimensional data and nonlinear relationships.

SVM algorithms are very effective as we try to find the maximum separating hyperplane between the different classes available in the target feature [38] , for more details refer to chapter II (Machine learning).

3.6.2Logistic regression classifier

Logistic regression is a supervised machine learning algorithm used for classification tasks where the goal is to predict the probability that an instance belongs to a given class or not. Logistic regression is a statistical algorithm which analyze the relationship between two data factors. The article explores the fundamentals of logistic regression, it's types and implementation [38]

3.7 Results with python

The final results of the data classification have two forms, the first form on text file of capture one and the second form on wave file and also, we did our test on two different programs MATLAB and pytorch

3.7.1 SVM Classification (sensor 1)

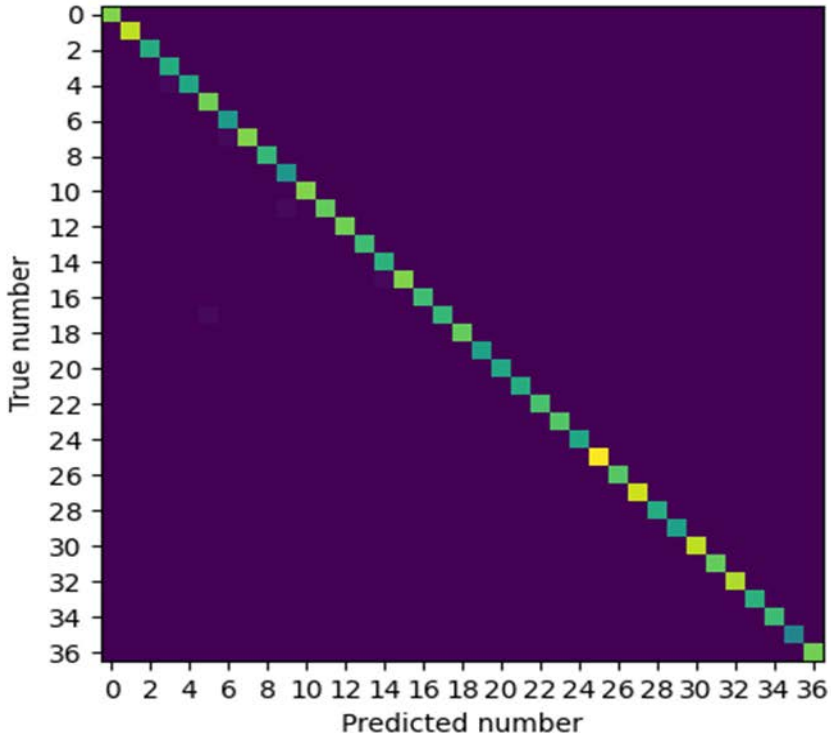


Figure 3-11 Confusion matrix classification results of unbalance classification for sensor 1 data with SVM classification in python

Confusion Matrix:

```
 [[34  0  0 ...  0  0  0]
 [ 0 38  0 ...  0  0  0]
 [ 0  0 26 ...  0  0  0]
 ...
 [ 0  0  0 ... 29  0  0]
 [ 0  0  0 ...  0 19  0]
 [ 0  0  0 ...  0  0 33]]
Accuracy: 99.54954954954955 %
Precision: [100.         100.         100.         96.2962963  100.
  97.05882353  95.83333333 100.         100.         95.65217391
 100.         100.         100.         100.         96.42857143
 100.         100.         100.         100.         100.
 100.         100.         100.         100.         100.
 100.         100.         100.         100.         100.
 100.         100.        ] %
Recall: [100.         100.         100.         100.         96.15384615
 100.         100.         97.14285714 100.         100.
 100.         96.96969697 100.         100.         100.
  97.14285714 100.         96.55172414 100.         100.
 100.         100.         100.         100.         100.
 100.         100.         100.         100.         100.
 100.         100.         100.         100.         100.
 100.         100.        ] %

F1-Score: [100.         100.         100.         98.11320755  98.03921569
  98.50746269  97.87234043  98.55072464 100.         97.77777778
 100.         98.46153846 100.         100.         98.18181818
  98.55072464 100.         98.24561404 100.         100.
 100.         100.         100.         100.         100.
 100.         100.         100.         100.         100.
 100.         100.         100.         100.         100.
 100.         100.        ] %
Execution time: 0.7142 seconds
```

3.7.2SVM Classification (phone sound)

initial data vector
torch.Size([5550, 65536])
torch.Size([5550])
training vector dimension xtr ytr 4440 80% of 5550
test vector dimension xtr ytr 1110 20% of 5550
qfter zqvelet scqttering
Sx_all.shape torch.Size ([5550, 336])
y_all.shape torch.Size([5550])
Sx_tr.shape torch.Size([4440, 336])
y_tr.shape torch.Size([4440])
Sx_te.shape torch.Size([1110, 336])
y_te.shape torch.Size([1110])

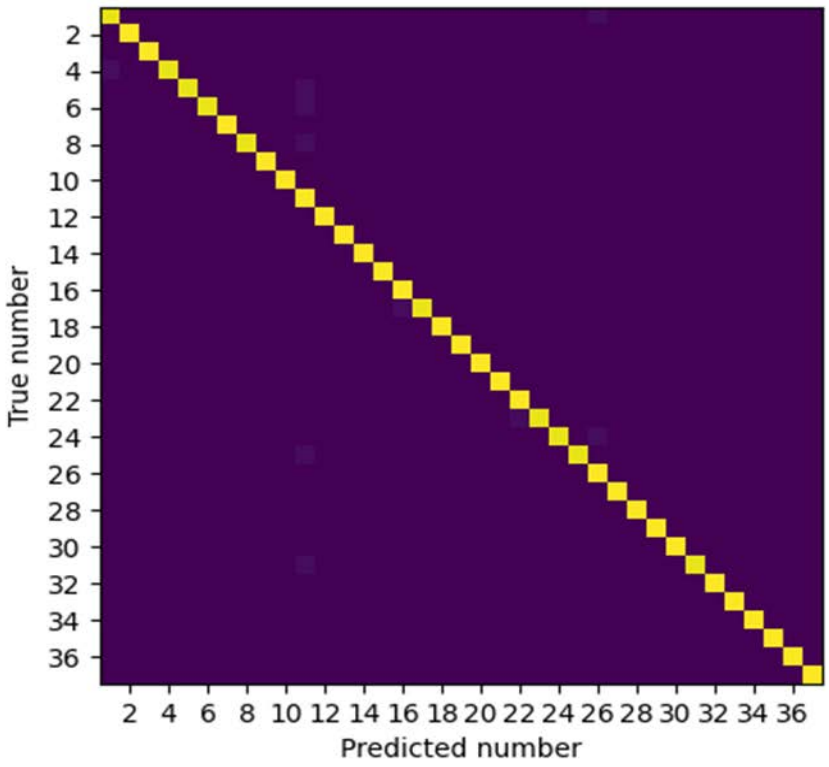


Figure 3-12 Confusion matrix classification results of unbalance classification for machine sound recorded with cell phone data with SVM classification in python

Epoch 0, average loss = 0.985, accuracy = 0.857
Epoch 1, average loss = 0.563, accuracy = 0.957
Epoch 2, average loss = 0.382, accuracy = 0.970
Epoch 3, average loss = 0.279, accuracy = 0.989
Epoch 4, average loss = 0.215, accuracy = 0.993
Epoch 5, average loss = 0.182, accuracy = 0.993
Epoch 6, average loss = 0.147, accuracy = 0.993
Epoch 7, average loss = 0.132, accuracy = 0.993
Epoch 8, average loss = 0.108, accuracy = 0.996
Epoch 9, average loss = 0.094, accuracy = 0.998
Epoch 10, average loss = 0.080, accuracy = 0.998
Epoch 11, average loss = 0.071, accuracy = 0.998
Epoch 12, average loss = 0.064, accuracy = 0.998
Epoch 13, average loss = 0.057, accuracy = 0.999
Epoch 14, average loss = 0.052, accuracy = 0.999
Epoch 15, average loss = 0.047, accuracy = 0.999
Epoch 16, average loss = 0.042, accuracy = 0.999
Epoch 17, average loss = 0.041, accuracy = 0.998
Epoch 18, average loss = 0.036, accuracy = 0.999

Epoch 19, average loss = 0.034, accuracy = 0.999
Epoch 20, average loss = 0.029, accuracy = 0.999
Epoch 21, average loss = 0.029, accuracy = 0.999
Epoch 22, average loss = 0.026, accuracy = 0.999
Epoch 23, average loss = 0.024, accuracy = 0.999
Epoch 24, average loss = 0.022, accuracy = 1.000
Epoch 25, average loss = 0.024, accuracy = 0.999
Epoch 26, average loss = 0.021, accuracy = 0.999
Epoch 27, average loss = 0.018, accuracy = 1.000
Epoch 28, average loss = 0.017, accuracy = 1.000
Epoch 29, average loss = 0.015, accuracy = 1.000
Epoch 30, average loss = 0.015, accuracy = 1.000
Epoch 31, average loss = 0.014, accuracy = 1.000
Epoch 32, average loss = 0.013, accuracy = 1.000
Epoch 33, average loss = 0.012, accuracy = 1.000
Epoch 34, average loss = 0.013, accuracy = 1.000
Epoch 35, average loss = 0.010, accuracy = 1.000
Epoch 36, average loss = 0.010, accuracy = 1.000
Epoch 37, average loss = 0.009, accuracy = 1.000
Epoch 38, average loss = 0.009, accuracy = 1.000
Epoch 39, average loss = 0.008, accuracy = 1.000
Epoch 40, average loss = 0.008, accuracy = 1.000
Epoch 41, average loss = 0.007, accuracy = 1.000
Epoch 42, average loss = 0.007, accuracy = 1.000
Epoch 43, average loss = 0.007, accuracy = 1.000
Epoch 44, average loss = 0.006, accuracy = 1.000
Epoch 45, average loss = 0.006, accuracy = 1.000
Epoch 46, average loss = 0.005, accuracy = 1.000
Epoch 47, average loss = 0.005, accuracy = 1.000
Epoch 48, average loss = 0.005, accuracy = 1.000
Epoch 49, average loss = 0.005, accuracy = 1.000
TEST, average loss = 0.210, accuracy = 0.991

Confusion Matrix:
[[29 0 0 ... 0 0 0]
[0 30 0 ... 0 0 0]
[0 0 30 ... 0 0 0]
...
[0 0 0 ... 30 0 0]
[0 0 0 ... 0 30 0]
[0 0 0 ... 0 0 30]]

Accuracy: 0.9910
Class 0: Precision: 0.9667, Recall: 0.9667, F1-score: 0.9667
Class 1: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000

Class 2: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 3: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 4: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 5: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 6: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 7: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 8: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 9: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 10: Precision: 0.8571, Recall: 1.0000, F1-score: 0.9231
Class 11: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 12: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 13: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 14: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 15: Precision: 0.9677, Recall: 1.0000, F1-score: 0.9836
Class 16: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 17: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 18: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 19: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 20: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 21: Precision: 0.9677, Recall: 1.0000, F1-score: 0.9836
Class 22: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 23: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 24: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 25: Precision: 0.9375, Recall: 1.0000, F1-score: 0.9677
Class 26: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 27: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 28: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 29: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 30: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 31: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 32: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 33: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 34: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 35: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 36: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000

Class 0 correspond to no unbalance added
Class 1 to class 18 correspond to unbalance in Disk1 at positions 1 to 18 respectively
Class 19 to class 36 correspond to unbalance in Disk2 at positions 1 to 18 respectively
Process finished with exit code 0

3.7.3Logistic Regression Classifier (sensor 1)

Elapsed time Reading: 1 min 19 sec

Elapsed time scattering:	955.01 seconds	15 min 52 sec
Elapsed time Saving:		0.70 seconds
Logistic Regression Execution time:	316.77 seconds	8 min 30 sec
Evaluation Time		0.35 sec
Total time		25 min 42 sec

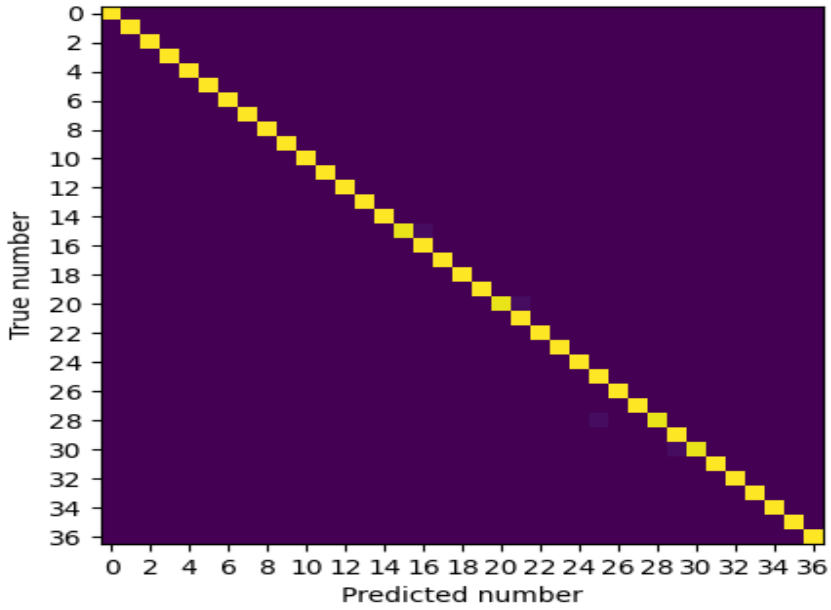


Figure 3-13 Confusion matrix classification results of unbalance classification for sensor 1 data with Logistic regression classifier in python

Confusion Matrix:
[[30 0 0 ... 0 0 0]
[0 30 0 ... 0 0 0]
[0 0 30 ... 0 0 0]
...
[0 0 0 ... 30 0 0]
[0 0 0 ... 0 30 0]
[0 0 0 ... 0 0 30]]

Accuracy: 0.9964

Class 0: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 1: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 2: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 3: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 4: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 5: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 6: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 7: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 8: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 9: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 10: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 11: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 12: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000

Class 13: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 14: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 15: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 16: Precision: 0.9677, Recall: 1.0000, F1-score: 0.9836
Class 17: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 18: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 19: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 20: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 21: Precision: 0.9677, Recall: 1.0000, F1-score: 0.9836
Class 22: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 23: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 24: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 25: Precision: 0.9677, Recall: 1.0000, F1-score: 0.9836
Class 26: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 27: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 28: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 29: Precision: 0.9677, Recall: 1.0000, F1-score: 0.9836
Class 30: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 31: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 32: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 33: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 34: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 35: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 36: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000

Class 0 correspond to no unbalance added

Class 1 to class 18 correspond to unbalance in Disk1 at positions 1 to 18 respectively

Class 19 to class 36 correspond to unbalance in Disk2 at positions 1 to 18 respectively

3.7.4Logistic Regression Classifier (phone sound)

Elapsed time Reading:	14.69 seconds
Elapsed time scattering: 955.01 seconds	15 min 55 sec
Elapsed time Saving:	0.71 seconds
Logistic Regression Execution time: 316.77 seconds	5 min 17 sec
Total time	21 min 27 sec

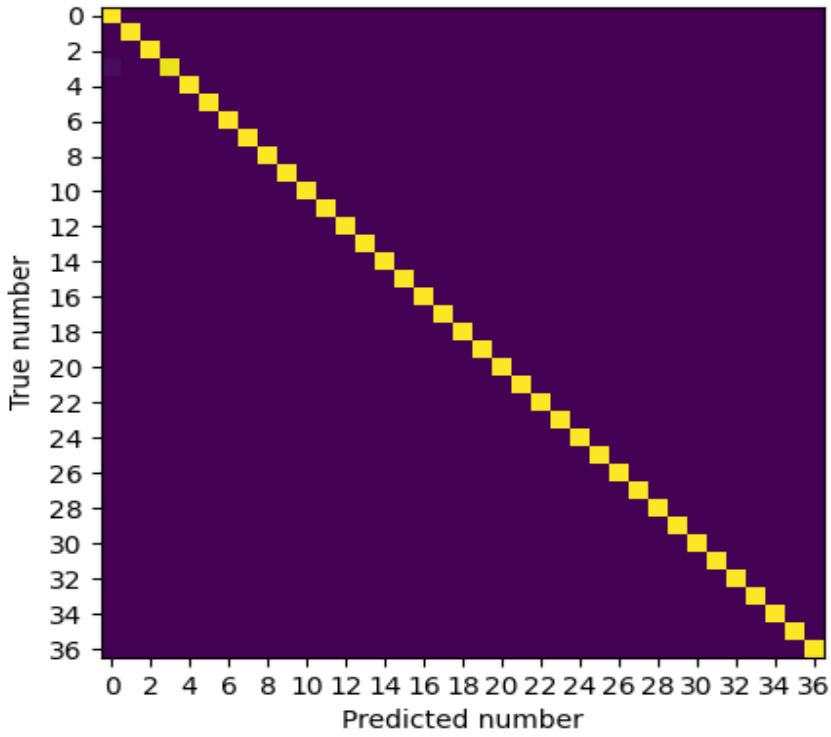


Figure 3-14 Confusion matrix classification results of unbalance classification for machine sound recorded with cell phone data with Logistic regression classifier in python

Epoch 138000, Loss: 0.0050

Confusion Matrix:

```
[[30  0  0 ...  0  0  0]
 [ 0 30  0 ...  0  0  0]
 [ 0  0 30 ...  0  0  0]
 ...
 [ 0  0  0 ... 30  0  0]
 [ 0  0  0 ...  0 30  0]
 [ 0  0  0 ...  0  0 30]]
```

Accuracy: 0.9991

Class 0: Precision: 0.9677, Recall: 1.0000, F1-score: 0.9836
Class 1: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 2: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 3: Precision: 1.0000, Recall: 0.9667, F1-score: 0.9831
Class 4: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 5: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 6: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 7: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000

Class 8: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 9: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 10: Precision: 1.0000, Recall: 1.0000, F1-score: 1.000
Class 11: Precision: 1.0000, Recall: 1.0000, F1-score: 1.000
Class 12: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 13: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 14: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 15: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 16: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 17: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 18: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 19: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 20: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 21: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 22: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 23: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 24: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 25: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 26: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 27: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 28: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 29: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 30: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 31: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 32: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 33: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 34: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 35: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000
Class 36: Precision: 1.0000, Recall: 1.0000, F1-score: 1.0000

Class 0 correspond to no unbalance added
Class 1 to class 18 correspond to unbalance in Disk1 at positions 1 to 18 respectively
Class 19 to class 36 correspond to unbalance in Disk2 at positions 1 to 18 respectively

3.8 Results with Matlab

3.8.1Neural Network sensor 1

```
tic;  % Start the timer
datasetLocation = fullfile('H:\vibration data\','wavfrommat','');
ads = audioDatastore(datasetLocation,'IncludeSubfolders',true,...
    'LabelSource','foldernames');
```

ans = 411

```
rng default
ads = shuffle(ads);
[adsTrain,adsTest] = splitEachLabel(ads,0.8,0.2);
```

```
uniqueLabels = unique(adsTrain.Labels);
tblTrain = countEachLabel(adsTrain);
tblTest = countEachLabel(adsTest);
H = bar(uniqueLabels,[tblTrain.Count, tblTest.Count],'stacked');
legend(H,["Training Set","Test Set"],'Location','NorthEastOutside')
```

Split the data into training and test sets. Use 80% of the data for training and hold out the remaining 20% for testing. Shuffle the data once before splitting

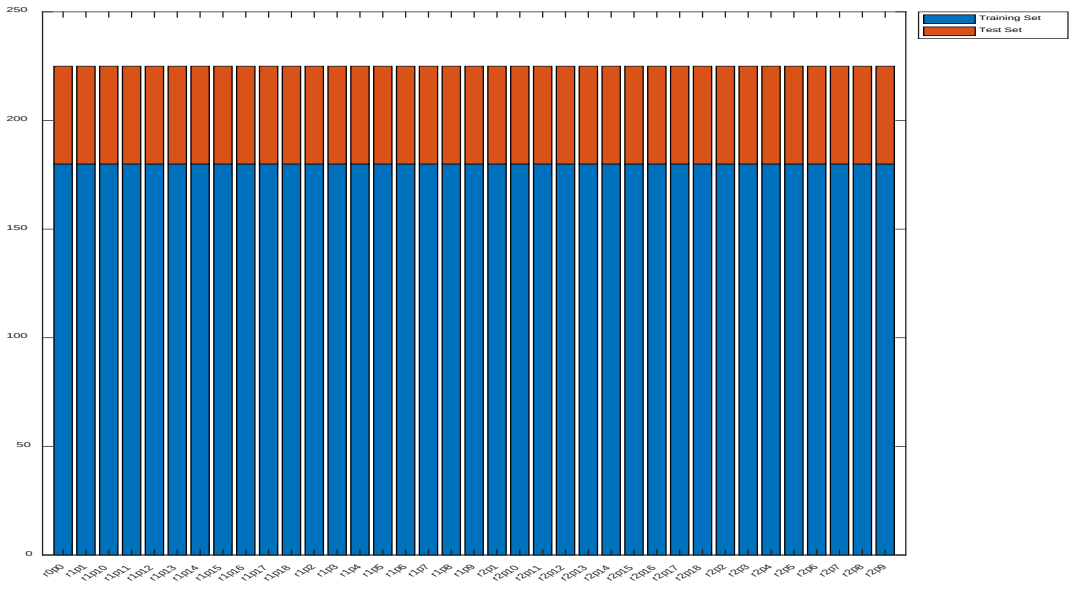


Figure 3-15 Distribution Data of training and test

```
idx = randperm(numel(adsTrain.Files),8);
Fs = 44100;
for n = 1:numel(idx)
    x = audioread(adsTrain.Files{idx(n)});
    t = (0:size(x,1)-1)/Fs;
    subplot(4,2,n);
    fprintf('size(x) is [%s]\n', int2str(size(x)))
```

```
    plot(t,x);
    if n == 7 || n == 8
        xlabel('Seconds');
    end
    title(string(adsTrain.Labels(idx(n))));
end
```

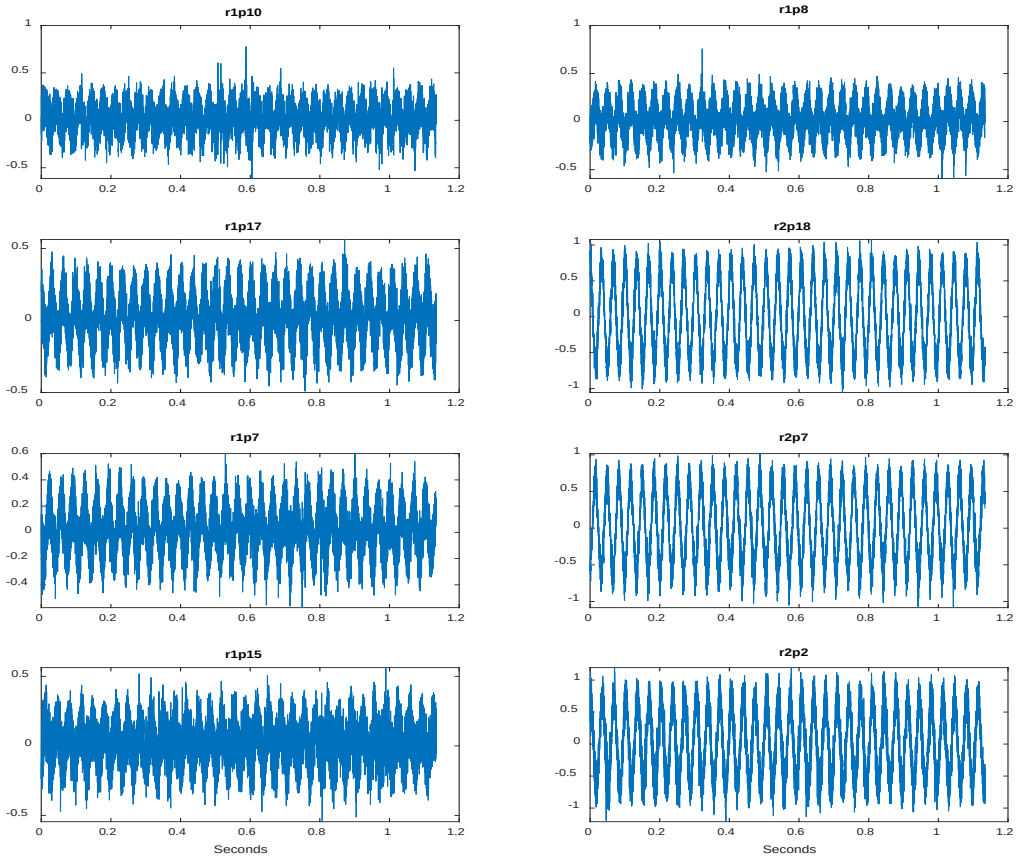


Figure 3-16 Signal for sensor 1 for different unbalance positions

Training stopped: Max epochs completed

```
N = 5e4;
Fs = 44100;
IS = 0.5;
sn = waveletScattering('SignalLength',N,'SamplingFrequency',Fs,...
    'InvarianceScale',0.5);
```

```
[~,numpaths] = paths(sn);
Ncfs = numCoefficients(sn);
sum(numpaths)
```

```
batchsize = 64;
```

```
useGPU = false;
scTrain = [];
while hasdata(adsTrain)
    sc = helperBatchScatFeatures(adsTrain,sn,N,batchsize,useGPU);
    scTrain = cat(3,scTrain,sc);
end
```

```
scTest = [];
while hasdata(adsTest)
   sc = helperBatchScatFeatures(adsTest,sn,N,batchsize,useGPU);
   scTest = cat(3,scTest,sc);
end
```

```
TrainFeatures = scTrain(2:end,:,:);
TrainFeatures = squeeze(num2cell(TrainFeatures,[1 2]));
YTrain = adsTrain.Labels;
TestFeatures = scTest(2:end,:,:);
TestFeatures = squeeze(num2cell(TestFeatures,[1 2]));
YTest = adsTest.Labels;
```

```
[inputSize, ~] = size(TrainFeatures{1});

numHiddenUnits = 512;
numClasses = numel(unique(YTrain));

layers = [ ...
    sequenceInputLayer(inputSize,'Normalization','zscore')
    lstmLayer(numHiddenUnits,'OutputMode','last')
    fullyConnectedLayer(numClasses)
    softmaxLayer
    ];
```

```
maxEpochs = 50;
miniBatchSize = 128;

options = trainingOptions('adam', ...
    'InitialLearnRate',1e-4, ...
    'MaxEpochs',maxEpochs, ...
    'MiniBatchSize',miniBatchSize, ...
    'SequenceLength','shortest', ...
    'Shuffle','every-epoch', ...
    'Plots','training-progress', ...
    'Metrics','accuracy', ...
    'Verbose',true, ...
    'InputDataFormats','CTB');

net = trainnet(TrainFeatures,YTrain,layers,"crossentropy",options);
```

Iteration	Epoch	TimeElapsed	LearnRate	TrainingLoss	TrainingAccuracy
1	1	00:00:07	0.0001	3.7017	2.3438

50	1	00:00:39	0.0001	1.6515	67.188
100	2	00:00:46	0.0001	0.70934	87.5
150	3	00:00:53	0.0001	0.26774	96.875
200	4	00:01:00	0.0001	0.16273	98.438
250	5	00:01:07	0.0001	0.10329	100
300	6	00:01:13	0.0001	0.079866	97.656
350	7	00:01:20	0.0001	0.048372	100

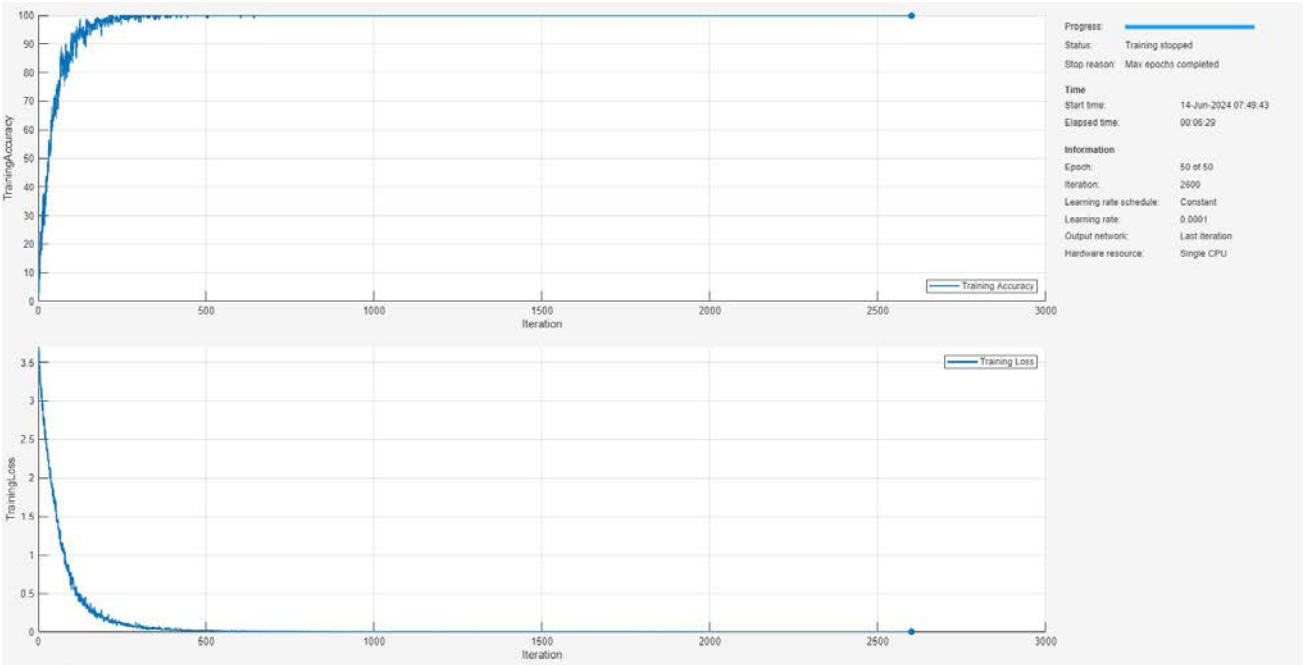


Figure 3-17 Evolution of Training accuracy and training loss

In training, the network has achieved near perfect performance. In order to ensure we have not overfit to the training data, use the held-out test set to determine how well our network generalizes to unseen data

```
scores = minibatchpredict(net,TestFeatures,'InputDataFormats','CTB');
classNames = categories(YTrain);
YPred = scores2label(scores,classNames);
accuracy = 100*sum(YPred == YTest) / numel(YTest)
```

accuracy = 98.6186

In this case, we see that the performance on the held-out test set is also excellent

```
figure
confusionchart(YTest, YPred)
```

```
elapsedTime = toc;  % Stop the timer and get elapsed time in seconds
disp(['Execution time: ', num2str(elapsedTime), ' seconds']);
confusionMatrix = confusionmat(YTest, YPred);
```

```
precision = @(cm) diag(cm) ./ sum(cm, 2);
```

```
recall = @(cm) diag(cm) ./ sum(cm, 1)';
f1Score = @(cm) 2 * (precision(cm) .* recall(cm)) ./ (precision(cm) +
recall(cm));
accuracy = @(cm) sum(diag(cm)) / sum(cm(:));

classPrecision = precision(confusionMatrix);
classRecall = recall(confusionMatrix);
classF1 = f1Score(confusionMatrix);
overallAccuracy = accuracy(confusionMatrix);
```

```
% Assuming you have class labels stored in 'classNames'
for i = 1:length(uniqueLabels)
    fprintf('Class %s:\n', uniqueLabels(i));
    fprintf('  Precision: %.4f\n', classPrecision(i));
    fprintf('  Recall: %.4f\n', classRecall(i));
    fprintf('  F1-Score: %.4f\n', classF1(i));
end
fprintf('Overall Accuracy: %.4f\n', overallAccuracy);
```

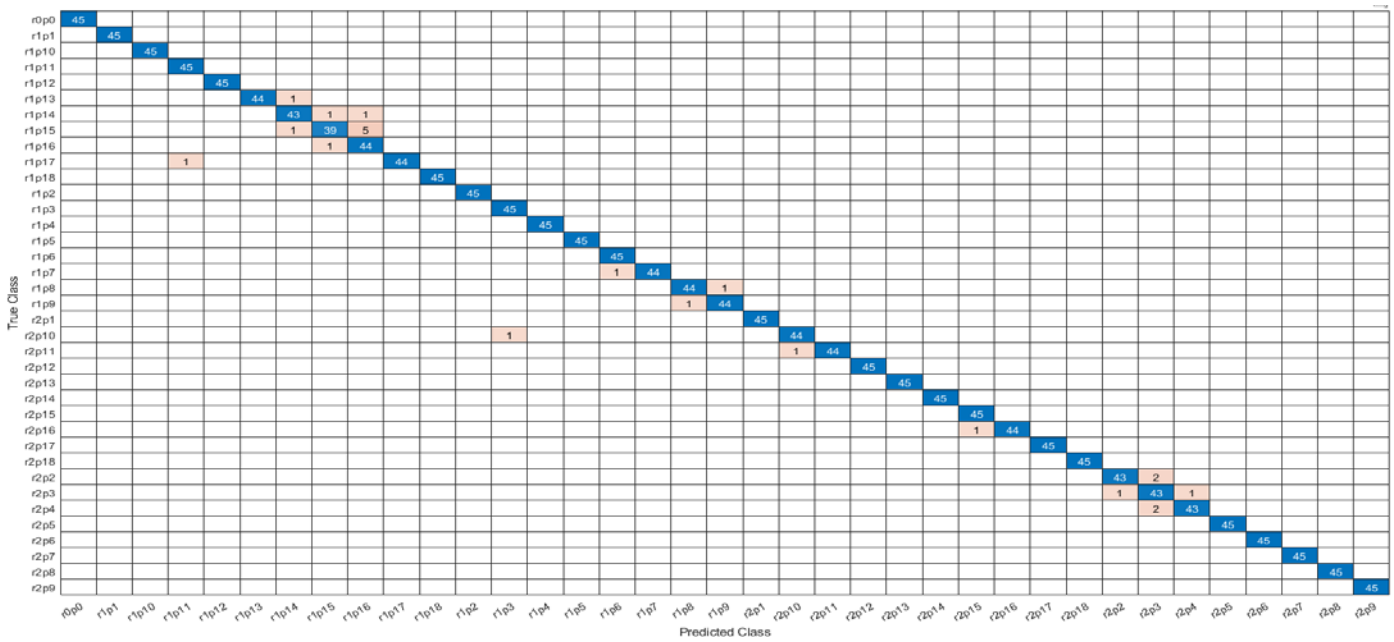


Figure 3-18 Confusion matrix for sensor 1 classifications results with RCNN with MATLAB

Execution time: 590.7942 seconds

Class r0p0: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p1: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p10: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p11: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p12: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p13: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p14: Precision: 0.9556 Recall: 0.9556 F1-Score: 0.9556
Class r1p15: Precision: 0.8667 Recall: 0.9512 F1-Score: 0.9070
Class r1p16: Precision: 0.9778 Recall: 0.8800 F1-Score: 0.9263

Class r1p17: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p18: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p2: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p3: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p4: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p5: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p6: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p7: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p8: Precision: 0.9778 Recall: 0.9778 F1-Score: 0.9778
Class r1p9: Precision: 0.9778 Recall: 0.9778 F1-Score: 0.9778
Class r2p1: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p10: Precision: 0.9778 Recall: 0.9778 F1-Score: 0.9778
Class r2p11: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r2p12: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p13: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p14: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p15: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r2p16: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r2p17: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p18: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p2: Precision: 0.9556 Recall: 0.9773 F1-Score: 0.9663
Class r2p3: Precision: 0.9556 Recall: 0.9149 F1-Score: 0.9348
Class r2p4: Precision: 0.9556 Recall: 0.9773 F1-Score: 0.9663
Class r2p5: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p6: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p7: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p8: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p9: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Overall Accuracy: 0.9862

Class r0p0 correspond to no unbalance added

Class r1p1 to class r1p18 correspond to unbalance in Disk1 at positions 1 to 18 respectively

Class r2p1 to class r2p18 correspond to unbalance in Disk2 at positions 1 to 18 respectively

In this case we inspired our model from example of air compressors using a wavelet scattering network paired with a recurrent neural network

[https://www.mathworks.com/help/wavelet/ug/fault-detection-using-wavelet-scattering-and-recurrent-deep-networks.html].

3.8.2 MATLAB Neural Network (phone sound)

Disk 2 only

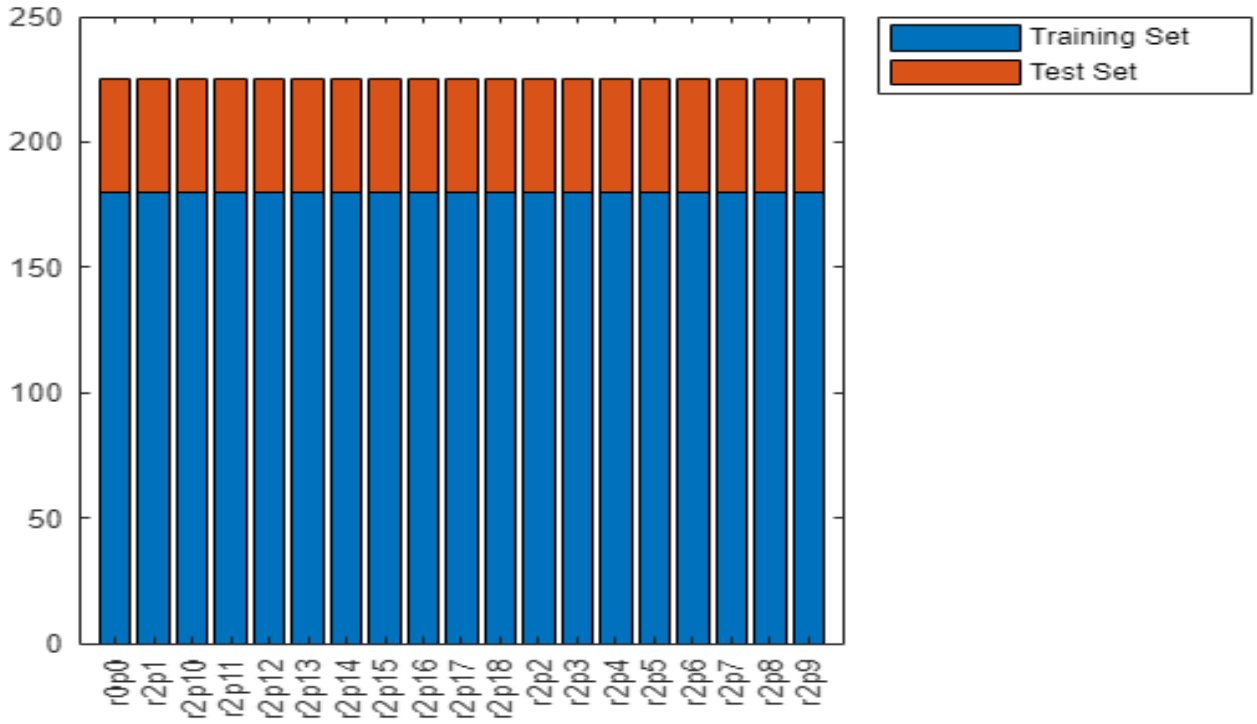


Figure 3-19 Signal for phone sound data Distribution of training and test for unbalance in Disk 2

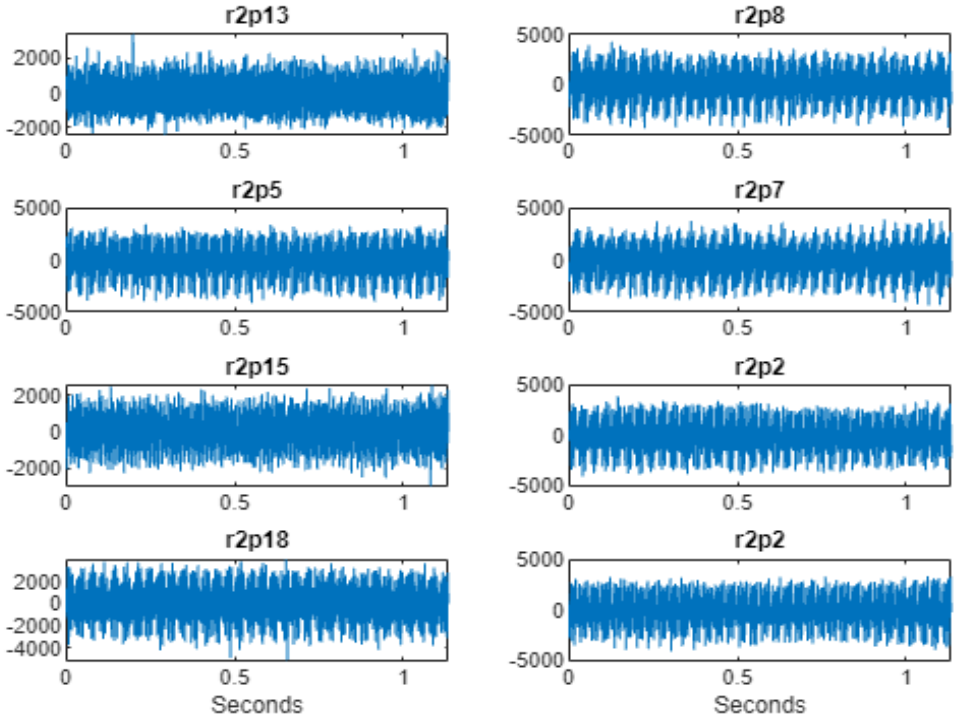


Figure 3-20 Signal for phone sound data at different unbalance positions on disk 2

ans = 411

Iteration	Epoch	Time Elapsed	Learn Rate	Training Loss	Training Accuracy
1	1	00:00:00	0.0001	2.9779	3.9062
50	2	00:00:25	0.0001	0.42651	97.656
100	4	00:00:32	0.0001	0.10382	98.438
150	6	00:00:39	0.0001	0.038513	100
200	8	00:00:46	0.0001	0.018228	100
		……			
		…….			
1250	49	00:03:12	0.0001	0.00084274	100
1300	50	00:03:19	0.0001	0.00074617	100

Training stopped: Max epochs completed

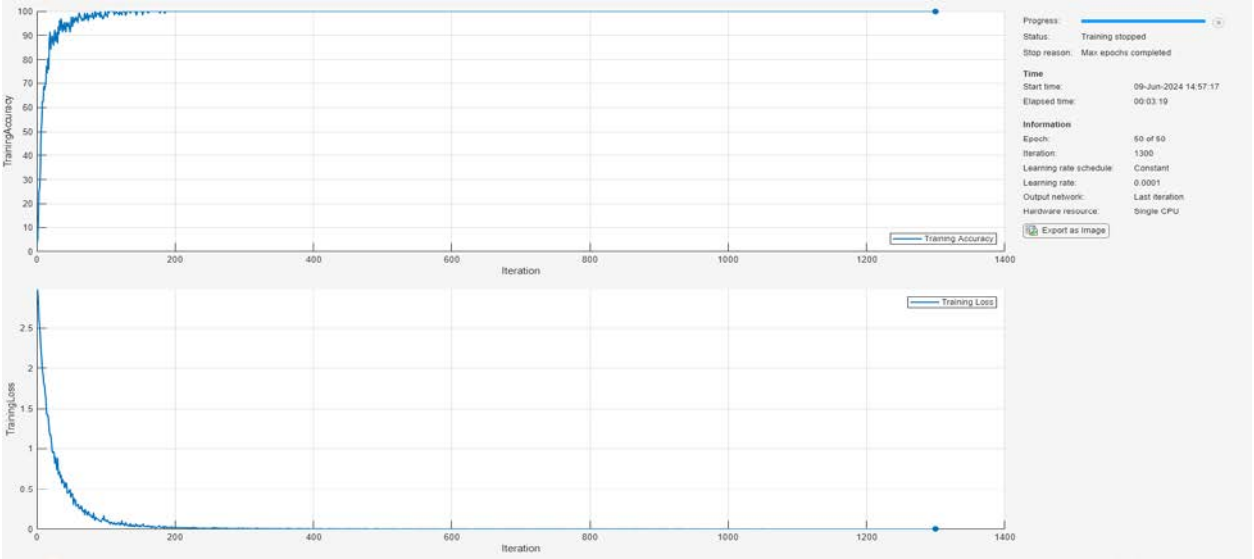


Figure 3-21 training accuracy and training loss

accuracy = 99.4152

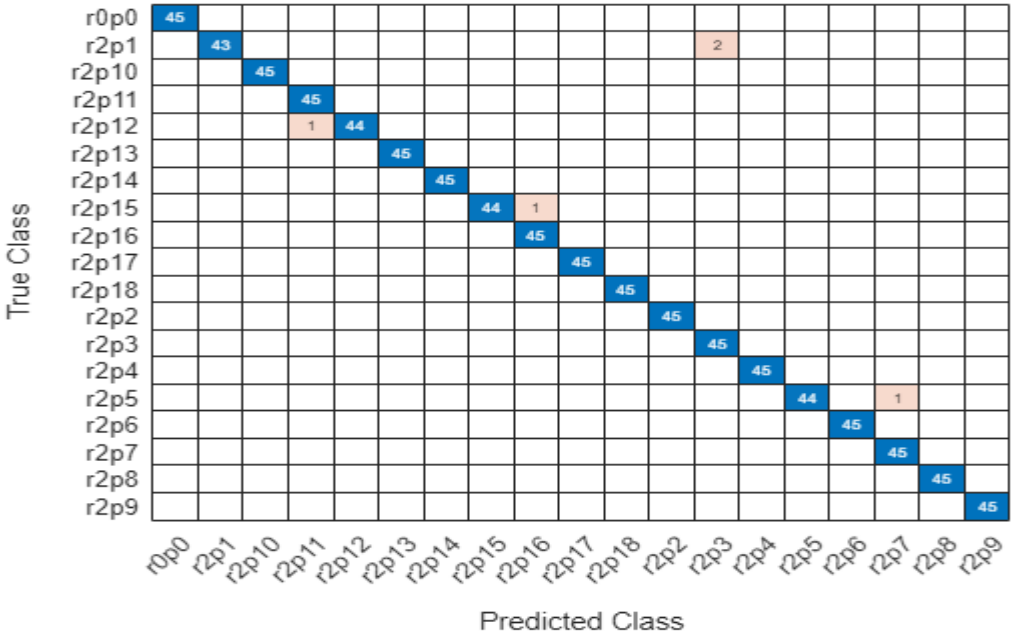


Figure 3-22 Confusion Matrix of unbalance in different classes

disk1 + disk 2

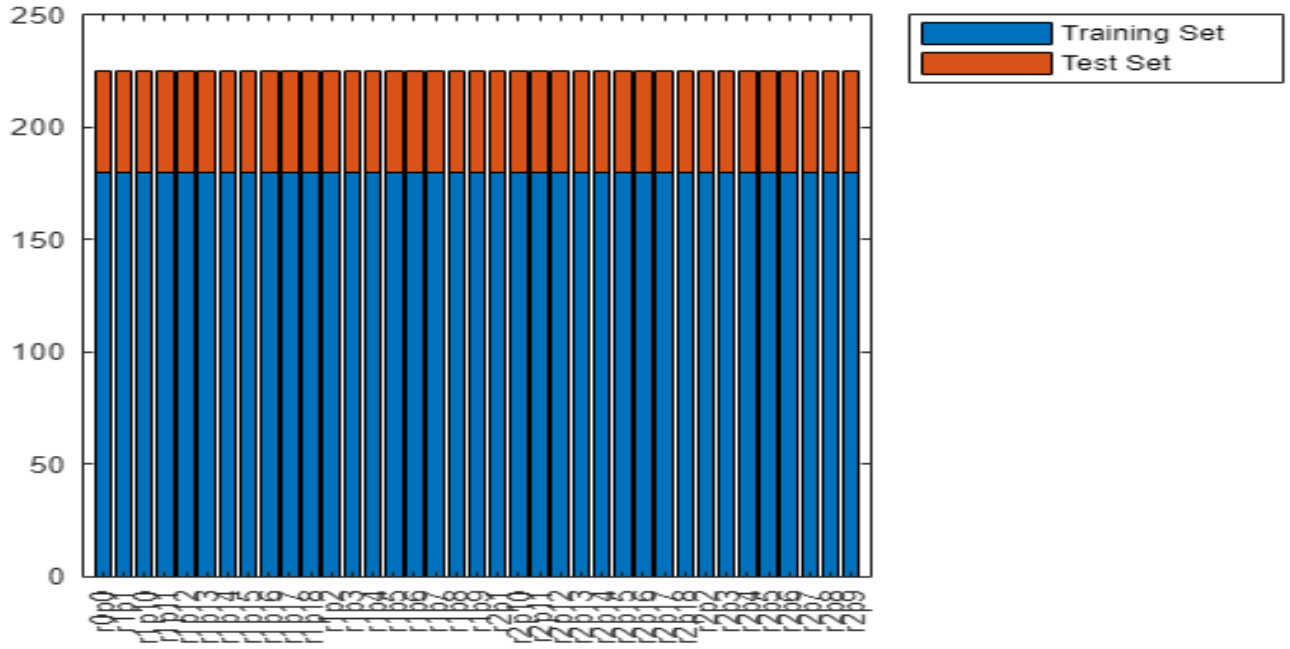


Figure 3-23 Distribution data for training and test

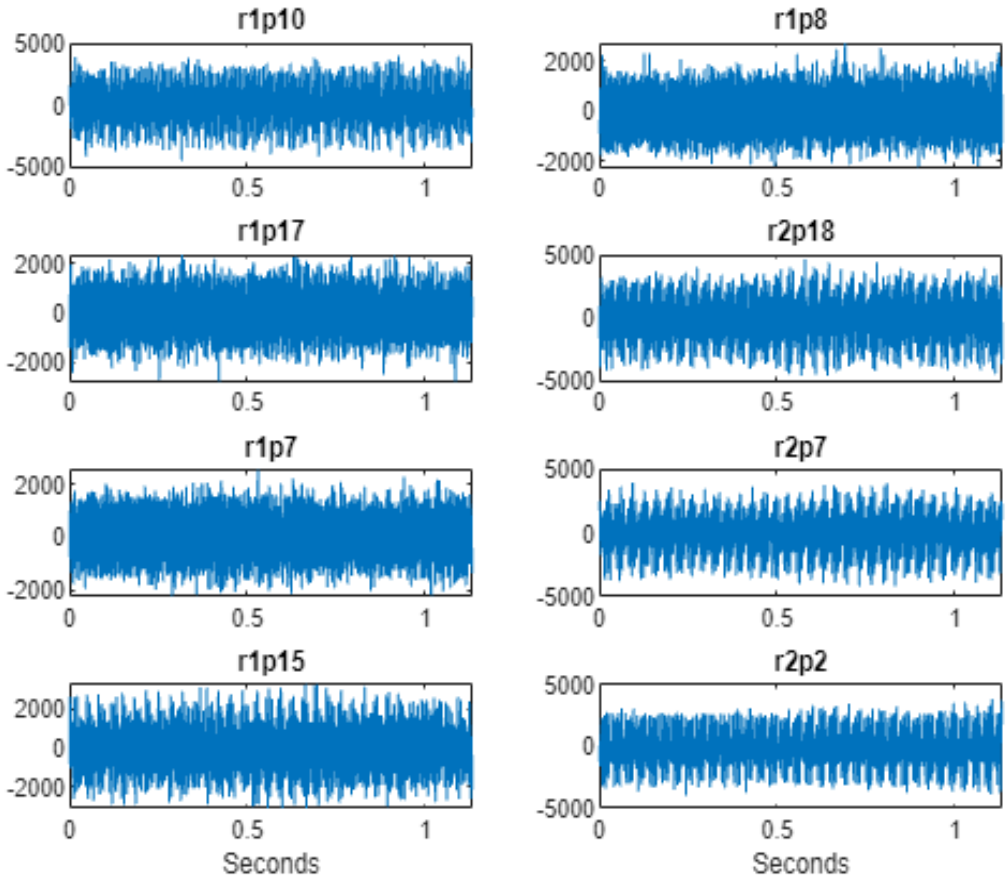


Figure 3-24 Signal for phone sound data at different unbalance positions on disk 1 and disk 2

Iteration	Epoch	TimeElapsed	LearnRate	TrainingLoss	TrainingAccuracy
1	1	00:00:03	0.0001	3.6471	6.25
50	1	00:00:33	0.0001	1.0744	87.5
100	2	00:00:40	0.0001	0.46018	95.312
150	3	00:00:47	0.0001	0.15261	98.438
200	4	00:00:54	0.0001	0.088364	99.219
250	5	00:01:01	0.0001	0.044686	100
300	6	00:01:08	0.0001	0.034907	100
350	7	00:01:15	0.0001	0.028073	100
2500	49	00:06:14	0.0001	0.00072781	100
2550	50	00:06:21	0.0001	0.00076323	100
2600	50	00:06:28	0.0001	0.00068288	100

Training stopped: Max epochs completed

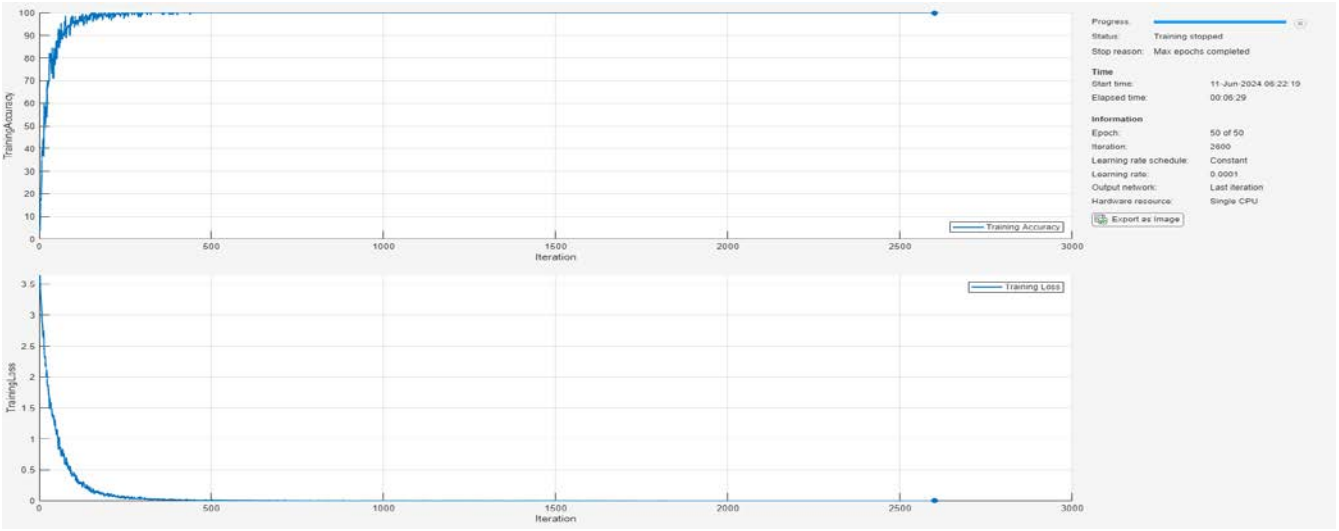


Figure 3-25 Training accuracy and training loss

accuracy = 99.3393

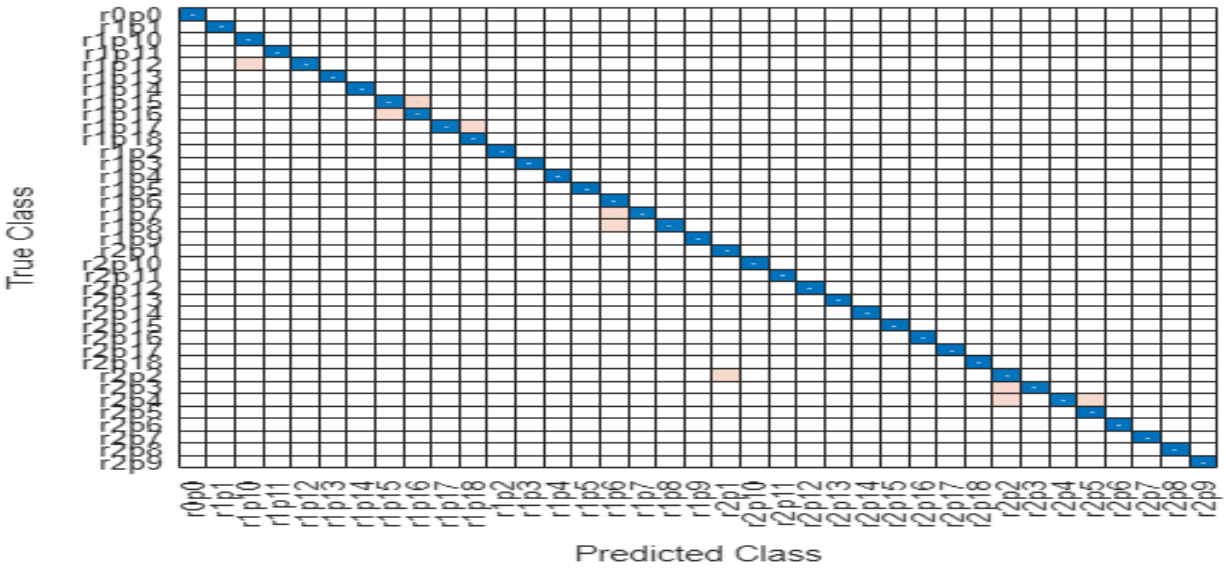


Figure 3-26 Confusion Matrix for classification of unbalance

Class r0p0: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p1: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p10: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p11: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p12: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p13: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p14: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p15: Precision: 0.9556 Recall: 0.9773 F1-Score: 0.9663
Class r1p16: Precision: 0.9778 Recall: 0.9565 F1-Score: 0.9670
Class r1p17: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p18: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p2: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p3: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p4: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p5: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p6: Precision: 1.0000 Recall: 0.9574 F1-Score: 0.9783
Class r1p7: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p8: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p9: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p1: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r2p10: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p11: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p12: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p13: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p14: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p15: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p16: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p17: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p18: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p2: Precision: 0.9778 Recall: 0.9565 F1-Score: 0.9670
Class r2p3: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r2p4: Precision: 0.9556 Recall: 1.0000 F1-Score: 0.9773
Class r2p5: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r2p6: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p7: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p8: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p9: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000

Overall Accuracy: 0.9934
Class r0p0 correspond to no unbalance added
Class r1p1 to class r1p18 correspond to unbalance in Disk1 at positions 1 to 18 respectively
Class r2p1 to class r2p18 correspond to unbalance in Disk2 at positions 1 to 18 respectively

3.8.3 MATLAB SVM (phone sound)

Sound wave from phone SVM classification of 37 classes

```
datasetLocation = fullfile('H:\vibration data\09062024\','split','');
ads = audioDatastore(datasetLocation,'IncludeSubfolders',true,...
    'LabelSource','foldernames');
```

```
countcats(ads.Labels)
```

ans = 37×1

```
  225
  225
  225
  225
  225
  225
  225
  225
  225
  225
```

```
rng default
ads = shuffle(ads);
[adsTrain,adsTest] = splitEachLabel(ads,0.8,0.2);

uniqueLabels = unique(adsTrain.Labels);
tblTrain = countEachLabel(adsTrain);
tblTest = countEachLabel(adsTest);
H = bar(uniqueLabels,[tblTrain.Count, tblTest.Count],'stacked');
legend(H,["Training Set","Test Set"],'Location','NorthEastOutside')
```

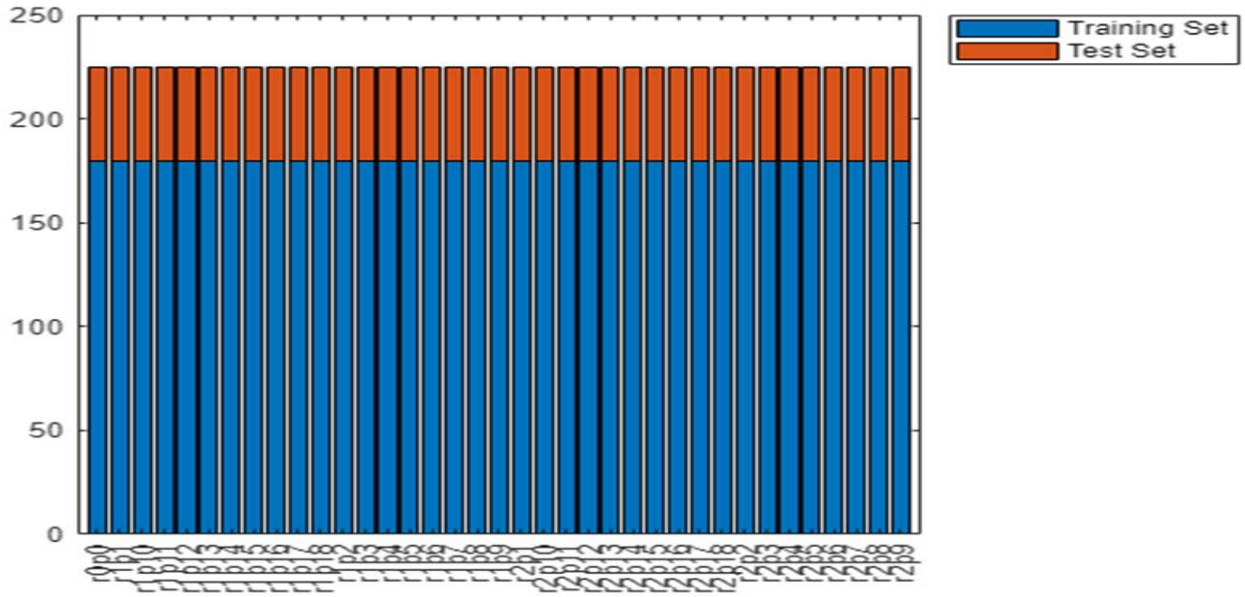


Figure 3-27 Distribution data for training and test

Select some random examples from the training set for plotting

```
idx = randperm(numel(adsTrain.Files),8);
Fs = 44100;
for n = 1:numel(idx)
    x = audioread(adsTrain.Files{idx(n)});
    t = (0:size(x,1)-1)/Fs;
    subplot(4,2,n);
    plot(t,x);
    if n == 7 || n == 8
        xlabel('Seconds');
    end
    title(string(adsTrain.Labels(idx(n))));
end
```

Wavelet Scattering Network

Construct a wavelet scattering network based on the data characteristics.

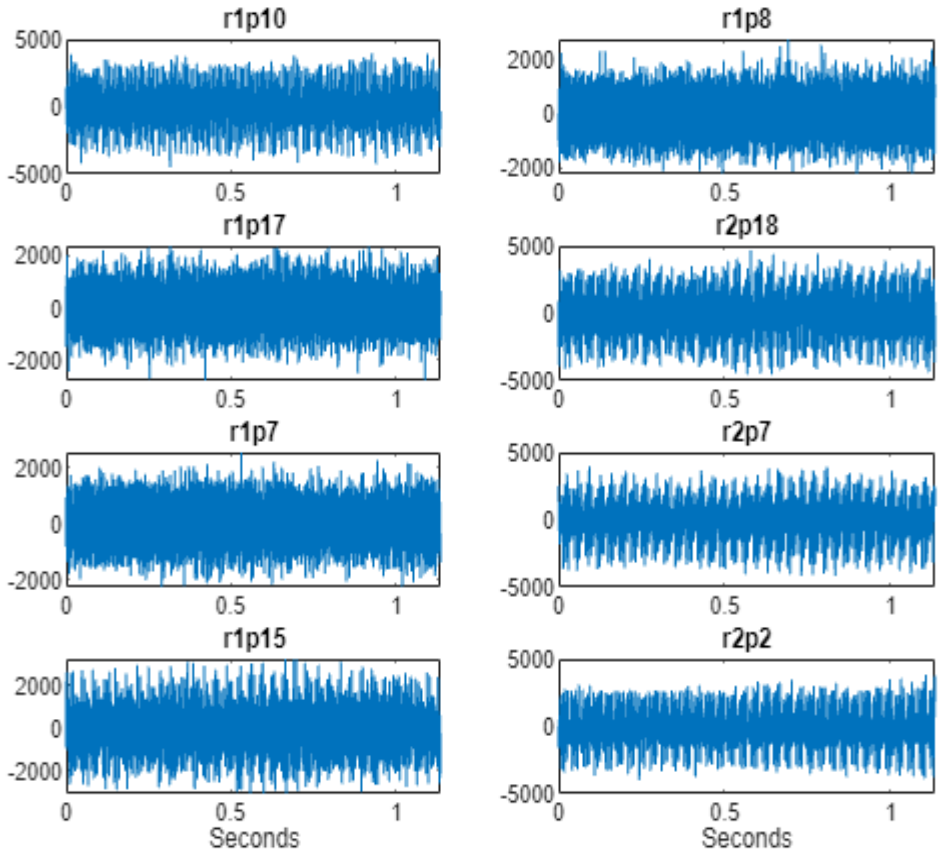


Figure 3-28 Signal for phone Sound data of different points of unbalance on disk1 and Disk2

Wavelet Scattering Network

Construct a wavelet scattering network based on the data characteristics.

```
N = 5e4;
Fs = 44100;
IS = 0.5;
sn = waveletScattering('SignalLength',N,'SamplingFrequency',Fs,...
    'InvarianceScale',0.5);

[~,numpaths] = paths(sn);
```

```
Ncfs = numCoefficients(sn);
sum(numpaths)
```

```
ans = 411
```

```
Ncfs
```

```
Ncfs = 13
```

```
batchsize = 64;
useGPU = false;
scTrain = [];
while hasdata(adsTrain)
    sc = helperBatchScatFeatures2(adsTrain,sn,N,batchsize,useGPU);
    scTrain = cat(3,scTrain,sc);
end
```

```
scTest = [];
while hasdata(adsTest)
   sc = helperBatchScatFeatures2(adsTest,sn,N,batchsize,useGPU);
   scTest = cat(3,scTest,sc);
end
scTest = gather(scTest);
```

```
[Npaths,numTimeWindows,~] = size(scTrain);
```

```
TrainFeatures = permute(scTrain,[2 3 1]);
TrainFeatures = reshape(TrainFeatures,[],Npaths,1);
TestFeatures = permute(scTest,[2 3 1]);
TestFeatures = reshape(TestFeatures,[],Npaths,1);
```

```
trainLabels = adsTrain.Labels;
numTrainSignals = numel(trainLabels);
trainLabels = repmat(trainLabels,1,numTimeWindows);
trainLabels = reshape(trainLabels',numTrainSignals*numTimeWindows,1);
```

SVM Classification

Use a multi-class support vector machine (SVM) classifier with a cubic polynomial kernel. Fit the SVM to the training data

```
template = templateSVM(...
    'KernelFunction', 'polynomial', ...
    'PolynomialOrder', 3, ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...
    'Standardize', true);
```

```
classificationSVM = fitcecoc(...
    TrainFeatures, ...
    trainLabels, ...
    'Learners', template, ...
    'Coding', 'onevsall','ClassNames',uniqueLabels);
```

```
partitionedModel = crossval(classificationSVM, 'KFold', 5);
validationAccuracy = (1 - kfoldLoss(partitionedModel))*100
```

```
validation Accuracy = single 99.3193
```

```
validationPredictions = kfoldPredict(partitionedModel);
TrainVotes =
helperMajorityVote(validationPredictions,adsTrain.Labels,uniqueLabels);
crossvalAccuracy = sum(TrainVotes ==
adsTrain.Labels)/numel(adsTrain.Labels)*100;
crossvalAccuracy
```

```
crossvalAccuracy = 99.7147
```

```
predLabels = predict(classificationSVM,TestFeatures);
[TestVotes,TestCounts] =
helperMajorityVote(predLabels,adsTest.Labels,uniqueLabels);
testAccuracy = sum(TestVotes == adsTest.Labels)/numel(adsTest.Labels)*100;
testAccuracy
```

```
testAccuracy = 99.3994
```

```
figure
confusionchart(adsTest.Labels,TestVotes)
```

```
confusionMatrix = confusionmat(adsTest.Labels, TestVotes);
```

```
precision = @(cm) diag(cm) ./ sum(cm, 2);
recall = @(cm) diag(cm) ./ sum(cm, 1)';
f1Score = @(cm) 2 * (precision(cm) .* recall(cm)) ./ (precision(cm) +
recall(cm));
accuracy = @(cm) sum(diag(cm)) / sum(cm(:));


classPrecision = precision(confusionMatrix);
classRecall = recall(confusionMatrix);
classF1 = f1Score(confusionMatrix);
overallAccuracy = accuracy(confusionMatrix);
```

```
% Assuming you have class labels stored in 'classNames'
for i = 1:length(uniqueLabels)
    fprintf('Class %s:\n', uniqueLabels(i));
    fprintf('  Precision: %.4f\n', classPrecision(i));
    fprintf('  Recall: %.4f\n', classRecall(i));
    fprintf('  F1-Score: %.4f\n', classF1(i));
end
fprintf('Overall Accuracy: %.4f\n', overallAccuracy);
```

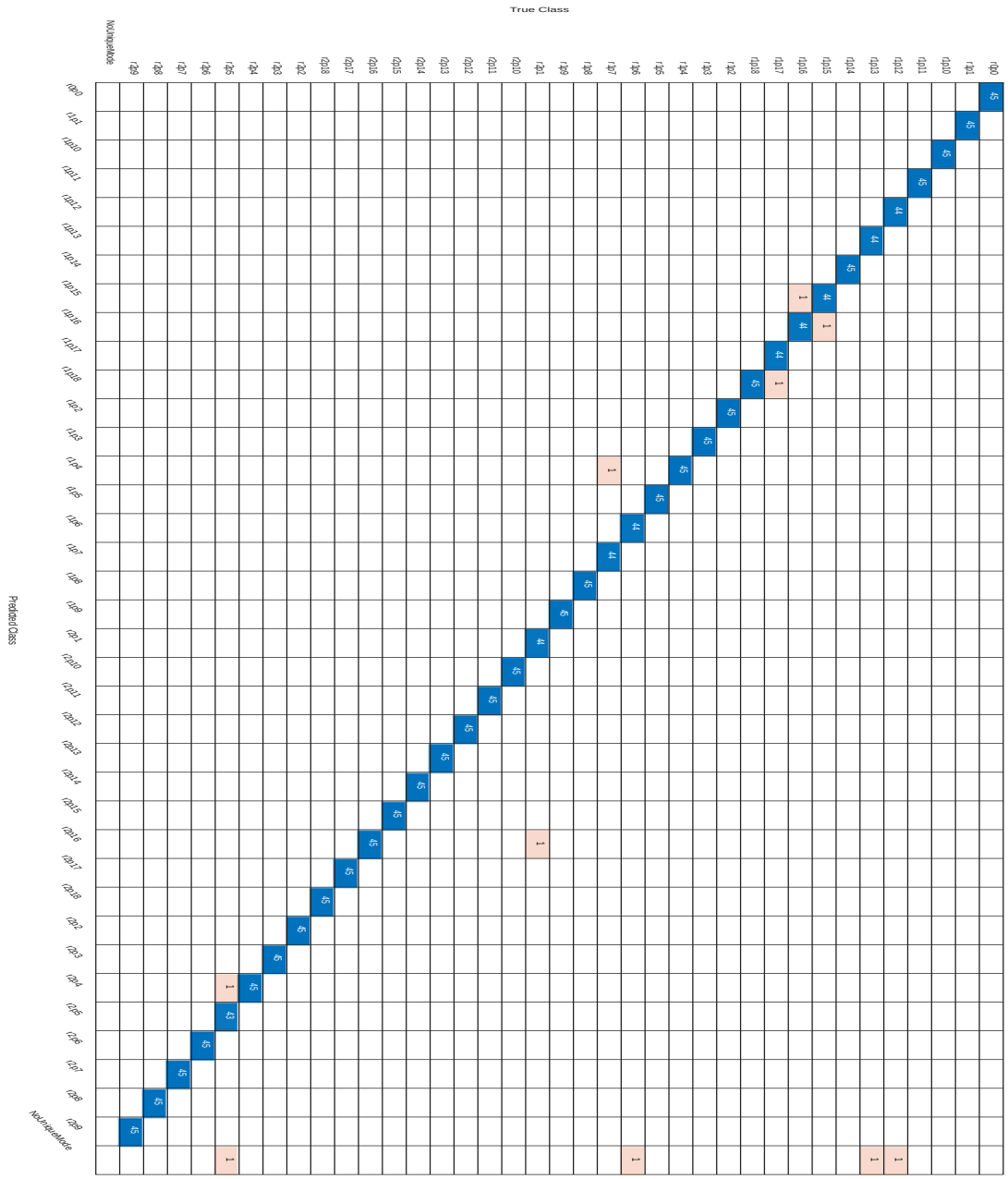


Figure 3-29 classification of unbalance in different classes

We inspired our example from example shows us how to classify faults in acoustic recordings of air compressors using a wavelet scattering network and a support vector machine (SVM) [https://www.mathworks.com/help/wavelet/ug/air-compressor-fault-detection-using-wavelet-scattering.html]

Class r0p0: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p1: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p10: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p11: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p12: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p13: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p14: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p15: Precision: 0.9778 Recall: 0.9778 F1-Score: 0.9778
Class r1p16: Precision: 0.9778 Recall: 0.9778 F1-Score: 0.9778
Class r1p17: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p18: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p2: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p3: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p4: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p5: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p6: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p7: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r1p8: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p9: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p1: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r2p10: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p11: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p12: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p13: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p14: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p15: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p16: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r2p17: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p18: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p2: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p3: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p4: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r2p5: Precision: 0.9556 Recall: 1.0000 F1-Score: 0.9773
Class r2p6: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p7: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p8: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p9: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000

Overall Accuracy: 0.9940
Class r0p0 correspond to no unbalance added

Class r1p1 to class r1p18 correspond to unbalance in Disk1 at positions 1 to 18 respectively

Class r2p1 to class r2p18 correspond to unbalance in Disk2 at positions 1 to 18 respectively

3.8.4 MATLAB SVM (sensor1)

SVM sensor 1 7 mn execution time ans = 37×1

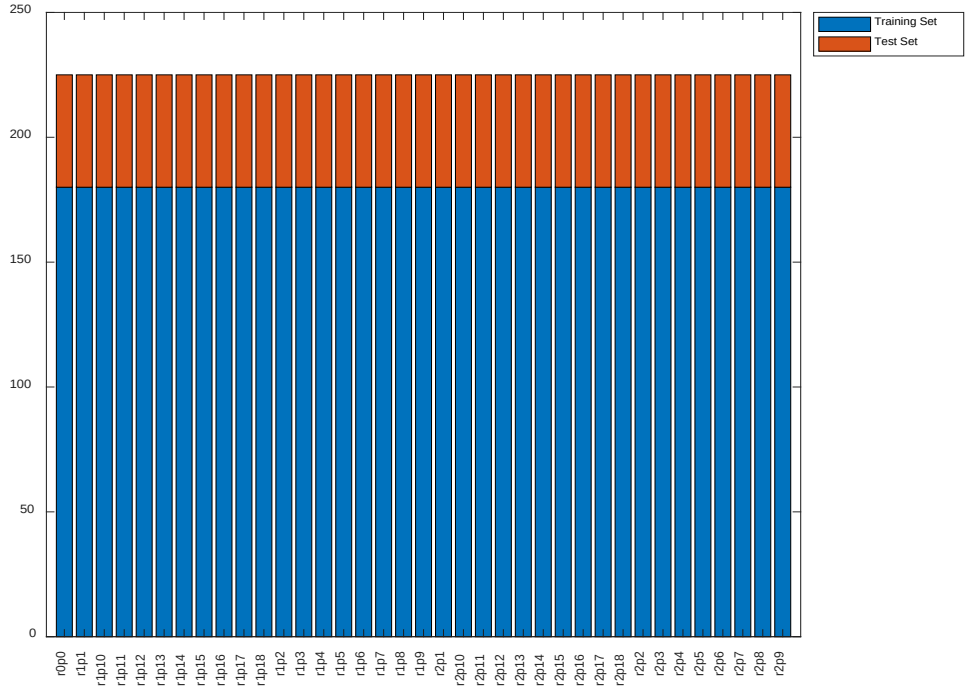


Figure 3-30 Distribution data for training and test

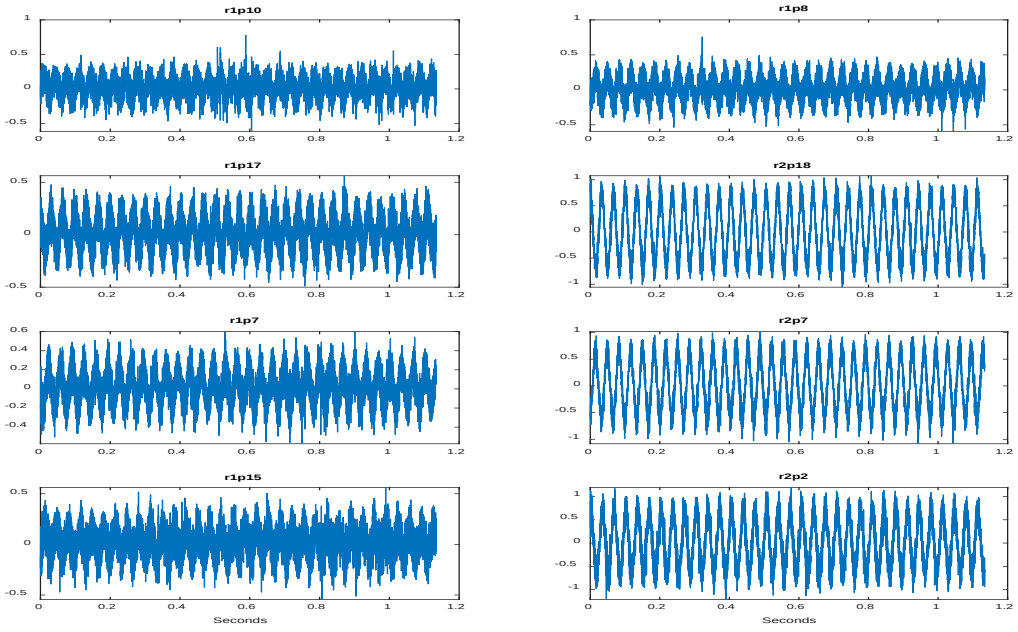


Figure 3-31 Signal data for sensor 1 for different position of unbalance on disk1 and disk2

ans = 411

Ncfs = 13
validation Accuracy = single
95.2104
crossvalAccuracy = 98.3634
test Accuracy = 97.0571

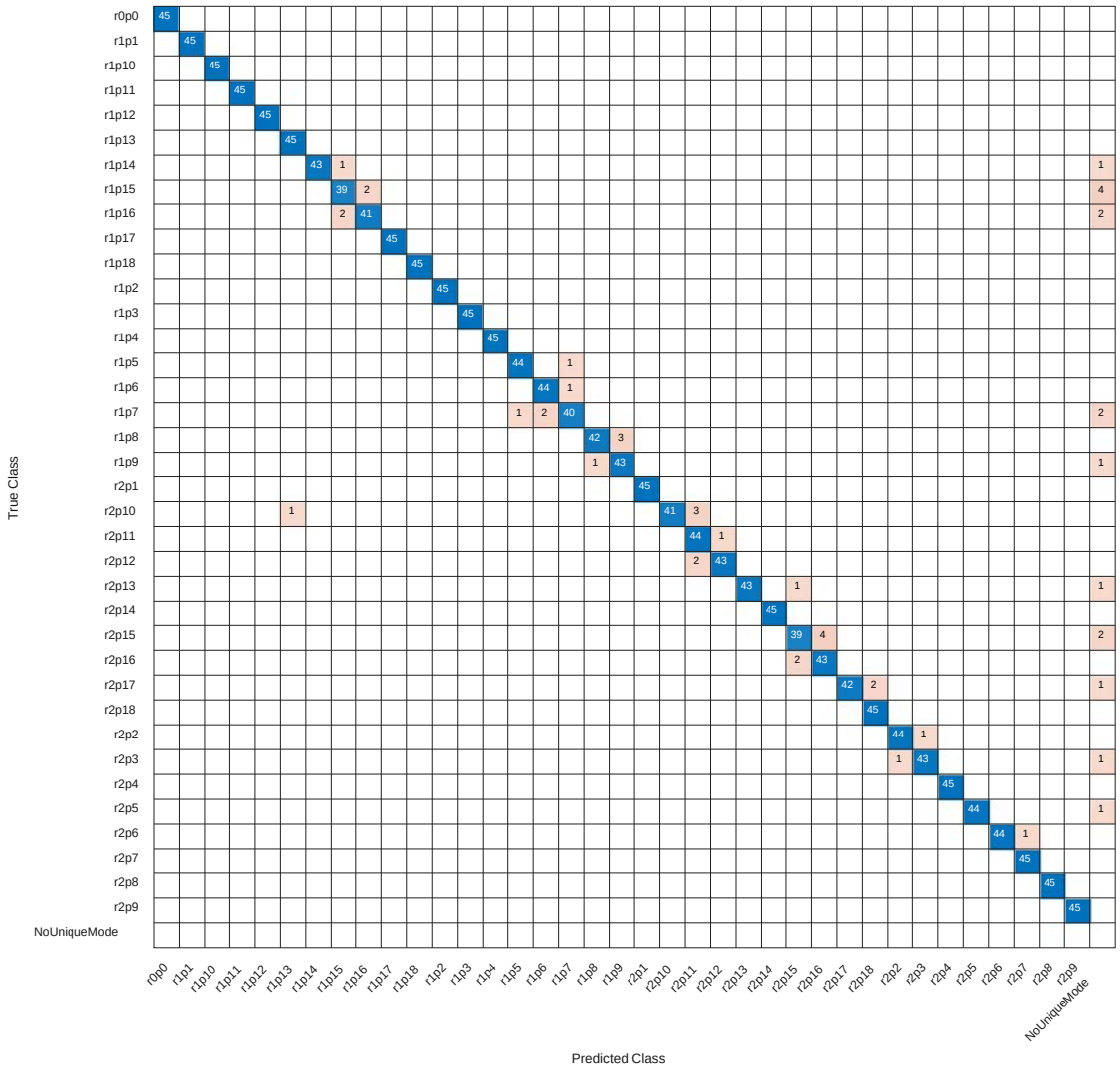


Figure 3-32 Confusion matrix of classification model for unbalance data in positions and different disks

Class r0p0: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p1: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p10: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p11: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p12: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p13: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r1p14: Precision: 0.9556 Recall: 1.0000 F1-Score: 0.9773
Class r1p15: Precision: 0.8667 Recall: 0.9286 F1-Score: 0.8966
Class r1p16: Precision: 0.9111 Recall: 0.9535 F1-Score: 0.9318
Class r1p17: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p18: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000

Class r1p2: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p3: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p4: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r1p5: Precision: 0.9778 Recall: 0.9778 F1-Score: 0.9778
Class r1p6: Precision: 0.9778 Recall: 0.9565 F1-Score: 0.9670
Class r1p7: Precision: 0.8889 Recall: 0.9524 F1-Score: 0.9195
Class r1p8: Precision: 0.9333 Recall: 0.9767 F1-Score: 0.9545
Class r1p9: Precision: 0.9556 Recall: 0.9348 F1-Score: 0.9451
Class r2p1: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p10: Precision: 0.9111 Recall: 1.0000 F1-Score: 0.9535
Class r2p11: Precision: 0.9778 Recall: 0.8980 F1-Score: 0.9362
Class r2p12: Precision: 0.9556 Recall: 0.9773 F1-Score: 0.9663
Class r2p13: Precision: 0.9556 Recall: 1.0000 F1-Score: 0.9773
Class r2p14: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p15: Precision: 0.8667 Recall: 0.9286 F1-Score: 0.8966
Class r2p16: Precision: 0.9556 Recall: 0.9149 F1-Score: 0.9348
Class r2p17: Precision: 0.9333 Recall: 1.0000 F1-Score: 0.9655
Class r2p18: Precision: 1.0000 Recall: 0.9574 F1-Score: 0.9783
Class r2p2: Precision: 0.9778 Recall: 0.9778 F1-Score: 0.9778
Class r2p3: Precision: 0.9556 Recall: 0.9773 F1-Score: 0.9663
Class r2p4: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p5: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r2p6: Precision: 0.9778 Recall: 1.0000 F1-Score: 0.9888
Class r2p7: Precision: 1.0000 Recall: 0.9783 F1-Score: 0.9890
Class r2p8: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000
Class r2p9: Precision: 1.0000 Recall: 1.0000 F1-Score: 1.0000

Overall Accuracy: 0.9706

Class r0p0 correspond to no unbalance added

Class r1p1 to class r1p18 correspond to unbalance in Disk1 at positions 1 to 18 respectively

Class r2p1 to class r2p18 correspond to unbalance in Disk2 at positions 1 to 18 respectively

3.9 Conclusion

In this chapter, we have introduced the dataset that is used to develop, test, and evaluate a predictive maintenance approach. Next, we described the preprocessing operations performed on the dataset in order to prepare it for the training phase. Finally, we briefly discussed a machine learning approach based on regression and classification, as well as the different metrics that can be used to evaluate the learning model. We also implemented a few learning algorithms and evaluated their performance. We found good results for classification model both when using sensor data and sound captured by usual cell phone, and the two approach of using MATLAB and python open-source library gave verry good performance models

General conclusion

The project presented in this thesis proposes a solution to exploit the potential of artificial intelligence models to improve the industrial maintenance process by predicting machine defects in advance. We have highlighted the importance of having an effective predictive maintenance program. Indeed, unplanned production line shutdowns and production delays lead to financial and time losses that directly affect the budgetary health and competitiveness of companies. However, to get good results or reliable predictions, it is crucial to have an extensive database with a lot of raw data collected on equipment. Indeed, the more training data a model has, the more correctly it is able to predict observations. In addition, this data must be consistent and of high quality. It should be noted that prediction from simulated datasets is largely achievable nowadays thanks to the development and open-source sharing of data platforms. In addition, the development of new Python libraries such as Scikit-learn has made it much easier to apply artificial intelligence in the field of maintenance, especially for people who are not specialists in this field.

In conclusion, the use of machine learning provides maintenance managers with valuable information to understand, improve, and prepare the necessary hardware and human resources before a failure occurs. This helps replace traditional maintenance strategies, such as corrective and preventive maintenance, with a predictive model-based approach to predictive maintenance. The evaluation of the performance of the models obtained shows that the application of artificial intelligence in the industrial field is extremely beneficial for companies. It increases the profitability of their production equipment and, as a result, increases the company's profits. By using predictive models, companies can anticipate breakdowns, plan maintenance operations efficiently, and avoid unplanned production shutdowns. This makes it possible to optimize the use of resources, minimize financial losses and to strengthen competitiveness in the market. Artificial intelligence therefore offers a powerful new approach to improve industrial maintenance management and maximize operational performance.

During the design and implementation of our project, we encountered several challenges that we managed to overcome, which was extremely beneficial. First, we had to gain new knowledge in areas such as data science and machine learning. Then, during the implementation of our solution, we discovered new programming tools such as Python, Jupyter Notebook, PyCharm, as well as different libraries to process data, implement machine learning algorithms and visualize results (NumPy, Pandas, Matplotlib, Scikit-learn, etc.). This experience also allowed us to improve our skills in research methodology, communication and writing. On a personal level, this project opens up many prospects for improvement. We worked on simulated

data rather than real data from machines in real use. It is therefore essential to carry out a data acquisition and collection phase in an industrial environment before implementing the system.

Given the limitations of machine learning techniques in some situations, we propose to use more advanced learning techniques, such as deep learning, to model maintenance tasks more efficiently. Finally, we plan to continue our work on the application of machine learning to industrial maintenance as part of a PhD, to deepen our knowledge in this fascinating field and contribute to scientific advances through research

References

[1] NF EN 13306 la norme AFNOR (Norme européenne, Éditée et diffusée par Association Française de Normalisation) NF-X 60 000 Juin 2001.

[2] https://www.researchgate.net/figure/Les-differents-types-de-maintenance_fig1_281793037

[3] Selon la définition AFNOR (norme x60-010).

[4] Norme x60-319/nf en 13306 terminologies de la maintenance, 2001.

[5]Cours maintenance industrielle université mostefa ben boulaid -batna2

[6]X60-319/NF EN 13306 2010 AFNOR Terminologie de la maintenance.

[7]Z. Cai, S. Sun, S. Si, N. Wang, "Research of Failure Prediction Bayesian Network Model", - IEEE, IE&EM '09. 16th International Conference on Industrial Engineering and Engineering Management, 2009.

[8] L.S. Dhamandea and M.B. Chaudharib, " Detection of Combined Gear Bearing Fault in Single Stage Spur
Gear Box Using Artificial Neural Network", Procedia Engineering, Vol. 144, pp. 759-766, 2016. DOI: 10.1016/j.proeng.2016.05.082

[9] Ilyés KHELF, « DIAGNOSTIC DES MACHINES TOURNANTES PAR LES TECHNIQUES DE L'INTELLIGENCE
ARTIFICIELLE », thèse de doctorat- UNIVERSITE BADJ MOKHTAR – ANNABA-2014.

[10] R.B. Randall. "Vibration-based Condition Monitoring INDUSTRIAL, AEROSPACE AND AUTOMOTIVE

[11] - Bruel et Kjaer,Schenck -vibrations, Equilibrage sur site, application a la maintenance conditionnelle- document technique -1999-

[12] - H Campagna,B Agsous -La mise en œuvre du diagnostic vibratoire sur machine tournante–dBVib – 2003.

[12] H. Caoa, T. Dörgelohb, O. Riemerb and E. Brinksmeierb," Adaptive separation of unbalance vibration in air bearing spindles", Procedia CIRP, Volume 62, pp. 357-362, 2017. DOI: 10.1016/j.procir.2016.06.069.

[14] A. Simm, Q. Wanga, S. Huan and W. Zhao, "Laser based measurement for the monitoring of shaft
misalignment", Measurement, Vol. 87, pp. 104–116, 2016. DOI: 10.1016/j.measurement.2016.02.034.

[15] T.H. Patel and A.K. Darpe, "Experimental investigations on vibration response of misaligned rotors",
Mechanical Systems and Signal Processing, Vol. 23 (7), pp. 2236-2252, 2009. DOI: 10.1016/j.ymssp.2009.04.004.

[16] Y. Fang, H. Cho and M. Jeong, "Health monitoring of a shaft transmission system via hybrid models of

PCR and PLS", Sixth SIAM International Conference on Data Mining, p. 554-558, 2006. DOI: 10.1137/1.9781611972764.59.

[17] A. Kr Jalan and A.R. Mohanty, "Model based fault diagnosis of a rotor bearing system for misalignment and unbalance under steady-state condition", Journal of Sound and Vibration, Vol. 327 (3–5), pp. 604- 622, 2009. DOI: 10.1016/j.jsv.2009.07.014.

[18] JL.F. Chacon, V. Kappatos, W. Balachandran and TH. Gan, "A novel approach for incipient defect detection in rolling bearings using acoustic emission technique", Applied Acoustics, Vol. 89, pp. 88-100, 2015.

[19] https://www.bksv.com/es/knowledge/blog/vibration/static-and-dynamic-balancing-of-rigid-rotors

[20] Innocent Mateyaunga, « Prédictive Maintenance Using Machine Learning », la thèse de master, sous la direction de Hadj Abdelkader, Faculté de technologie de l'université de Tlemcen

[21] https://www.oracle.com/dz/artificial-intelligence/what-is-ai/ 11/03/2024

[22] https://www.sicara.fr/parlons-data/machine-learning 14/03/2024

[23]https://aws.amazon.com/fr/solutions/implementations/predictive-maintenance-using-machine-learning/ 14/03/2024

[24] https://myriad-data.com/wp-content/uploads/2019/07/nlp.pdf7 18/03/2024

[25] https://ia-data-analytics.fr/machine-learning/usages/23/03/2024

[26]https://newsinitiative.withgoogle.com/fr-fr/resources/lessons/different-approaches-to-machine-learning/27/03/2024

[27] https://www.talend.com/fr/resources/what-is-machine-learning/ 1/04/2024

[28] https://www.actuia.com/keras/pourquoi-utiliser-keras/ 19/04/2024

[29] https://www.actuia.com/keras/pourquoi-utiliser-keras/ 19/04/2024

[30] https://brightcape.co/machine-learning-open-source/ 11/04/2024

[31]https://www.quora.com/Which-deep-learning-framework-between-Tensorflow-or-PyTorch-should-I-choose-for-NLP-research

[32]https://www.geeksforgeeks.org/time-series-analysis-and-forecasting

[33] https://www.geeksforgeeks.org/components-of-time-series-data/

[34] https://towardsdatascience.com/arima-simplified-b63315f27cbc

[35] https://www.geeksforgeeks.org/anomaly-detection-in-time-series-data/

[36]https://medium.com/@sanghaviharsh666/mastering-svm-kernel-tricks-a-comprehensive-guide-to-dual-problems-and-kernel-functions-612bfff2061e

[37] https://www.mateconferences.org/articles/matecconf/pdf/2023/04/

[38] https://www.geeksforgeeks.org/support-vector-machine-algorithm/?ref=header_search

[39] J.-P. Bruna & J. Mallat (2011). Scattering analysis for signal processing

https://www.mathworks.com/help/wavelet/ug/wavelet-scattering.html

Abstract

Predictive maintenance has emerged as a valuable approach to reduce unplanned downtime and minimize maintenance costs in industries reliant on rotating machinery. By leveraging advanced machine learning techniques, it is possible to harness the wealth of vibration data collected from these critical assets to forecast their future operational state. In this project, the goal was to exploit a extensive vibration dataset for rotating machinery in order to train models capable of predicting the remaining useful life and impending failures of the equipment. To achieve this, we employed two powerful machine learning algorithms - support vector machines (SVM) and logistic regression classifiers. The SVM model was trained to estimate the remaining lifespan of the rotating components, providing operators with advance warning of impending failures. Complementing this, the logistic regression classifier was used to identify which pieces of equipment are likely to fail within a specified time period, enabling proactive maintenance scheduling. These predictive models were rigorously validated and demonstrated high accuracy in forecasting the future operational state of the rotating machinery, paving the way for data-driven, condition-based maintenance strategies that optimize asset performance and minimize unplanned downtime

ملخص

برزت الصيانة التنبؤية كنهج قيم لتقليل وقت التوقف غير المخطط له وتقليل تكاليف الصيانة في الصناعات التي تعتمد على الآلات الدوارة. من خلال الاستفادة من تقنيات التعلم الآلي المتقدمة، من الممكن تسخير ثروة بيانات الاهتزاز التي تم جمعها من هذه الأصول الهامة للتنبؤ بحالتها التشغيلية المستقبلية. في هذا المشروع، كان الهدف هو استغلال مجموعة بيانات اهتزاز واسعة النطاق للآلات الدوارة من أجل تدريب نماذج قادرة على التنبؤ بالعمر الإنتاجي المتبقي والأعطال الوشيكة للمعدات. ومصنفات الانحدار اللوجستي. تم (SVM) لتحقيق ذلك، استخدمنا خوارزميتين قويتين للتعلم الآلي - دعم آلات المتجهات لتقدير العمر المتبقي للمكونات الدوارة، مما يوفر للمشغلين تحذيرا مسبقا من الأعطال الوشيكة. SVM تدريب نموذج واستكمالا لذلك، تم استخدام مصنف الانحدار اللوجستي لتحديد قطع المعدات التي من المحتمل أن تتعطل خلال فترة زمنية محددة، مما يتيح جدولة الصيانة الاستباقية. تم التحقق من صحة هذه النماذج التنبؤية بدقة وأظهرت دقة عالية في التنبؤ بالحالة التشغيلية المستقبلية للآلية الدوارة، مما يمهد الطريق لاستراتيجيات الصيانة القائمة على البيانات والقائمة على الحالة التي تعمل على تحسين أداء الأصول وتقليل وقت التوقف غير المخطط له

Résumé

La maintenance prédictive est devenue une approche précieuse pour réduire les temps d'arrêt imprévus et minimiser les coûts de maintenance dans les industries dépendantes des machines tournantes. En tirant parti des techniques avancées d’apprentissage automatique, il est possible d’exploiter la richesse des données vibratoires collectées sur ces actifs critiques pour prévoir leur état opérationnel futur. Dans ce projet, l'objectif était d'exploiter un vaste ensemble de données sur les vibrations des machines tournantes afin de former des modèles capables de prédire la durée de vie utile restante et les pannes imminentes de l'équipement. Pour y parvenir, nous avons utilisé deux puissants algorithmes d'apprentissage automatique : les machines à vecteurs de support (SVM) et les classificateurs de régression logistique. Le modèle SVM a été formé pour estimer la durée de vie restante des composants rotatifs, fournissant ainsi aux opérateurs un avertissement préalable en cas de panne imminente. En complément, le classificateur de régression logistique a été utilisé pour identifier les équipements susceptibles de tomber en panne dans un délai spécifié, permettant ainsi une planification de maintenance proactive. Ces modèles prédictifs ont été rigoureusement validés et ont démontré une grande précision dans la prévision de l'état opérationnel futur des machines tournantes, ouvrant la voie à des stratégies de maintenance basées sur les données et basées sur l'état qui optimisent les performances des actifs et minimisent les temps d'arrêt imprévus.