FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Identification of Rail Faults using AI

Pedro Miguel da Cunha Morais

WORKING VERSION

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Gil Manuel Gonçalves

July 28, 2023

Abstract

The efficient prediction of faults in railway sections is of utmost importance for ensuring safe and reliable transportation. This dissertation aims to develop a functional AI model capable of accurately predicting the presence of faults in railway tracks. To achieve this objective, various Machine Learning models are constructed and evaluated to identify the most accurate approach for the given problem.

The methodology for rail fault prediction encompasses several critical steps, starting with data preprocessing, where simulated data is generated based on a geometric representation of the railway track, generating Irregularity and acceleration datasets. By utilizing Irregularity features, a binary label is obtained to signify the presence (1) or absence (0) of faults, enabling supervised learning.

Three distinct methods are proposed for the Machine Learning models. The first method involves training a single classifier on acceleration data, using the generated labels as the target. The second approach employs a regressor trained on acceleration data to predict irregularities, which are subsequently compared to normative thresholds to generate the predicted fault labels. The third method incorporates a regressor and a classifier, leveraging transfer learning to enhance model performance.

Despite extensive efforts, the obtained results did not meet the desired expectations. The best-performing model achieved an accuracy of 59.39%, with a true positive rate at 70.16% and a true negative rate at 51.95%, and a precision score of 50.19%. This model was built using a Keras Neural Network Regressor and classification using normative tables. While showing promise, this level of performance falls short of practical real-life applications. Method 2 and 3, using regression to predict the irregularity indicators from the acceleration values, displayed potential and need further investigation.

Due to the unavailability of real-world data, the models had to rely solely on simulated data, which may have limited their performance in capturing real-world complexities. Future work should prioritize obtaining authentic data to enhance the model's applicability and accuracy in practical railway fault prediction scenarios.

Keywords: Artificial Intelligence, Machine Learning, Fault Prediction, Condition Monitoring, Transfer Learning

Acknowledgements

To my incredible family, thank you for being my rock, my guiding light, and my constant source of love. Your presence in my life is a treasure beyond measure, and I am forever grateful for the bond we share. I love you all deeply and unconditionally. Without you, none of what I have achieved would be possible.

To all my closest friends from Paredes, thank you for being the constants in my life, the ones who have laughed, cried, and grown with me. Your friendship is a cherished gift, and I am grateful for the love and support you have always shown me. Here's to many more adventures and memories together.

To all the friends I've made in college, thank you for all of the support and help in the hardest moments. A special thank you to Alex who has been there since day 1 of college, I can never repay you for how much you've helped me these past 5 years.

To all the professors who have guided and inspired me throughout my academic journey, from high school to college, and not forgetting those in SabiUs. Your impact on my life is immeasurable, and I am forever grateful for the knowledge and skills you've provided.

To my advisor, Gil Gonçalves, thank you for all of your patience and determination and for listening to all the problems, I'm very grateful for your help. A special thank you to Pedro Montenegro as well for providing all of the necessary data and domain knowledge to develop this thesis.

"Failure is an option here.
If things are not failing, you are not innovating enough."

Elon Reeve Musk

Contents

List of Figures

List of Tables

Abbreviations

ADAM	Adaptive Moment Estimation
AI	Artificial Intelligence
AL	Alert Limit
ANN	Artificial Neural Network
AR	Autoregressive
CEN	European Committee for Standardization
CNN	Convolutional Neural Network
CP	Comboios de Portugal
CRA	Collaborative Recommendation Approach
CRISP-DM	Cross Industry Standard Process for Data Mining
GOSS	Gradient-based One-Side Sampling
GPU	Graphic Processing Unit IL
Intervention Limit	
IAL	Immediate Action Limit
IoT	Internet of Things
k-NN	k-Nearest Neighbors
LSTM	Long Short-Term Memory
MC	Multiple Classifier
ML	Machine Learning
MLP	Multi-Layer Perceptron
PCA	Principal Component Analysis
PdM	Predictive Maintenance
PvM	Preventive Maintenance
R2F	Run-to-Failure
RBF	Radial Basis Function
ReLu	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
SDSS	Smart Decision Support System
SVM	Support Vector Machine
TPR	True Positive Rate
TPU	Tensor Processing Unit TNR
True Negative Rate	
VR	Virtual Reality
WSN	Wireless Sensor Network

Chapter 1

Introduction

The railway sector is an indispensable aspect of modern transportation, serving a crucial function in transporting goods and people worldwide. The industry has witnessed remarkable growth and progress over the years, including the advent of high-speed trains, electrification, and the adoption of digital technologies. Nevertheless, the industry is confronted with significant challenges, particularly in terms of maintenance and safety, which adversely impact its overall efficiency and productivity.

The primary obstacle faced by the railway industry is the requisite for regular maintenance. The railway network is enormous, with thousands of miles of tracks, numerous stations, and a high number of trains in service. The maintenance of this infrastructure necessitates significant resources, including time and money. Furthermore, conventional manual inspection procedures are laborious and time-consuming, resulting in significant downtime and service disruptions.

Moreover, ensuring the safety of passengers and employees has become a major challenge for the railway industry. Several high-profile accidents in recent years have emphasized the need for improved safety measures. Although accidents can happen due to various reasons, such as human error, improper maintenance or infrastructure failures can also lead to such incidents. The safety issue is further intensified by the limitations of conventional manual inspection methods. These methods are often not effective in identifying faults and failures, resulting in service disruptions and accidents.

In recent years, the application of artificial intelligence (AI) techniques has emerged as a promising solution to address the challenges of railway maintenance and safety. Due to their potential to automate and speed up the inspection process and thus enable more effective and efficient maintenance practices, AI-based fault identification systems in particular have received significant attention.

1.1 Background and Context

1.1.1 Context of the dissertation

This dissertation is inserted in the context of the "FERROVIA 4.0" project. The "FERROVIA 4.0" project, which involves businesses and Investigation and Innovation entities led by EFACEC Engineering and Systems and the Cluster of the Portuguese Ferroviary Platform, aims to address technological and market challenges that have been placed on the ferroviary sector [2].

The ambition of the project is to develop different components, tools, and systems oriented to the economic and ecological sustainability of the railway system, to the reduction of operating and maintenance costs, to the creation of reliable information systems to support decision-making in asset management, and to the creation of safety systems capable of monitoring the infrastructure and triggering alerts and protection/intervention measures.

1.1.2 Historical Background on the Portuguese Railway Infrastructure

The history of the Portuguese railway system dates back to the mid-19th century when the first railway line was inaugurated in 1856, connecting Lisbon to Carregado. Over the years, the network expanded, driven by the need for efficient transportation and improved connectivity. The early railway lines were primarily constructed by private companies, but in 1947, the nationalization of the railways led to the formation of Portuguese Railways (CP - Comboios de Portugal), the national railway operator responsible for managing the network.

The Portuguese railway network is well-developed and covers the majority of the country as shown in Figure 1.1 [19]. It is organized into several main lines, regional lines, and suburban lines, catering to different transportation needs. The main lines connect major cities and towns, while the regional lines serve smaller communities and provide links to rural areas. The suburban lines focus on commuter traffic in urban areas, particularly around Lisbon and Porto.

The tracks are classified based on gauge, with the Iberian gauge ($1668mm$) being the most prevalent. The rolling stock consists of a total of 264 trains that operate on the Portuguese railway network. These include high-speed trains - the "Alfa Pendular", regional trains, suburban trains, intercity trains, and freight trains [20]. The "Alfa Pendular" reaches a maximum speed of $220km/h$ which is the highest possible value in the Portuguese railway infrastructure. However, this high-speed train can only reach this speed in three short sections that make up a total of $78km$ of the railway [15].

1.1.3 Rail Faults

A major cause of railway accidents and derailments is track geometry irregularities, which refer to deviations from the desired track geometry. These irregularities can be caused by a range of factors, such as wear and tear, environmental conditions, and insufficient maintenance.

These irregularities can manifest in different forms, including wide gauge, excessive warp/twist, and horizontal and vertical rail deformities. Wide gauge refers to a condition where the distance

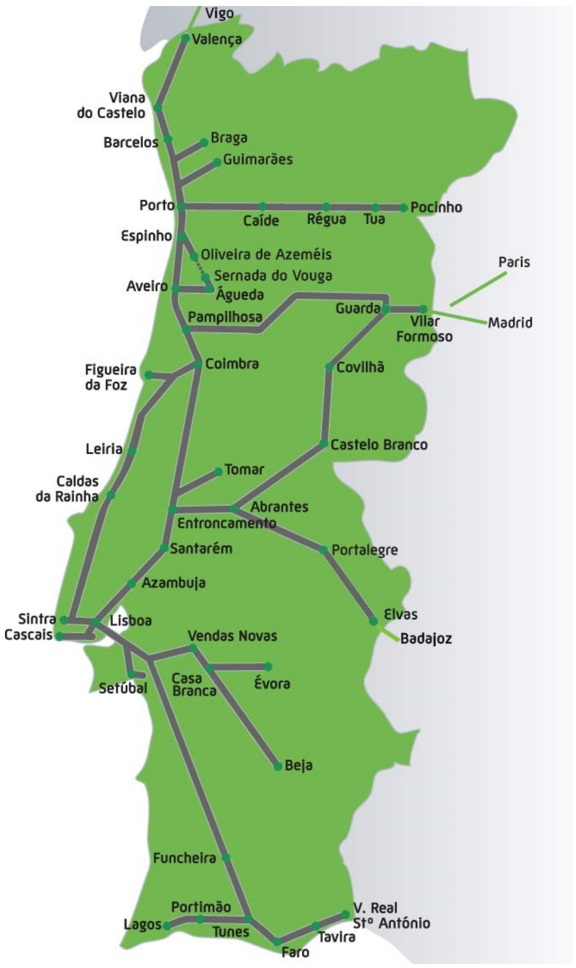


Figure 1.1: Portuguese Railway Network [19]

between the rails is wider than the standard gauge, which can lead to stability issues and derailments. Excessive warp/twist refers to a condition where the rail deviates from a straight line, also leading to instability and derailments. Horizontal and vertical rail deformities can result from wear and tear or manufacturing defects. If left unaddressed, all these irregularities can ultimately result in broken rails and derailments, posing a serious risk to railway safety. Figure 1.2 demonstrates the most normal types of track irregularities [49].

The CEN (European Committee for Standardization) defined 3 levels of hazardous situations in relation to track geometry quality for railway applications [22]. The immediate action limit (IAL), intervention limit (IL), and alert limit (AL) are all critical thresholds in railway maintenance management. The IAL corresponds to the value that, if exceeded, requires immediate action to be taken to reduce the risk of derailment to acceptable levels. This may involve closing the line, reducing speed, or correcting track geometry. The IL corresponds to the value that, if exceeded, requires corrective maintenance action to be taken so that the IAL cannot be reached before the next inspection. The AL corresponds to the value that, if exceeded, requires a detailed analysis of the track geometry condition and consideration in scheduled maintenance works.

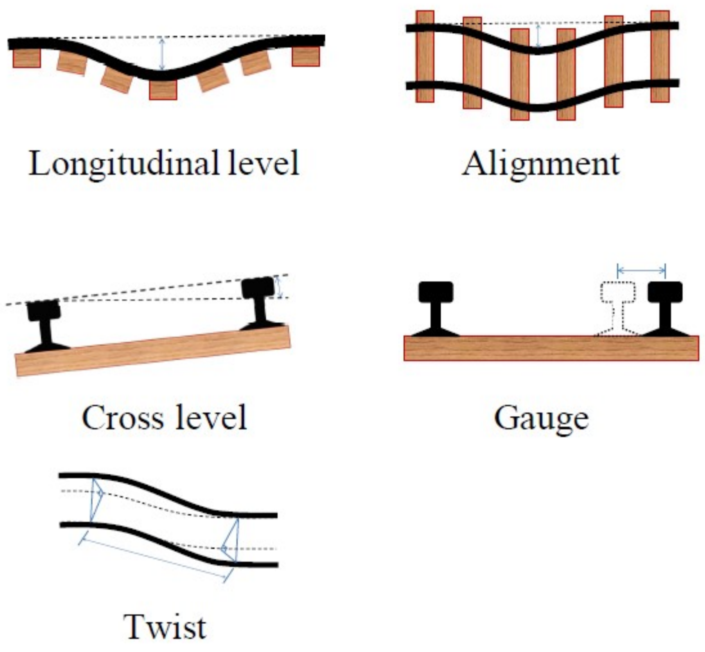


Figure 1.2: Examples of Track Irregularities [49]

1.1.4 Artificial Intelligence and Machine Learning

The application of Artificial Intelligence and Machine Learning (ML) techniques in the railway industry has gained significant interest in the past few years. These technologies have the potential to transform the way maintenance and safety are managed in this industry. Through the use of AI and ML, it is possible to predict potential faults and failures before they occur, allowing maintenance teams to take necessary measures to prevent downtime and enhance overall efficiency.

By leveraging advanced machine learning algorithms, these fault identification systems can analyze vast amounts of data collected from various sources, such as track sensors, cameras, and historical maintenance records. The AI algorithms can detect anomalies, deviations, and potential faults in the railway infrastructure, including tracks, switches, signaling systems, and rolling stock. This enables early identification of issues before they escalate into more significant problems, reducing the risk of accidents and minimizing service disruptions.

Additionally, AI-based fault identification systems have many of benefits over standard manual inspection techniques. They can operate continuously and in real-time, providing constant monitoring and evaluation of the railway network. This proactive approach helps with early fault detection and correction, potential failure prevention, and reduced need for unscheduled maintenance activities. In addition, AI systems are able to analyze historical data patterns and trends, enabling predictive maintenance approaches that maximize resource allocation and cut costs.

In conclusion, AI-based fault identification systems have enormous potential to revolutionize the maintenance and safety procedures used in the railroad industry. These systems can automate and streamline the inspection process by applying AI and ML. This enables the early detection of faults and proactive maintenance strategies. In the long run, both the industry and its customers will profit from the successful application of these techniques, which will increase the effectiveness, dependability, and safety of railway operations.

1.1.5 Maintenance Strategies

As per [11], there are normally three types of maintenance strategies: Run-to-Failure (R2F) which is unthinkable in the context of detecting faults in railways; Preventive Maintenance (PvM) which is a time-based or scheduled type of maintenance. It is performed periodically and sometimes unnecessary corrective actions are taken, leading to increased operating costs; Predictive Maintenance (PdM) uses predictive tools, in this case, Machine Learning methods, to determine when maintenance actions are necessary. Here, continuous monitoring is also necessary as it allows maintenance to be performed only when needed, permitting the early detection of failures.

1.2 Objective of the study

The main objective of this dissertation is to build a working AI model that can effectively predict the existence of faults in given sections of the railway, in whom the model is implemented. To reach this goal, different models are built and evaluated to find the most accurate model possible for the problem at hand.

When it comes to building AI models, one indispensable requirement stands out: the data. To build the models, datasets of simulated data were supplied consisting of two different types: track-related irregularities and train-related accelerations. The data was elaborated on the basis of the geometric drawing of the track, corresponding to the 5th section of the Portuguese Northern line with approximately 5400m of length. This geometric drawing takes into account the curvature and the superelevation of the rails so that the simulated data is as close to real data as possible.

In a real-life scenario, track-related irregularities are collected via a specialized inspection train called EM120 [1], represented in figure 1.3. This train executes scheduled passages on the track and collects tons of data about the overall state of the track. A special machine in this train collects four features that make up the simulated data related to the irregularities. These are the left and right rail horizontal alignment and the left and right rail vertical level.

The second dataset related to accelerations is much more easily obtained. Any train can be equipped with accelerometers to capture the oscillations of the acceleration of the train. In the simulated scenario, there are four accelerometers placed in the axle boxes of a train wagon. Each of these accelerometers collects horizontal and vertical accelerations, resulting in a total of eight features for the accelerations dataset.

The data can be easily labeled with the use of the thresholds provided by the CEN in regard to the alert limit [22]. These indicate acceptable standard deviation levels of the irregularities, based on the speed of the train. Any standard deviations exceeding the respective thresholds are classified as faults, i.e., ’1’ while those within the limit are classified as ’0’. These standard deviation values must be calculated in a length of 200m of the track. This process of label assignment is

Figure 1.3: EM120 Inspection Train [1]

an indispensable part of supervised Machine Learning, providing the ground truth upon which the model will learn and be evaluated. The mentioned thresholds are represented in the following tables, 1.1 and 1.2.

Table 1.1: Longitudinal Level - Alert Limit

Speed (in km/h)	Standard Deviation (in mm)
$V \leq 80$	2.3 to 3
$80 < V \leq 120$	1.8 to 2.7
$120 < V \leq 160$	1.4 to 2.4
$160 < V \leq 230$	1.2 to 1.9
$230 < V \leq 300$	1.0 to 1.5

Table 1.2: Alignment - Alert Limit

Speed (in km/h)	Standard Deviation (in mm)
$V \leq 80$	1.5 to 1.8
$80 < V \leq 120$	1.2 to 1.5
$120 < V \leq 160$	1.0 to 1.3
$160 < V \leq 230$	0.8 to 1.1
$230 < V \leq 300$	0.7 to 1.0

As explained, the damage classification of the railway is easy to do, when there is access to the irregularity data. The ambition of this thesis is to find a way of classifying the state of the track exclusively using acceleration data as an input, which is much easier to get but is not directly related to the existence of faults in the rails.

1.3 Significance of the study

The significance of this study lies in its potential to improve railway safety and reduce the number of accidents and derailments caused by track geometry irregularities. Accidents and derailments can have serious consequences, including loss of life, injury, and damage to equipment and infrastructure. By identifying the state of the rails and predicting their remaining useful life, this study could help to make railway travel safer and more reliable.

In addition, this study could have economic benefits, as accidents and derailments can result in significant costs to railway operators and the wider economy. Also, as per [32], railway companies spend billions worldwide in preventive maintenance actions that sometimes don't even need to happen. Using a model that can predict exactly when maintenance is necessary also helps reduce the cost of these maintenance actions, by avoiding unnecessary ones. Hence, this study could help to reduce these costs and promote more efficient and cost-effective railway operations.

Furthermore, being able to directly connect acceleration values to rail faults would mean that the hassle of having a specialized train do the recon of faults would not be necessary as any normal train could have the ability to detect rail faults.

Overall, this study has the potential to make a significant contribution to improving railway safety and reliability, with potential benefits for passengers, railway operators, and the wider economy.

1.4 Structure of the Document

The remainder of this thesis is organized as follows: Chapter 2 provides a literature review related to railway infrastructure condition monitoring, fault prediction, and the use of different ML models; Chapter 3 presents an overview of the methodology used to accomplish the goal of this thesis; Chapter 4 presents an in-depth look into the methodology used for data collection and preprocessing while also describing the different ML methods used to develop predictive models. A quick description of the workings of each different ML model is also presented; Chapter 5 presents the results of the models, including their performance over different testing scenarios, and presents conclusions on these results; Finally, Chapter 6 concludes the thesis by summarizing the key findings and outlining possible avenues for future work.

Chapter 2

State of the Art

The maintenance of railway systems is a critical task to ensure the safety and efficiency of the transportation network. PdM using Machine Learning is an emerging field that has the potential to revolutionize the maintenance practices of various industries, including transportation. Several recent studies have focused on using machine learning techniques to detect and classify faults in all kinds of systems.

2.1 Introduction

The safety and availability of railway networks depend on accurate fault prediction in railway systems. For maintaining secure and dependable train operations, it is essential to promptly detect and identify faults in railway track circuits [8]. Train delays, maintenance costs, and even fatal accidents can result from faults in railway tracks. In order to detect and diagnose faults in railway systems, it is crucial to develop appropriate condition-monitoring techniques [25].

Due to the potential impact on train operations and passenger safety, identifying faults in railway tracks is of the utmost importance. Rail transportation can be seriously threatened by factors like cracks, ballast problems, rail discontinuity, loose nuts and bolts, burnt wheels, superelevation, and misalignment. It is ineffective and prone to bias and human error to manually inspect railroad tracks with a railway cart using traditional methods. To prevent accidents and save lives, automated fault detection methods for railway tracks are therefore required [44].

In conclusion, it is essential for ensuring the availability and safety of railway networks that fault prediction in railway systems be implemented. To avoid accidents and maintain dependable train operations, it's critical to identify faults in railroad tracks. The use of AI models for fault prediction and diagnosis in railway systems has produced encouraging results. These models are accurate fault detectors and can learn from measurement signals. To get around the shortcomings of manual inspection techniques, automated approaches to fault detection in railway tracks are required.

2.2 AI Models for Fault Prediction

Railway systems have used a variety of methods and techniques for fault prediction. Manual inspection techniques, such as visual inspection of railroad tracks with a railroad cart, are part of traditional methods. These techniques are laborious, time-consuming, and subject to biases and human error. Automated methods, such as the use of acoustic analysis [44], wireless sensor networks (WSN) [26], and ML models [34] have been developed to get around these restrictions. In order to identify faults, acoustic analysis is used to examine the sound signals produced by railroad tracks. To keep track of the state of the equipment and infrastructure on railroads, WSNs use a network of Wireless Sensors.

More and more AI models are being used to predict track faults on railways. Recurrent Neural Networks (RNN), for instance, have demonstrated promising results in the fault diagnosis of railway track circuits. RNNs, particularly the Long Short-Term Memory (LSTM) network, can identify faults accurately by learning the spatial and temporal dependencies present in measurement signals [8].

The current methods for predicting faults in railroad systems have advantages and disadvantages. According to [8], using AI models, such as RNNs and deep learning models, has demonstrated high accuracy in fault detection and identification. These models are able to deduce intricate dependencies and patterns from data, leading to precise predictions. WSNs can also be used to continuously monitor railway equipment and vehicles, obviating the need for manual inspections and facilitating the early detection of faults [26]. The use of these strategies, though, is not without its difficulties. For instance, it may be difficult to find high-quality, labeled data to train AI models [18]. In addition, it can be difficult and expensive to deploy and maintain WSNs in large-scale railway systems. Furthermore, because AI models frequently operate as "black boxes," it can be challenging to understand the underlying causes of their predictions [8]. Despite these drawbacks, using AI models and automated fault prediction methods in railway systems has a lot of potential to increase the security and dependability of rail networks.

Support Vector Machines (SVM), Logistic Regression (LR), Random Forest (RF), Decision Tree (DT) classifiers, Artificial Neural Networks (ANN), and Convolutional Neural Networks (CNN) are some other AI techniques that have been used for fault detection in railway systems [44][33].

The authors of [31] conducted a systematic review of 109 articles and were able to identify three main application domains and nine different technologies that are currently being used in the industry. The application domains include monitoring, decision and planification techniques, and communication and security, while the technologies being used are AI, Internet of Things (IoT), Cloud Computing, Big Data, Cybersecurity, Modelling and Simulation, Smart Decision Support Systems (SDSS), Computer Vision, and Virtual Reality (VR). The paper concludes that the integration of Industry 4.0 technologies in railway transportation has significant potential for addressing various issues, including train failures, train station security, rail system control, and communication in hard-to-reach areas. Then, a literature review by [11] analyzed the use of machine learning methods in predictive maintenance. They identified that the most commonly used techniques were Support Vector Machines, Random Forests, Artificial Neural Networks, and K-means. The review also highlighted the importance of feature selection in the performance of the models. The authors suggest that future research should focus on developing hybrid models that combine multiple machine-learning methods for better prediction accuracy. Another study by [17] investigated the use of machine learning and reasoning techniques in predictive maintenance for Industry 4.0. The authors found that the most commonly used algorithms were RF, SVM, and k-Nearest Neighbors (k-NN). The study also identified that the integration of human knowledge and reasoning with machine learning methods can improve the accuracy of predictive maintenance models.

Different AI models have frequently been employed in railway systems to predict faults. Recurrent neural networks, like the Long Short-Term Memory network, are among the models that use them. These RNNs have shown promising results in the fault diagnosis of railway track circuits because they can capture temporal dependencies in the data. Convolutional neural networks, random forest, decision tree classifiers, support vector machines, logistic regression, and other AI models have also been used for fault detection in railway systems [48]. In order to analyze vast amounts of data and find patterns suggestive of faults in railway tracks, these models make use of the power of AI algorithms.

AI models have been used in several studies to forecast track faults in railroads. For instance, [44] developed an automatic railway track fault detection system using acoustic analysis and applied different classification techniques, such as SVM, logistic regression, random forest, and decision tree classifiers, to detect various types of faults in railway tracks. To detect and categorize track faults based on car-body vibrations, a track condition monitoring system was created [48]. This system used machine learning techniques, including SVM. These studies highlight the potential for enhancing the safety and dependability of railway systems and show how AI models are effective at predicting faults in railway tracks.

The use of image analysis in the predictive maintenance of railway systems has also gained attention in recent years. For example, [23] proposed the use of deep convolutional neural networks for the detection of rail surface defects. The study used a large dataset of images of rail surfaces to train the CNN model. The results showed that the proposed approach could accurately detect various types of defects. Another study by [4] also proposed using image processing and deep learning to detect and classify these defects. The method, however, involved multiple deep learning models for feature extraction, feature combining, and classification using SVMs. This approach is expected to be more effective in detecting rail defects compared to single deep learning models, especially under low contrast conditions. Furthermore [56], presented a deep transfer learning framework for detecting rail surface cracks using a limited amount of training images.

The authors of [54] developed ML models for railway inspection including a feature-based method and a deep neural network-based method. The models were intended to detect rail defects, such as localized surface collapse, rail end batter, or rail components, using acceleration data. The paper emphasizes the importance of smart railway maintenance and the successful deployment of technologies such as condition-based monitoring and predictive maintenance to improve operation safety and efficiency.

In [37], the authors present an unsupervised methodology for identifying railway wheel flats based on acceleration data evaluated on the rails. The proposed algorithm involves a two-step procedure that builds a confidence boundary using baseline responses evaluated from the rail and classifies damages based on different severity levels. The methodology is based on machine learning and includes steps such as data acquisition from sensors, feature extraction using an Autoregressive (AR) model, feature normalization using Principal Component Analysis (PCA), data fusion, and unsupervised feature classification using outlier and cluster analyses. The authors also investigate whether the number of sensors used to detect and classify wheel flats can be optimized.

Similarly, in [32] the authors propose a track quality index that uses machine learning to estimate the quality of railway tracks. They also use PCA to extract the most important features and then employ an SVM classifier to predict the track quality. Using this technique it was found that there are three key principal components that can summarize the whole quality of the track. These are vertical, longitudinal, and transverse irregularities. The results show that their approach can accurately estimate the track quality and outperforms other methods.

In another study that explores the use of ML for predictive maintenance [47] the authors introduce a new methodology for PdM based on multiple classifiers (MC) for integral type faults, which are failures caused by the cumulative "wear and tear" effects on equipment parts. The authors note that supervised solutions are generally preferable in PdM problems. Regression-based formulations arise when predicting the Remaining Useful Life of equipment, while classification-based formulations occur when seeking to discriminate between healthy and unhealthy conditions. The proposed methodology is demonstrated for a semiconductor manufacturing ion implanter-related maintenance task and shown to outperform classical PvM approaches and a single SVM classifier distance-based PdM alternative. The authors also found that SVMs offer superior performance to k-NN classifiers when implementing MC-PdM and that both consistently outperform PvM approaches.

Vibration signals are another type of data that is very valuable in this area. In [21] the authors propose a PdM methodology for wind turbines using vibration signal analysis based on a collaborative recommendation approach (CRA). The authors apply models such as k-NN, Support Vector Machines, and k-Means to classify the type of fault in the system. Vibration signal analysis is performed to extract the behavioral pattern of the bearings. The CRA is then applied to suggest in advance the replacement and correction of the deteriorating units and prevent severe system breakdowns and disruptions. [45] reviews recent literature on machine learning models used for condition monitoring in wind turbines. The authors come to the conclusion that the most commonly used algorithms include neural networks, support vector machines, and decision trees. [46] uses vibration signals collected from vibration acceleration sensors, which are distributed along the railway. The feature parameters are extracted from both the time domain and time-frequency domain and then the sequential backward selection method is used to select the important features. After optimizing the feature parameter set, SVM is applied for recognizing and classifying rail defects. In [49], the author proposes a ML technique for condition monitoring of railway tracks using car-body vibrations. He uses a compact onboard sensing device to collect data. Then, SVM is used to classify the vibrations and estimate the track condition. Simulation studies and field tests were carried out to detect and isolate track faults from car-body vibration. Another study that uses car-body vibrations for track geometry estimation is presented in [40]. The authors proposed a technique to monitor the condition of railway tracks using car-body acceleration data, which can be easily measured by in-service vehicles. The technique uses inverse dynamics to estimate track irregularities and applies a Kalman filter to solve this problem. The estimation technique supports a change in vehicle velocity by selecting an appropriate impulse response in the measurement equation. The proposed technique was tested in simulation and full-scale tests, and the results showed that it is effective for track condition monitoring with acceptable accuracy for conventional railways.

2.3 Datasets for Fault Prediction

Various types of data gathered from sensors or measurements typically make up the datasets used in fault prediction studies for railway systems. These datasets could contain details about the track geometry, train or vehicle vibration or acceleration data, acoustic signals, and other pertinent parameters. These datasets' characteristics can change depending on the particular study and the kind of predicted fault. For instance, datasets may include details about train accelerations, track irregularities, and other elements that point to track faults [44][9]. The datasets' sizes can also differ, ranging from small datasets gathered from particular locations to large datasets gathered from numerous railway systems [48][57]. The availability and quality of labeled data are important considerations in fault prediction studies, as they impact the training and evaluation of AI models [52].

In railway systems, track-related irregularities and train-related accelerations are crucial factors in fault prediction. Cracks, ballast problems, rail discontinuity, loose nuts and bolts, burnt wheels, superelevation, and misalignment are just a few examples of track-related irregularities that may be signs of track faults. Monitoring and examining these irregularities can aid in early fault detection and diagnosis, enabling prompt maintenance and accident prevention. Measured by sensors or onboard equipment, train-related accelerations offer important details about the dynamic behavior of the train and its interaction with the track. Acceleration patterns that change or accelerate abnormally may be a sign of track faults or irregularities [9]. Therefore, it is essential for precise and efficient fault detection in railway systems that track-related irregularities and train-related accelerations are included in fault prediction datasets.

In [53], the authors reviewed the research status of rolling bearing fault diagnosis for railway vehicles, including vibration fault diagnosis and acoustic signal fault diagnosis methods. These studies demonstrate the use of diverse datasets and AI models for fault prediction in railway systems, highlighting the importance of data collection and analysis in improving the safety and reliability of railway networks.

2.4 Challenges and Opportunities

There are many difficulties and restrictions associated with fault prediction using acceleration data. Since acceleration data can be gathered from numerous sensors and in numerous dimensions (for example y, and z axes), this presents a challenge. This high-dimensional data can make the analysis more complex and demand more computing power. The noise in the acceleration data is another issue because it can reduce the precision of fault prediction models. Before training the models, noise may need to be removed or reduced using filtering techniques or signal processing techniques [36].

Despite the difficulties, there are a number of potential advantages and opportunities when using acceleration data exclusively for fault classification. The system's dynamic behavior is revealed by acceleration data, which also records vibrations and movements that might be fault indicators. Fault classification models can be created to be more specific and sensitive to fault-related patterns by concentrating on acceleration data, potentially increasing the accuracy of fault detection. Furthermore, acceleration data can be gathered in real-time or very close to real-time, enabling continuous monitoring and quick fault detection. This can improve the safety and dependability of railway systems by allowing for timely maintenance and accident prevention [29].

2.5 Conclusion

This literature review revealed several key findings in the field of fault prediction, condition monitoring, and predictive maintenance in railway systems. Studies have highlighted the importance of fault detection and diagnosis in ensuring the safety and reliability of railway networks. The use of AI models has shown promising results in fault prediction and classification. These models leverage various types of data, including track geometry data, vibration or acceleration data, and acoustic signals, to accurately detect and classify faults in railway tracks. However, challenges such as high dimensionality of data and the presence of noise need to be addressed. Overall, the literature emphasizes the potential of AI models and automated techniques in improving the safety and reliability of railway systems. The integration of human knowledge and reasoning with ML methods can improve the accuracy of predictive maintenance models.

Despite the progress made in fault prediction research, there are still gaps and limitations that need to be addressed. One major gap is the need for more comprehensive and diverse datasets that accurately represent the real-world conditions of railway systems. The availability of high-quality labeled data is crucial for training and evaluating AI models. Additionally, there is a need for standardized evaluation protocols and benchmarks to facilitate fair performance comparison among different models. Another gap is the interpretability of AI models, as they often function as black boxes, making it difficult to understand the underlying reasons for their predictions. Furthermore, the integration of fault prediction models with maintenance and repair strategies is an area that requires further exploration. Future research should also focus on developing hybrid models that combine multiple ML methods and data sources for better prediction accuracy. By addressing these gaps, researchers can advance the field of fault prediction in railway systems and develop more effective and reliable approaches.

Chapter 3

Methodology

This chapter provides an overview of the methodology that will be employed to address the research problem. However, it's important to note that this chapter only offers a concise presentation of the methodology. A comprehensive and detailed description of each step will be presented in the subsequent chapter. The methodology in this thesis follows the Cross Industry Standard Process for Data Mining (CRISP-DM) which is a process model that serves as the base for a data science process. Figure 3.1 has a representation of the process.

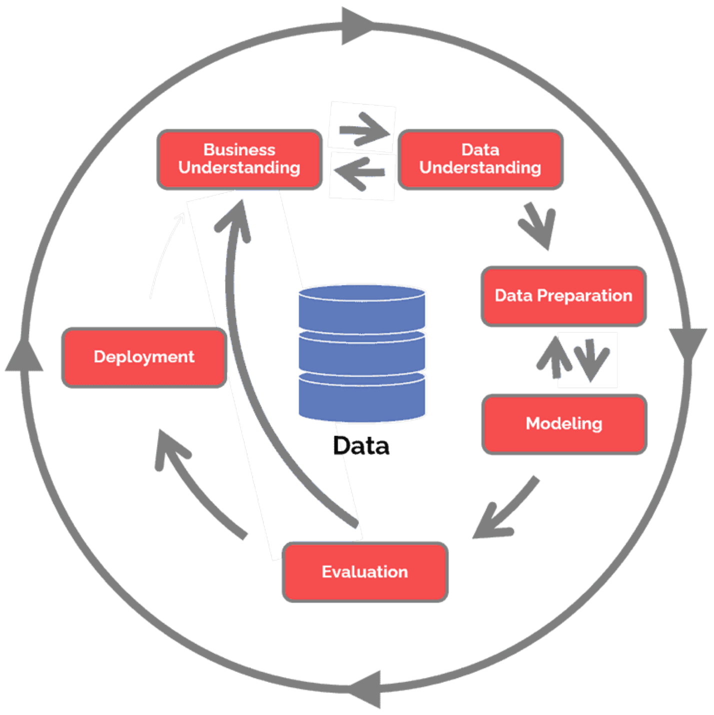


Figure 3.1: CRISP DM Diagram [28]

This process has six main steps that aim to answer different questions:

1. **Business understanding:** What does the business need?

   This question is already answered in the first chapter.

2. **Data understanding:** What data do we have/need? Is it clean?

   First, simulated data is generated based on a geometric drawing of the railway track. Irregularities and accelerations are obtained for nine different scenarios. In the early stage of this work, there was an expectation to have access to real-world data for training the models. However, during the course of the project, it became apparent that obtaining such real data would not be possible, for the time being. Consequently, the models had to be trained and tested solely on simulated data due to the unavailability of authentic data. While using simulated data introduces certain limitations, the focus shifted towards optimizing the models to make the most out of the available simulated dataset for accurate rail fault prediction.

3. **Data preparation:** How do we organize the data for modeling?

   First, the irregularity and acceleration datasets are aligned through interpolation to create unified datasets, allowing for a comprehensive analysis of their relationship along the track. The features of the irregularity data are then used to obtain a binary label (0 - no fault or 1 - fault), according to tables 1.1 and 1.2, which is a vital step for supervised learning. By comparing the irregularities with normative thresholds, fault labels are generated for each data point, and now both irregularities and accelerations have an associated label. In the last step, data scaling is applied to ensure all features are on a consistent scale, which enhances the machine learning model's performance. Additionally, a train-test split is applied to the datasets. To ensure a balanced representation for training and evaluation, all the individual datasets were concatenated. It is important to note that the individual datasets themselves are not balanced, as will be detailed in the subsequent chapter. By combining the datasets, the resulting unified dataset provides a more comprehensive and equitable distribution of data, allowing for more reliable model training and testing.

4. **Modeling:** What modeling techniques should we apply?

   Three methods are proposed for the ML models. The first method involves building a single classifier that is trained on acceleration data, using the labels generated from irregularities as the target. The second approach includes training a regressor using acceleration data as input and irregularities as the target. The regressor's predictions for irregularities are then compared to the normative thresholds to generate the fault labels. The third method incorporates two models: a regressor and a classifier. The regressor is trained on acceleration data to predict irregularities. These predicted irregularities, along with the original acceleration data, are then used to train the classifier with theoretical fault labels as the target. This approach leverages transfer learning in an effort to improve the model's performance.

5. **Evaluation:** Which model best meets the business objectives?

   After training all the models, they are tested using the split previously mentioned. Then, appropriate evaluation metrics are applied to make conclusions about their performances. For regressors, the metrics used are Root Mean Squared Error, R-Squared, and Mean Absolute Error. For the classifiers, the metrics used are Accuracy, Recall, Precision, and Specificity. The goal here is to find if any of the methods/models show a good balance between all the metrics and if they show potential to be used in a real-world application.

6. **Deployment:** How do stakeholders access the results?

   In this thesis, this particular point is not important and the results of the models will not be publicly deployed for use.

The full methodology is represented in figure 3.2 for improved understanding.

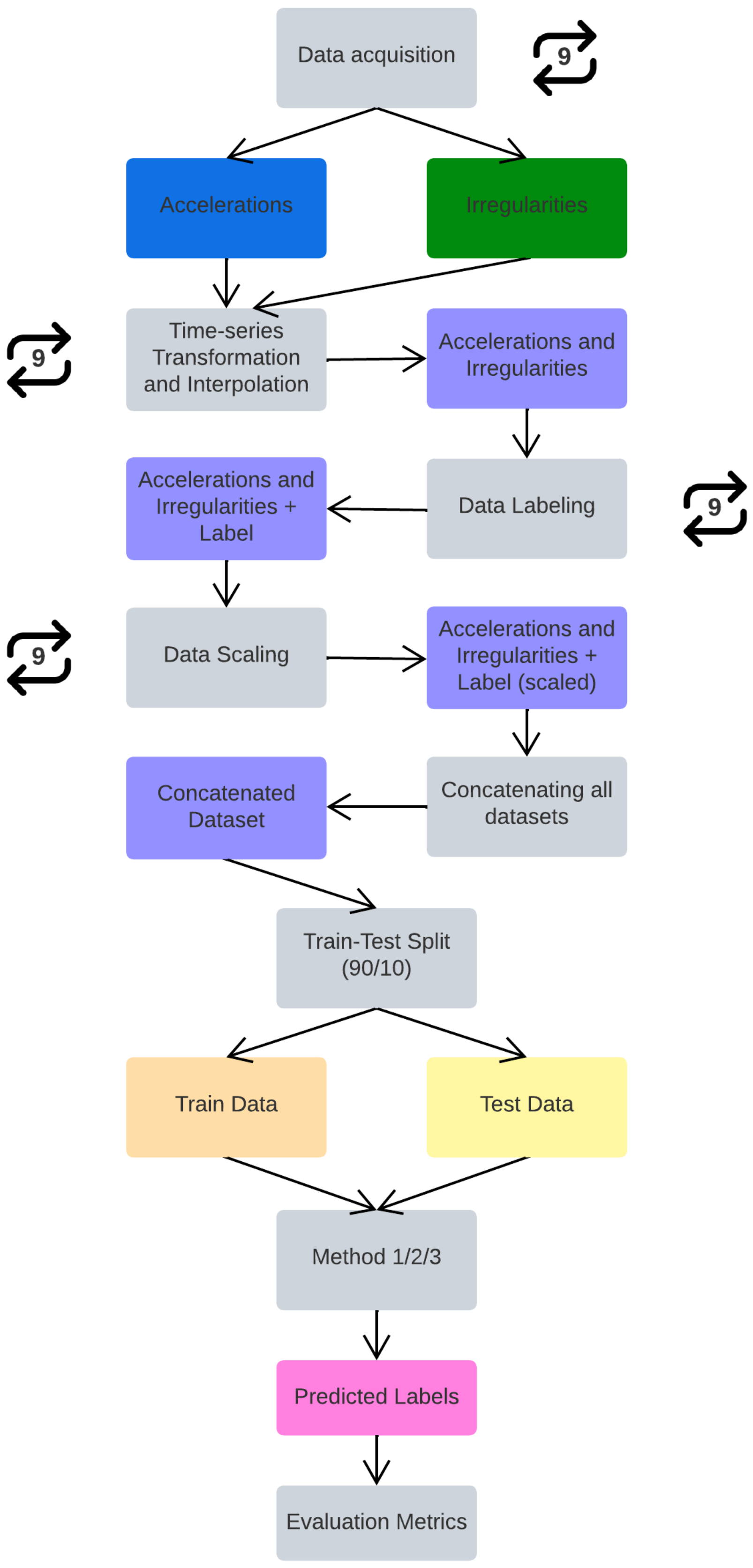

Data acquisition
9
Accelerations
Irregularities
Time-series Transformation and Interpolation
9
Accelerations and Irregularities
Accelerations and Irregularities + Label
Data Labeling
9
Data Scaling
9
Accelerations and Irregularities + Label (scaled)
Concatenated Dataset
Concatenating all datasets
Train-Test Split (90/10)
Train Data
Test Data
Method 1/2/3
Predicted Labels
Evaluation Metrics

Figure 3.2: Full Methodology Diagram

Chapter 4

Exploring the solution to the problem

In this chapter, the methodology used to accomplish the goals of this thesis is thoroughly explained. As per 3, the approach is divided into several stages, each intended to address a particular component of the research objectives. This chapter provides a detailed overview of the key steps involved in this approach. The tools, algorithms, and methods used throughout the process are also reviewed.

4.1 Data preprocessing

Data preprocessing is a critical step in the Machine Learning pipeline that involves transforming raw data into a format suitable for training and evaluating models. This section includes a description of the datasets used as well as a discussion of the various data preprocessing techniques used.

4.1.1 Data acquisition and description

As mentioned before, the ML model will be trained using simulated data. The data was elaborated on the basis of the geometric drawing of the track, corresponding to the 5th section of the Portuguese Northern line with approximately $5400m$ of length. This geometric drawing already takes into account the curvature and the superelevation of the rails so that the simulated data is as close to real data as possible. The data is simulated through a program called SimPack that is representing passages of the, already mentioned, Portuguese EM120 inspection train. This train has a special onboard device that outputs the irregularities of the rail every centimeter. The train also has four different accelerometers distributed in such a way that there are two on the front axle and two on the rear axle (per axle there is one accelerometer on the left and one on the right). It's also important to note that the train makes its journey at a constant speed of $80km/h$.

As described, there are two different types of data that are used in this paper: Track-related irregularities and train-related accelerations. For the irregularities data, there are four features:

"ALE" and "NLE" refer to the left rail horizontal and vertical deviations, respectively. "ALD" and "NLD" represent the right rail's horizontal and vertical deviations. This data is obtained every centimeter.

The horizontal and vertical deviations are instrumental in examining any irregularities in the rail tracks. They act as indicators of faulty or distorted sections. Consequently, these attributes are expected to contribute significantly to the model's predictive power. By having these features for both sides of the rail track, we can form a complete image of the rail condition. When evaluated together, these features provide a balanced perspective on the structural integrity of the entire track system.

In order to use tables 1.1 and 1.2 to correctly label the data, it is necessary to calculate the standard deviations within every $200m$. To perform this, a moving window was elaborated. This window calculated the standard deviation within the first $200m$. Then, the window moves $1m$ and performs the calculations again. This way, 5200 standard deviations were obtained.

Regarding the acceleration data, things are a bit different, as the accelerometers collect data at a constant frequency of $5kHz$. Given that the train's speed is $80km/h$ there is now a new measurement approximately every $0,0444cm$. This way, each scenario has 1215000 values for each acceleration. The features present in the acceleration data are "S4_y1", "S4_z1", "S1_y1", "S1_z1", "S2_y1", "S2_z1", "S3_y1", "S3_z1" where y represents the acceleration in the horizontal axis and z in the vertical axis and 1, 2, 3 and 4 represent one of the axle boxes, according to 4.1. 1 represents the rear right wheel, 2 the front left wheel, 3 the front right wheel, and 4 the rear left wheel.

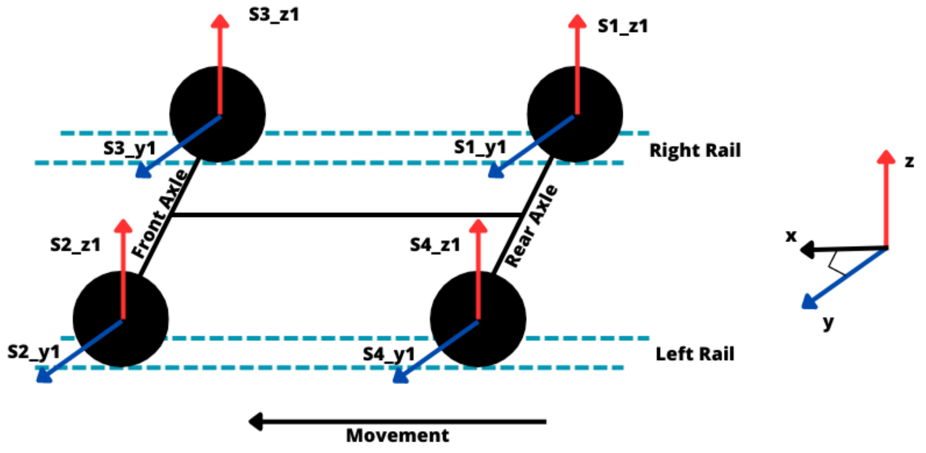


Figure 4.1: Representation of the train and the acceleration features

In order to enlarge the dataset nine different damage profiles were simulated in an effort to try and encapsulate multiple possible scenarios of ALs. From each normative table, three standard deviations are confronted with other three standard deviations from the opposite tabled, and nine different outcomes emerge. Each of these scenarios has its own set of irregularities and accelerations. Table 4.1 represents these scenarios.

Table 4.1: Simulation of Damaged scenarios

Description of Damaged Scenarios	**ALSTD for vertical level [mm]**	**ALSTD for lateral alignment [mm]**	**Vehicle Speed [km/h]**
Damaged Scenario 1	1.4	1.0	80
Damaged Scenario 2	1.4	1.2	80
Damaged Scenario 3	1.4	1.5	80
Damaged Scenario 4	1.8	1.0	80
Damaged Scenario 5	1.8	1.2	80
Damaged Scenario 6	1.8	1.5	80
Damaged Scenario 7	2.3	1.0	80
Damaged Scenario 8	2.3	1.2	80
Damaged Scenario 9	2.3	1.5	80

In order to have a visual representation of the data, the irregularity values for damaged scenario 1 and the vertical acceleration for the front left wheel in the first scenario are represented in figure 4.2 and 4.3, respectively. In the figure of the irregularities, the three lines in each graph represent different ALs for different speeds. Here we are interested in the red line that represents the threshold for speeds equal to or under $80km/h$.

The remainder of the damaged scenarios and their respective standard deviations for each feature of irregularity are represented in Appendix A. In Appendix B the remainder of the accelerations for the first scenario are represented, to facilitate understanding of the data.

In summary, these datasets provide a wealth of information that encapsulates the fundamental physical attributes of the rail tracks. This rich assortment of features offers a granular view of the rail system, thereby setting the stage for a ML model that can effectively distinguish between faulty sections and non-faulty ones.

4.1.2 Matching irregularities and accelerations

As explained previously, this work uses irregularity data and acceleration data, two different measurement types. The goal is to combine these two datasets into a unified representation for further analysis. To achieve this, it is first necessary to transform the data into a time series format and apply interpolation on the smaller dataset to align their frequencies.

Time series data refers to a sequence of data points collected over time, typically at regular intervals. It captures the evolution of a variable or phenomenon over a specific period [6]. In this case, both the irregularity data and acceleration data are time-dependent, where each data point corresponds to a specific timestamp or position along a railway track.

Interpolation is a mathematical technique used to estimate values between known data points. It enables the filling of missing or irregularly spaced data with estimated values based on the available information [42]. Here, interpolation is necessary because accelerations are being collected at a higher rate than irregularities which results in incompatible datasets, even though they are both

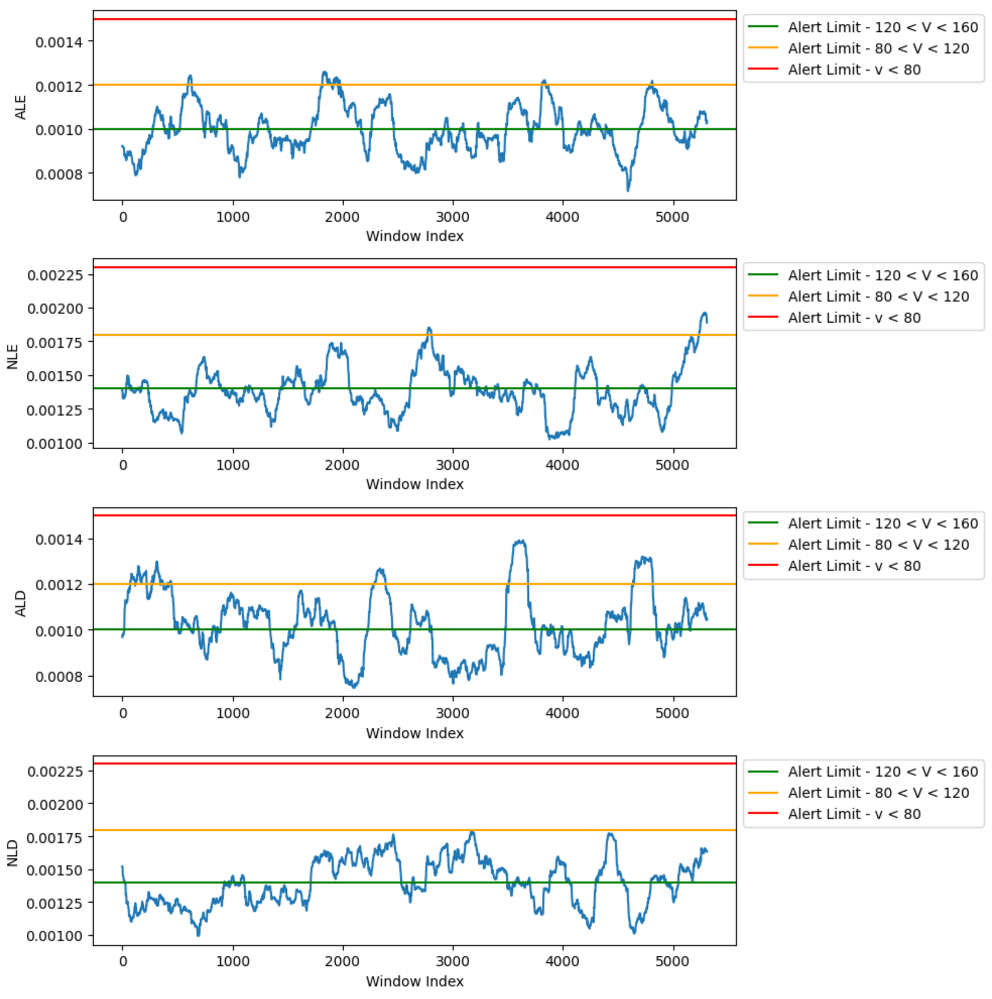


Figure 4.2: Irregularities for Damaged Scenario 1

being simulated in the same length of the railway. So, interpolation is applied to the irregularities to ensure that they have the same frequency and timestamps as the accelerations.

The reason for aligning the frequencies through interpolation is to facilitate their combination and analysis. By matching the frequencies exactly, it is ensured that each data point in the interpolated irregularity data corresponds to the same timestamp in the acceleration data. This alignment allows the creation of a single dataset (for each scenario) where each timestamp has both the irregularity and acceleration values side by side. With this combination, it is now possible to analyze and study the relationship between irregularities and accelerations along the railway track.

4.1.3 Data Labeling

Labeled data is necessary for supervised Machine Learning in order to train models that can make accurate predictions. To allow for supervised learning algorithms to figure out the underlying patterns and relationships, input data must be paired with corresponding output labels. The model can

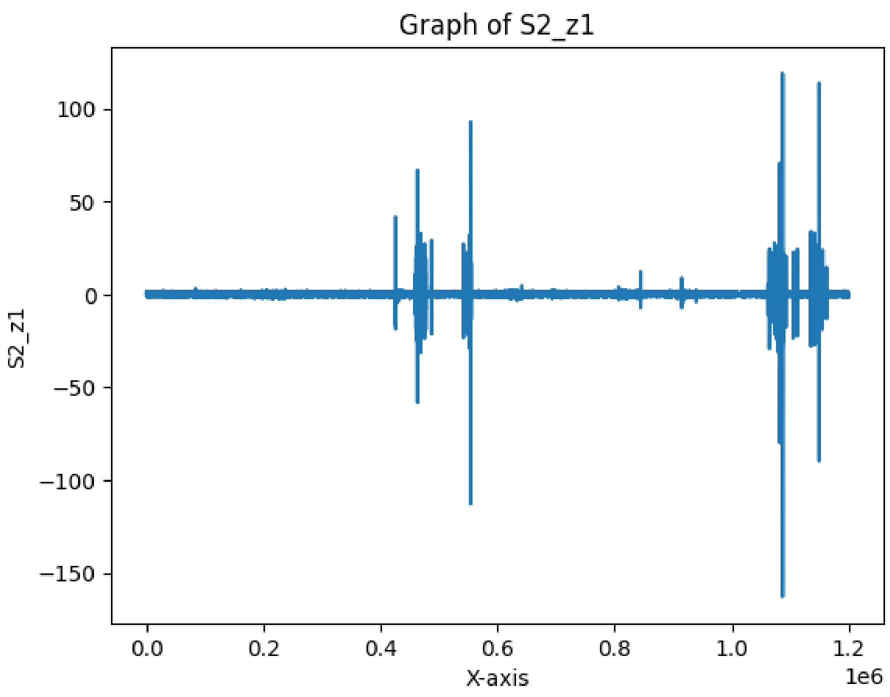


Figure 4.3: Front Left Wheel Vertical Acceleration

comprehend the relationship between input features and the desired output thanks to the reference provided by these labeled examples. Because it directly affects the model's capacity to generalize and make precise predictions on unobserved data, data labeling is of special importance.

Additionally, the quantity and quality of labeled data can significantly impact the performance of supervised learning models. Sufficient amounts of diverse and well-labeled data allow models to capture a wide range of variations and improve their generalization capabilities, hence the nine different scenarios.

As discussed earlier, here, data labeling occurs with the use of the theoretical tables 1.1 and 1.2. Simply crossing the calculated standard deviation values in 4.1.1 with the thresholds in the tables, keeping in mind that the measurements were made with the train's speed at $80km/h$, is enough to obtain all the labels. Because the irregularities and accelerations datasets were concatenated in 4.1.2, now every acceleration has a label attached to it, even though the accelerations are not directly related to the irregularities.

Table 4.2 represents the percentage of faults (label equal to 1) and non-faults (label equal to 0) present in each dataset. Bearing in mind that each dataset represents different scenarios, it is expected that the datasets are not balanced. However, concatenating all the different scenarios produces a well-balanced dataset, as observed in the last row of the table.

Table 4.2: Percentage of faults and non-faults

Damaged Scenarios	**No fault (0) [%]**	**Fault (1) [%]**
Damaged Scenario 1	99.13	0.87
Damaged Scenario 2	93.38	6.62
Damaged Scenario 3	31.32	66.68
Damaged Scenario 4	98.8	1.2
Damaged Scenario 5	91.45	8.55
Damaged Scenario 6	30.77	69.23
Damaged Scenario 7	31.9	68.1
Damaged Scenario 8	31.47	68.53
Damaged Scenario 9	14.96	85.04
All scenarios united	58.13	41.87

4.1.4 Data Scaling

Data scaling is also a crucial preprocessing step in ML models that involves transforming the data to a consistent scale. This step is essential for several ML algorithms as it improves the model's overall performance and facilitates the convergence process.

In the context of ML models, data scaling is important to address issues related to the varying magnitudes and ranges of different features. When features have significantly different scales, some with larger values might dominate the learning process and overshadow the contributions from smaller-scaled features. This can lead to biased results and inefficient convergence [10].

By scaling the data, we ensure that each feature contributes proportionately during the model's computations. Scaling brings all features to a similar scale, normalizing their values and preventing any particular feature from dominating the learning process. This promotes a more balanced consideration of all features, improving the model's ability to learn and make accurate predictions.

Furthermore, data scaling aids in regularization, which is a technique used to prevent overfitting and enhance the model's generalization ability. Regularization parameters in ML models, such as the regularization coefficient, control the trade-off between model complexity and accuracy on the training data. If the features are not scaled, regularization can be biased towards features with larger scales, leading to an imbalance in the regularization process. Scaling the features ensures that regularization is applied uniformly across all features, resulting in a more effective and fair regularization process.

Additionally, data scaling has benefits when using kernel methods in ML models. Kernel functions, such as the radial basis function (RBF) kernel, are commonly employed to capture complex relationships between data points. These kernel functions are sensitive to the magnitude of input data. If the features are not scaled, dominant features with larger scales can influence kernel computations, leading to distorted results. Scaling the data helps in maintaining the integrity of the kernel computations by equalizing the contributions of all features, regardless of their original scale.

In summary, data scaling is a vital preprocessing step in Machine Learning models, regardless of the specific algorithm used. It ensures balanced consideration of all features, improves convergence, aids in regularization, and maintains the accuracy of kernel computations. By scaling the data, we create a more robust and effective ML model that can generalize well to unseen data.

4.1.5 Feature correlation

Having all the features aligned and their respective label, it's possible to obtain the full correlation matrix between all the features and the label. Through the analysis of this matrix, several conclusions are drawn. This matrix is represented in figure 4.4.

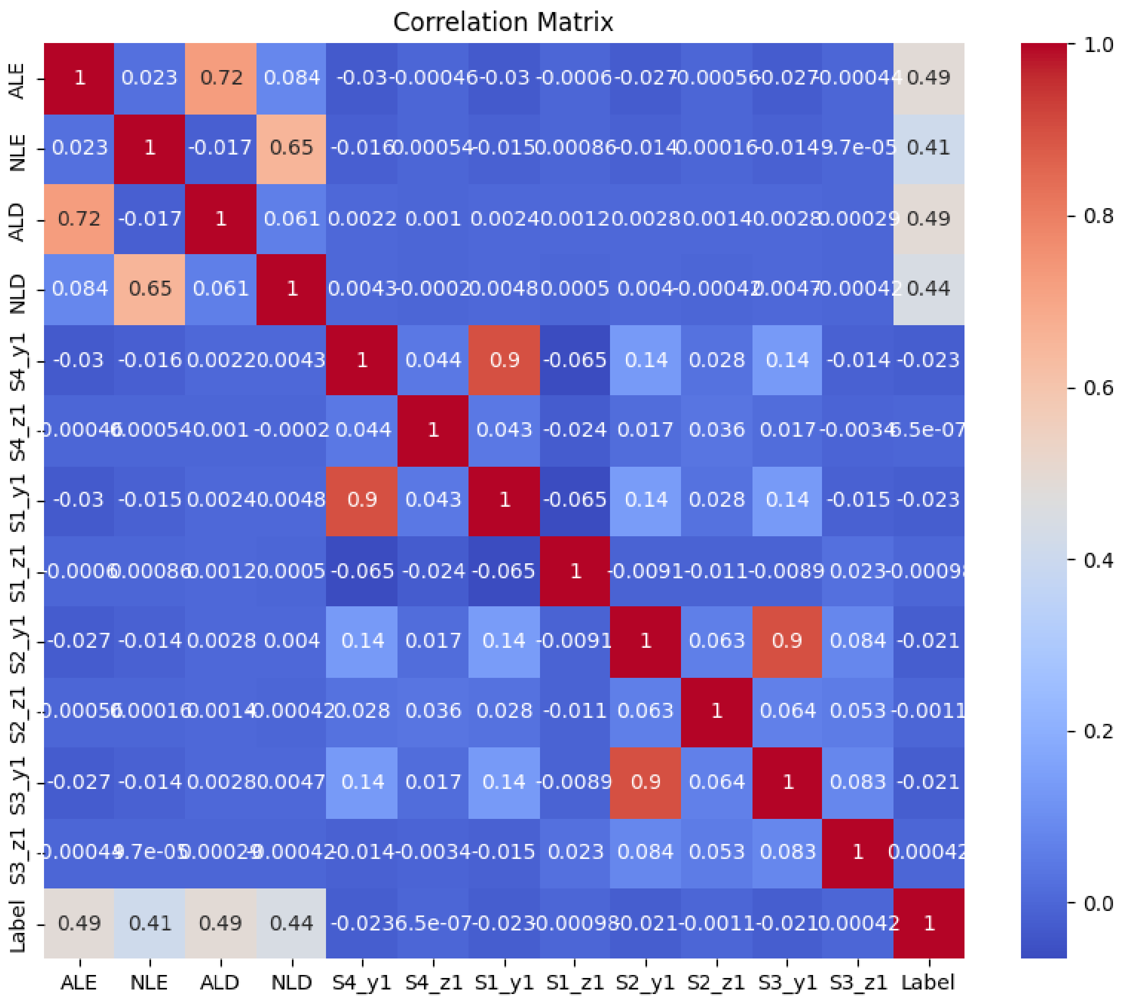


Figure 4.4: Correlation Matrix between the features and the label

The figure yields the following conclusions:

1. Values for vertical and horizontal irregularities do not show much correlation with each other. The same goes for vertical and horizontal accelerations.

2. As expected, the irregularity values represent a high correlation with the label, while acceleration values represent a very low correlation with the label. Additionally, it's also possible to see that all accelerations on the y-axis have a higher correlation than the ones on the z-axis.

3. As presumed, a very high correlation between the values of S1 and S4, and between the values of S2 and S3 is observable. As per 4.1, S1 and S4 both belong on the rear axle, and S2 and S3 to the front axle of the train.

4. Vertical acceleration values have a much smaller correlation with the label than horizontal acceleration values. This means that the vertical accelerations could be dispensable. However, since the device that captures the vertical and lateral accelerations is the same, there is no point in excluding one of these values because, in a real scenario, the cost of having the accelerometers would be the same.

5. Despite the differences outlined in the previous point, when looking at the values on the y-axis, all the accelerations show similar correlations with the label (0.021 and 0.023).

6. Overall, the correlations between the accelerations and the label are very small when compared to the irregularities. Even though this scenario was predictable, it might be an indicator that accelerations on their own will not be enough to predict the label of unseen data accurately.

4.2 Methodologies for the models

As brought up earlier, the main goal of the ML models is to classify as fault or no-fault (binary classification) every instance of the length of a given section of the rail. The predictions must all be made using only acceleration data since this data is much easier to obtain than irregularities.

This means that the provided irregularities can not be directly used to train one single model as it is not possible to use a certain amount of features i.e. irregularities plus the accelerations to train a model and after that make predictions using a reduced amount of features i.e. only the accelerations.

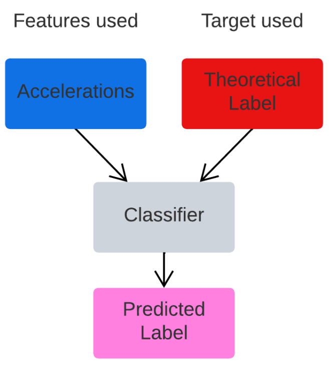
In order to bypass this issue, three methods were formulated:

1. **Classifier:** Build a single Classifier that is trained using the acceleration data and uses the labels, created using the irregularities and the respective thresholds denominated by the normative tables, as a target.

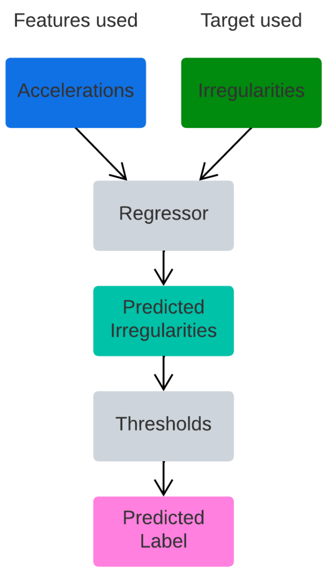
2. **Regressor:** Build a single Regressor that is trained using the acceleration data and uses the irregularities as a target (eight inputs and four outputs). Then, the thresholds denominated by the normative tables are applied and the predicted labels are obtained, using the predicted irregularities.

3. **Regressor + Classifier:** Build two models - one Regressor and one Classifier. First, the regressor is the same as before, it is trained using the acceleration data and uses the irregularities as a target. The difference to the previous technique is that a classifier is now built and trained using the predicted irregularities from the regressor concatenated with the previously used accelerations as an input and the theoretical labels as a target. This technique in which knowledge learned from a task is re-used in order to boost performance on a related task is known in ML as Transfer Learning [51].

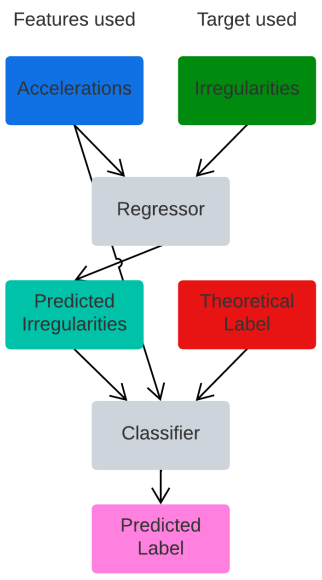
These three different methods are represented in figures 4.5a, 4.5b, and 4.5c respectively.



(a) Structure using Method 1



(b) Structure using Method 2



(c) Structure using Method 3

Figure 4.5: Comparison of different methods

4.3 Classifiers and Regressors tested

4.3.1 Support Vector Machines

Support Vector Machines represent a potent supervised learning algorithm commonly employed for classification analyses [50]. They have drawn a lot of interest because of their solid theoretical foundations and empirical successes across various domains, including fault detection.

At the heart of SVMs lies the principle of decision planes, which help create decision boundaries demarcating objects based on their class affiliations. One of the distinguishing features of SVMs is their objective to locate an optimal hyperplane that maximizes the margin, which is the distance between the decision boundary and the nearest data points, also known as support vectors [5]. The pursuit of maximum margin is integral to the SVM's design as a larger margin signifies superior generalization performance, thus enhancing the model's robustness against overfitting.

Linear SVMs, often utilized when the data is linearly separable, rely on a hyperplane to bisect the data points into distinct classes. However, real-world data is often non-linear, making linear techniques inadequate. As such, SVMs exhibit their true effectiveness by introducing kernel functions, such as the RBF, which project the data points into a higher-dimensional space. This transformation enables the creation of non-linear decision boundaries, extending SVM's applicability to non-linear problems [43].

In comparison to other classification algorithms, SVMs present several notable advantages. They possess the capability to handle high-dimensional data, an attribute vital in many real-world applications with a multitude of features. Moreover, SVMs exhibit tolerance towards noise and outliers, a trait that enhances their reliability in the presence of imperfect data [13]. Lastly, their ability to manage non-linear decision boundaries further cements their superiority as a versatile and robust ML algorithm.

In the context of this thesis, SVMs were created using the 'scikit-learn' library for Python.

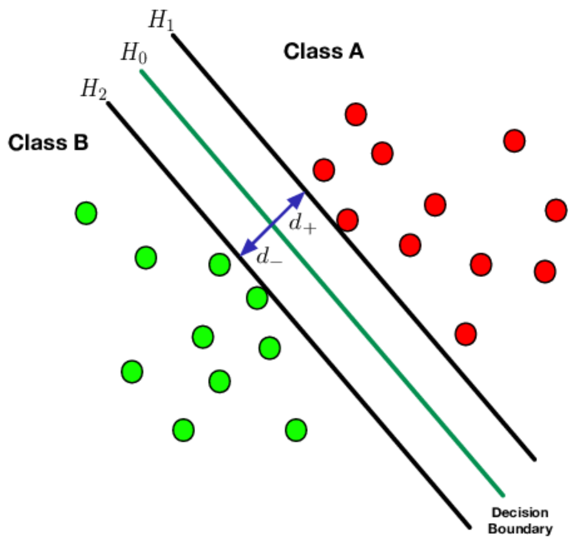


Figure 4.6: Structure of a linear SVM [3]

4.3.2 Gradient Boosting Machines

Gradient Boosting Machines (GBM) have emerged as powerful and versatile ML algorithms for both regression and classification tasks. They belong to the family of ensemble methods and are renowned for their ability to combine weak learners into a strong predictive model. In this section, we will focus on two prominent implementations of GBMs: LightGBM and XGBoost. These frameworks have gained significant popularity due to their efficiency, scalability, and superior predictive performance.

LightGBM, a gradient-boosting framework developed by Microsoft Research, stands out for its efficiency and adaptability to large-scale datasets.

LightGBM introduces a novel tree construction algorithm called Gradient-based One-Side Sampling (GOSS). This technique efficiently selects a subset of data instances with significant gradients for building trees. As a result, it greatly reduces computational costs and improves the speed of the training process [30].

Secondly, the framework adopts a histogram-based approach to store and manipulate feature values during tree construction. By discretizing continuous features into bins, LightGBM effectively reduces memory usage and enhances overall computational efficiency.

Thirdly, LightGBM supports parallel and distributed training, making it well-suited for handling large datasets that may not fit into memory. It can efficiently utilize multiple CPU cores and even scale across multiple machines, allowing for faster model training.

Lastly, LightGBM offers a range of advanced features, including automatic handling of missing values, implementation of regularization techniques like L1 and L2 regularization, and the ability to enable early stopping based on evaluation metrics. These features contribute to the framework's versatility and effectiveness in various ML tasks [30].

XGBoost, short for eXtreme Gradient Boosting, has become immensely popular in both academic and industrial circles since its inception by Tianqi Chen. It is a powerful gradient-boosting framework with a host of features and optimizations, making it stand out in terms of performance. Some of the notable aspects of XGBoost include a regularized learning objective function that balances a loss function and regularization term, providing better control over model complexity and preventing overfitting. Moreover, it employs a second-order gradient approximation, considering both first and second derivatives of the loss function, resulting in more accurate estimates of optimal model parameters [12].

XGBoost incorporates a weighted quantile sketch algorithm for approximate tree construction, significantly improving the efficiency of split search and reducing computational costs, particularly for high-dimensional datasets. The framework also offers support for various regularization techniques like L1 and L2 regularization, along with tree pruning to manage model complexity. Additionally, it provides capabilities for early stopping, cross-validation, and model interpretation.

Another advantage of XGBoost is its extensive set of APIs and interfaces, making it compatible with multiple programming languages and frameworks. This flexibility enables seamless integration into existing ML pipelines and systems.

In summary, LightGBM and XGBoost are two prominent implementations of gradient-boosting machines that have revolutionized the field of Machine Learning. Their efficient algorithms, advanced optimizations, and support for parallel processing make them well-suited for handling large-scale datasets. With their superior predictive performance and flexibility, LightGBM and XGBoost have become indispensable tools for data scientists and ML practitioners seeking to tackle complex regression and classification problems.

In the context of this thesis, LightGBM models were created using the LightGBM official library for Python, and XGBoost models were also created using the official library for Python.

4.3.3 Random Forests

Random Forests have emerged as a powerful and versatile ML algorithm that has gained significant popularity in various fields. Comprising an ensemble of decision trees, this method combines the wisdom of the crowd by leveraging the collective knowledge of multiple individual models. Random Forests exhibit remarkable flexibility, robustness, and excellent predictive performance, making them a prominent choice for both classification and regression tasks [39].

RFs operate on the principle of ensemble learning, harnessing the strengths of multiple decision trees to form a more accurate and stable prediction model. Each decision tree in the ensemble is constructed independently, allowing for diversity in the modeling process. Consequently, the combination of multiple decision trees mitigates the limitations of individual trees and collectively produces superior predictive performance.

A key feature of Random Forests is the injection of randomness during both the training and prediction phases. This randomness is introduced in two main ways: random sampling of training instances and random feature selection. During the training phase, each tree is trained on a bootstrap sample, which is created by randomly sampling the original dataset with replacement. This sampling scheme introduces diversity in the training process and ensures that each tree has its own unique perspective of the data. Moreover, at each node of the decision tree, only a random subset of features is considered for splitting. By doing so, Random Forests reduce the correlation among trees and enhance the overall diversity of the ensemble [7].

The construction of individual decision trees in a Random Forest follows the standard decision tree learning algorithm. At each node, the algorithm searches for the best split by evaluating different candidate features and splitting points based on various criteria such as information gain, Gini impurity, or mean squared error. This process is repeated recursively until a stopping criterion, such as reaching a maximum depth or a minimum number of instances per leaf, is met. The resulting decision trees form the foundation of the Random Forest ensemble.

To make predictions using a Random Forest, each decision tree in the ensemble independently casts its vote or provides a predicted value. For classification tasks, the class with the majority of votes is assigned as the final prediction. In regression tasks, the predicted values from all trees are averaged to obtain the final prediction. This aggregation mechanism allows Random Forests to capture complex patterns in the data and provide robust predictions that are less susceptible to outliers or overfitting.

Random Forests offer several advantages that contribute to their wide-ranging applicability. Firstly, they can handle high-dimensional data with a large number of features while maintaining good predictive accuracy. Additionally, they are resistant to overfitting due to the ensemble structure and the introduction of randomness. Moreover, Random Forests are capable of handling missing values and outliers effectively [7].

In the context of this thesis, Random Forests were created using the 'sckikit-learn' library for Python.

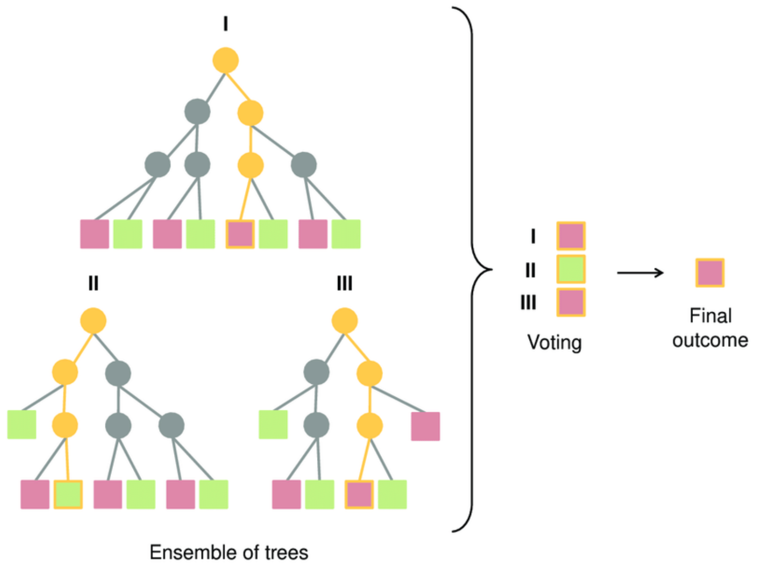


Figure 4.7: Structure of an RF [35]

4.3.4 Neural Networks

Neural networks, encompassing various architectures such as Multi-layer Perceptron (MLP), Convolutional Neural Networks, and Recurrent Neural Networks, have transformed the field of ML. They have not only proven to be highly effective in solving complex classification problems but have also demonstrated remarkable performance as regressors. This chapter explores the versatility of neural networks as both classifiers and regressors.

Artificial Neural Networks, also known as MLP classifiers, are frequently used to handle challenging classification tasks. MLP classifiers mimic the learning and prediction processes of interconnected neurons by simulating biological neural networks. The Perceptron, a synthetic neuron that gives input features weights based on their relative importance, is at the core of an MLP classifier. The Perceptron generates an output by adding the weighted inputs and using an activation function, such as sigmoid, hyperbolic tangent, or rectified linear unit (ReLU). The choice of activation function depends on the specific problem and the desired properties of the classifier [41]. The feedforward architecture teaches it how to represent input data hierarchically. Training an MLP classifier involves an iterative optimization process using the backpropagation algorithm, which adjusts the perceptron's weights based on prediction errors. Regularization techniques, such as weight decay or dropout, help mitigate overfitting issues and enhance generalization. MLP classifiers excel at capturing non-linear relationships in high-dimensional datasets, accommodating various types of input features, including numerical and categorical variables [24].

In addition to classification, Neural Networks have proven to be powerful regressors. Deep learning techniques, including MLP architectures, CNNs, and RNNs, have gained prominence in regression problems. Deep learning refers to training neural networks with multiple hidden layers, enabling them to learn intricate patterns and representations from complex data. MLP architectures are widely used for regression, capturing non-linear relationships between input features and target outputs.

RNNs excel in capturing temporal dependencies and sequences, making them suitable for regression tasks involving time-series data. RNNs are proficient at modeling sequences and using historical data to forecast future values. The vanishing gradient problem and the need to capture long-term dependencies in regression tasks led to the development of Long Short-Term Memory. Sequence-to-sequence regression is a function that RNNs can perform when the input and output are both sequences. RNNs can map input sequences to output sequences in a regression context thanks to encoder-decoder architectures [55].

When dealing with multi-target regression problems such as the case at hand, it is a must that the Regressor can predict multiple outputs simultaneously. The task of predicting multiple outputs requires Regressors capable of handling this complexity. Unfortunately, in such cases, the offer of ML models becomes very limited as Neural Networks and Random Forests are the best options that support multi-output by default [16].

In the context of this thesis, both ANNs and RNNs were created using the 'Keras' API, built on top of 'TensorFlow', for Python.

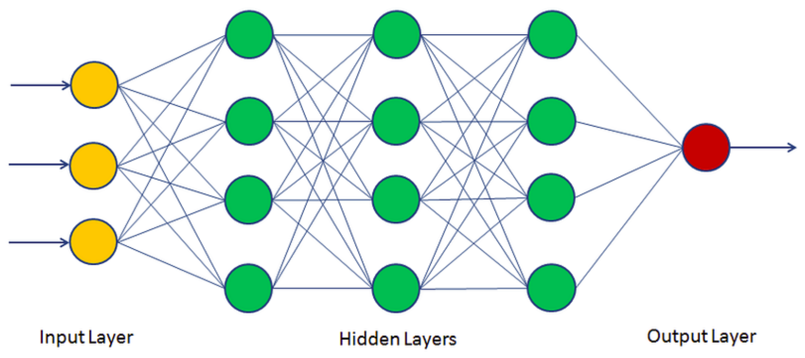


Figure 4.8: Structure of an ANN [38]

4.4 Model Evaluation and Performance Metrics

In this section, we will discuss the evaluation of the Classifiers and Regressors developed for fault identification in rails using AI. We will focus on important performance metrics and the confusion matrix for classifiers and RMSE, R-squared, and MAE for regressors.

4.4.1 Classifier Evaluation

The evaluation of Classifiers is essential to determine their effectiveness in accurately identifying faults in the rails [27]. The confusion matrix, as shown in table 4.3, is a powerful tool for understanding the classifier's performance.

The key performance metrics used for classifier evaluation are:

Table 4.3: Confusion Matrix

		Actual Class	
		Positive	**Negative**
Predicted Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

- **Accuracy**: The overall correctness of the classifier's predictions, calculated as the ratio of the sum of true positives and true negatives to the total number of instances. It is computed as Accuracy $= \frac{\text{True Positives (TP)}+\text{True Negatives (TN)}}{\text{True Positives (TP)}+\text{False Positives (FP)}+\text{True Negatives (TN)}+\text{False Negatives (FN)}}$

- **Recall (True Positive Rate)**: The proportion of actual fault instances that the classifier correctly identified, computed as Recall $= \frac{\text{True Positives (TP)}}{\text{True Positives (TP)}+\text{False Negatives (FN)}}$. In the context of fault identification in rails, recall is of paramount importance as it ensures that most actual faults are captured, minimizing the risk of missing critical issues.

- **Precision**: The proportion of predicted fault instances that are correct, given by Precision $= \frac{\text{True Positives (TP)}}{\text{True Positives (TP)}+\text{False Positives (FP)}}$. A high precision score indicates that the model is accurately identifying actual faults with minimal false positives.

- **Specificity (True Negative Rate)**: The proportion of correctly identified non-fault instances out of all actual non-fault instances, given by Specificity $= \frac{\text{True Negatives (TN)}}{\text{True Negatives (TN)}+\text{False Positives (FP)}}$. In the context of fault identification, a high specificity is desirable to minimize the risk of false alarms and unnecessary maintenance actions.

4.4.2 Regressor Evaluation

The evaluation of regressors is crucial to assess their ability to predict numerical values related to the irregularities accurately [14]. The key metrics used for regressor evaluation are:

- **RMSE (Root Mean Squared Error)**: Measures the average squared difference between the predicted and actual values. A lower RMSE indicates a better fit of the model.

- **R-squared (Coefficient of Determination)**: Quantifies the proportion of the variance in the dependent variable (output) that is predictable from the independent variables (input). A value close to 1 suggests that the model can explain a large portion of the variability.

- **MAE (Mean Absolute Error)**: Measures the average absolute difference between the predicted and actual numerical values. It provides an easily interpretable metric of prediction accuracy.

Chapter 5

Results and discussion

In this chapter, all the results obtained are presented.[1] Diverse models were created and tested. This chapter will give a clear understanding of the models and methods that work correctly and also what deserves further research and enhancement.

5.1 Results

This section is divided into four subsections, one per each different method tested and one subsection regarding some early testing.

5.1.1 Early Considerations

Before creating concrete models some tests were carried out regarding only small sections of all the concatenated datasets, in order to have an idea of the computational requirements of different models and methods. All of the tests that follow were made using Google Colab Premium. Google Colab (short for Google Colaboratory) is a cloud-based Jupyter Notebook environment offered by Google. It provides a platform for researchers, developers, and students to write, execute, and share code, especially in the domain of ML. One of its primary benefits for Machine Learning is that it offers access to powerful hardware resources without requiring users to set up their own local computing infrastructure. Colab allows users to run code on Google's high-end GPUs (Graphic Processing Units) and TPUs (Tensor Processing Units), significantly accelerating the training of machine learning models. Additionally, Colab comes pre-installed with popular ML libraries like TensorFlow, Keras, and PyTorch, streamlining the process of building and experimenting with models.

Table 5.1 has a simple representation of most of the models created and their difference in run times. In this first estimate, the support vector machines had "RBF" kernels, the RFs, LightGBMs and XGBoosts all had 10 estimators, and the neural networks had 2 hidden layers iterated over 10 epochs.

[1] All of the results are available in the following Google Spread Sheet for consultation: `https://docs.google.com/spreadsheets/d/19FRU3COVT8NsxSN1q8XVs6MeEeRZuqHTaMkrAd647n4/edit?usp=sharing`

Table 5.1: Comparison of models run-time

AI Models	Sample Size [%]	Run-Time
Support Vector Machine	1	8:00
Support Vector Machine	5	1:21:00
Random Forest	1	0:10
Random Forest	5	4:14
Random Forest	25	6:39
XGBoost	1	0:10
XGBoost	5	0:58
XGBoost	25	4:02
LightGBM	1	0:02
LightGBM	5	0:05
LightGBM	25	0:25
LightGBM	100	1:38
Neural Network	1	3:00
Neural Network	25	13:00

By looking at table 5.1 it is clear that LightGBM is by far the fastest model of the bunch. In terms of computational requirements, it is a lot faster than every other model tested. It is also clear that Support Vector Machines might not be the best option as it takes 1 hour and 20 minutes to run on just 5% of the data. This means that if the SVM was ever to be run on 100% of the data, it would take a really long time to train. Random Forests, XGBoost, and Neural Networks are more computationally expensive than LightGBM. However, they all run in very respectable times and will undergo further testing.

As discussed in Section 4.4, the performance metrics of utmost importance in this specific context are accuracy, recall, and specificity. Accuracy provides a well-rounded assessment of the overall model performance, making it a vital metric. Equally significant is recall, as it ensures the effective identification of actual faults, reducing the risk of overlooking critical issues. However, specificity, representing the True Negative Rate, also holds great importance in minimizing false alarms. Striking a fine balance between the True Positive Rate and the true negative rate becomes the ultimate goal in this scenario. For instance, a model may exhibit high accuracy and True Negative Rate but suffer from low recall. This leads the model to classify most unseen data as non-fault, even when faults are present, undermining its primary objective of accurately predicting fault occurrences.

5.1.2 Method 1: Classifier

Method 1 consists of building a Classifier that takes the acceleration values as inputs and is trained using the theoretical labels as a target. These Classifiers output directly the predicted label (0/1) for unseen data using only the eight acceleration values in a given point, as per 4.2.

The following results were all obtained by concatenating all the datasets into one and making a train-test split of 90% and 10%. The first classifier tested was the Support Vector Machine with an RBF kernel. Its results are represented in table 5.2.

Table 5.2: Results for Method 1 - SVM

Sample Size [%]	**Accuracy** [%]	**Precision** [%]	**Recall (TPR)** [%]	**Specificity (TNR)** [%]
1	84.71	78.36	15.62	99.07
5	69.87	74.46	17.94	96.81

SVM requires a lot of computational power and its run times are significantly larger than other models. Its results also show that they provide a high imbalance between TPR and TNR, meaning that they can't distinguish well between the two labels and the extra computational requirements that SVM requires are not worth it.

The second and third classifiers tested were the GBMs, LightGBM, and XGBoost. Its results are represented in tables 5.3 and 5.4

Table 5.3: Results for Method 1 - LightGBm

Sample Size [%]	**Accuracy** [%]	**Precision** [%]	**Recall (TPR)** [%]	**Specificity (TNR)** [%]
1	78.13	70.15	20.73	97.09
5	69.36	64.11	23.41	93.20
25	61.63	64.69	32.66	86.29
100	61.2	64.76	37.44	83.62

Table 5.4: Results for Method 1 - XGBoost

Sample Size [%]	**Accuracy** [%]	**Precision** [%]	**Recall (TPR)** [%]	**Specificity (TNR)** [%]
1	78.16	65.69	25.11	95.67
5	69.64	63.93	25.51	92.53
25	62.8	65.02	32.21	87.14
100	62.26	65.14	37.17	84.45

The gradient-boosting machines showed an improvement in the balance between recall and specificity. However, the recall value is still very low when compared to the specificity.

Next, Random Forests with 100 estimators were tested. Their results are shown in table 5.5

Table 5.5: Results for Method 1 - RF

Sample Size [%]	**Accuracy** [%]	**Precision** [%]	**Recall (TPR)** [%]	**Specificity (TNR)** [%]
1	78.71	71.43	23.72	96.87
5	70.14	63.83	29.01	91.47
25	65.34	61.82	36.48	85.70
100	65.36	61.89	39.48	83.22

Random Forests performed slightly better than the gradient boosting machines, but have not presented good enough results. Additionally, tests were also made using 50 and 200 estimators. However, no major improvements happened.

Lastly, Artificial Neural Networks were also built. Different tests were executed, trying different numbers of hidden layers, different activation functions, and increasing and decreasing the number of epochs and the learning rate. The best and most balanced result for a neural network used ten epochs, a learning rate of 0.001 three fully connected hidden layers with ReLU activation functions, and 64 neurons each. The model also has an Adaptive Moment Estimation (Adam) optimizer. The last layer of the neural network had two neurons with a softmax activation function to solve the classification problem. The results are represented in table 5.6.

Table 5.6: Results for Method 1 - ANN

Sample Size [%]	**Accuracy** [%]	**Precision** [%]	**Recall (TPR)** [%]	**Specificity (TNR)** [%]
1	79.01	67.95	29.23	95.45
5	70	65.85	25.29	93.20
25	65.33	62.38	35.03	85.76
100	65.13	60.74	41.30	81.58

Tests were also made using a Recurrent Neural Network with three LSTM layers to find if it performs better. The results are in table 5.7.

Table 5.7: Results for Method 1 - RNN

Sample Size [%]	**Accuracy** [%]	**Precision** [%]	**Recall (TPR)** [%]	**Specificity (TNR)** [%]
5	69.64	63.23	26.58	91.98
100	65.12	61.17	42.62	80.96

Observing tables 5.6 and 5.7 it's possible to conclude that these neural networks performed on the same level as Random Forests. RNNs performed just a bit better but they add a lot of complexity and computational requirements so they do not compensate for the slightly better performance.

Overall, most classifiers suffered from the issue of having low recall percentages. Although some have respectable values of accuracy, low recall means that the models are not classifying accurately enough the positive instances. This outcome was forecast in section 4.1.5. Random Forests, Artificial Neural Networks, and Recurrent Neural Networks showed the most promise while gradient-boosting machines were not far behind while having the benefit of being a lot faster. SVMs performed poorly and are very computationally expensive so they are not the right answer for this method.

5.1.3 Method 2: Regressor

In this second method, as mentioned before, the goal is to predict the values of the irregularities using the values of the accelerations as the starting point. Then, the predicted irregularity values are checked against the normative tables, and the labels are obtained.

As highlighted previously, the choice of regressors, in this case, is very limited since only Random Forests and Neural Networks possess the ability to predict multiple outputs by default. Regressors such as Gradient Boosting Regressors or Support Vector Regressors do not natively support multi-output regression.

For the following tests, the RF Regressor had 50 estimators and 30 estimators for the sample size of 100% (because of RAM limitations). The Neural Networks have an input layer with eight neurons (one for each of the acceleration features) and an output layer with four neurons (one for each of the irregularity values). The model also has three fully connected hidden layers with a total of 64 neurons each, with ReLU activation functions. The learning rate was set to 0.001 and an Adam optimizer was also used.

Table 5.8 shows the evaluation metrics for the regressors built.

Table 5.8: Evaluation of the Regressors

Model	**Sample Size [%]**	**RMSE**	**R-Squared**	**MAE**
RF	1	0.3319	0.8897	0.2303
RF	5	0.36199	0.8689	0.252
RF	25	0.365	0.8667	0.2698
RF	100	0.3731	0.8607	0.2856
NN	1	0.8077	0.3471	0.5413
NN	5	0.9228	0.1478	0.6441
NN	25	0.9502	0.0969	0.7186
NN	100	0.9476	0.1018	0.7465

The results of the previous table clearly show that Random Forests outperform neural networks in this specific task. RFs have roughly three times better values for the RMSE and the MAE. Additionally, they have a really high value of R-squared while NNs perform really poorly in the R-squared metric. This means that RFs can explain a large portion of the variability of the data while NNs may struggle.

The results for the Regressor plus classification with the thresholds are shown in table 5.9.

Table 5.9: Results for Method 2

Regressor Model	**Sample Size [%]**	**Accuracy [%]**	**Precision [%]**	**Recall (TPR) [%]**	**Specificity (TNR) [%]**
RF	1	26.37	25.09	99.02	2.38
RF	5	46.25	37.39	85.07	26.11
RF	25	51.01	44.26	83.38	29.19
RF	100	53.07	45.81	81.68	33.33
NN	1	24.82	24.82	100	0
NN	5	54.55	40.8	73.25	44.85
NN	25	55.01	46.44	76.52	40.50
NN	100	59.39	50.19	70.16	51.95

Observing table 5.9, although the overall accuracy of the models obtained was a bit smaller when compared to the results of Method 1, Method 2 exhibits high recall for both RF and NN regressors, which is a notable strength. This shows that the method is effective in identifying a substantial amount of real faults. In safety-critical applications like identifying railway faults, a high recall is essential because it makes sure that the majority of actual faults are found, reducing the chance of missing critical issues. The analysis also reveals that the method's performance is influenced by the dataset size, as expected. Accuracy and specificity typically get better as the sample size grows. With larger sample sizes, the recall does, however, slightly decline, which is a trade-off to take into account. Throughout the evaluations, RF consistently outperforms NN as the regressor in the recall metric. However, NN exhibits higher precision, accuracy, and specificity.

5.1.4 Method 3: Regressor + Classifier

Method 2 showed an improvement over Method 1 which shows that the use of a Regressor helped increase performance. However, a simple Regressor was susceptible to small oscillations in the values of the predicted irregularities. As such, employing a Classifier that learns the classification using the predicted irregularities could make the overall model perform better. With this method, tests were made using three different sample sizes. The tables below show the results from all these tests and conclusions are drawn after. The Regressors are the same as used in Method 2 and the classifiers follow the same architecture as the one used in Method 1.

Table 5.10: Results for Method 3 - 5% sample size

Regressor Model	**Classifier Model**	**Accuracy [%]**	**Precision [%]**	**Recall (TPR) [%]**	**Specificity (TNR) [%]**
RF	LightGBM	47.28	35.13	64.21	38.50
RF	XGBoost	46.31	35	66.72	35.72
RF	NN	45.54	34.56	66.52	34.66
RF	RF	46.96	35.02	64.58	37.82
NN	LightGBM	54.42	34.47	37.1	63.41
NN	XGBoost	54.06	35.89	43.85	59.36
NN	NN	57.42	38.71	42.28	65.27
NN	RF	51.41	35.33	50.86	51.70

From tables 5.10, 5.11 and 5.12 it's possible to take out some conclusions:

1. **Impact of Sample Size on Performance:** As expected, it's possible to see that the overall performance metrics tend to get better as the sample size rises from 5% to 100%. Only recall doesn't quite follow this trend. Larger sample sizes give the models access to more data, which improves generalization and performance. However, the improvements are not substantial, suggesting that the model might be approaching a level where performance does not improve much, even with more data.

2. **Regressor Performance:** Across all sample sizes, the Regressor models generally show similar performance when used in combination with different classifiers. There is no clear

Table 5.11: Results for Method 3 - 25% sample size

Regressor Model	**Classifier Model**	**Accuracy [%]**	**Precision [%]**	**Recall (TPR) [%]**	**Specificity (TNR) [%]**
RF	LightGBM	51.37	42.53	59.13	46.13
RF	XGBoost	51.14	42.35	58.98	45.86
RF	NN	49.58	41.62	62.61	40.79
RF	RF	50.85	42.19	59.53	45.00
NN	LightGBM	52.09	42.15	50.93	52.87
NN	XGBoost	50.77	42.13	59.6	44.81
NN	NN	52.21	42.7	54.58	50.62
NN	RF	49.94	41.64	60.45	42.86

Table 5.12: Results for Method 3 - 100% sample size

Regressor Model	**Classifier Model**	**Accuracy [%]**	**Precision [%]**	**Recall (TPR) [%]**	**Specificity (TNR) [%]**
RF	LightGBM	54.66	44.43	44.02	62
RF	XGBoost	54.15	43.94	44.52	60.80
RF	NN	53.34	43	43.84	59.90
RF	RF	54.2	44.04	44.91	60.62
NN	LightGBM	52.69	43.04	49.07	55.18
NN	XGBoost	52.25	42.65	49.18	54.36
NN	NN	53.13	43.35	48.22	56.51
NN	RF	49.93	41.92	58.72	43.86

indication that one Regressor consistently outperforms the other in terms of accuracy, precision, recall, or specificity.

3. **Classifier Performance:** The Classifier models also demonstrate comparable performance across different Regressors and sample sizes. Again, no single Classifier emerges as the clear winner across all metrics and sample sizes.

4. **Accuracy:** The accuracy values are generally moderate, ranging from around 45% to 57% for 5% sample size, 49% to 52% for 25% sample size, and 50% to 55% for 100% sample size. These values indicate that the models are performing better than random guessing, but there is room for improvement.

5. **Precision:** Precision measures the proportion of true positives among predicted positives (faults). In most cases, the precision values are relatively low, ranging from 34% to 39% for 5% sample size, 41% to 43% for 25% sample size, and 42% to 44% for 100% sample size. This means that when the models predict a fault, they often include false positives.

6. **Recall (True Positive Rate, TPR):** Recall values, representing the ability to capture actual faults, vary significantly. For 5% sample size, the recall ranges from around 37% to 67%, for 25% sample size, it ranges from 51% to 62%, and for 100% sample size, it ranges from

43% to 59%. This indicates that the models have difficulty capturing all the true faults, and there is room for improvement in this area.

7. **Specificity (True Negative Rate, TNR):** Specificity values show the ability to correctly identify non-faults. Across models and sample sizes, specificity oscillates a lot, ranging from around 35% to 65% for 5% sample size, 40% to 53% for 25% sample size, and 43% to 62% for 100% sample size. This suggests that the models perform relatively better in identifying non-fault cases when there is access to more data.

8. **Regressor and Classifier Pairing:** Sometimes using the same model for the Regressor and the Classifier (i.e. RF Regressor with RF Classifier or NN Regressor with NN Classifier) improves performance just a little bit. This implies that when the two models are the same, they are probably learning complementary representations. In some situations, however, using a different Regressor and Classifier model can result in better performance (for example, RF Regressor with LightGBM Classifier, NN Regressor with XGBoost Classifier). This indicates that different models might be capturing different aspects of the data that are beneficial for the final classification.

9. **Balance Between Specificity and Recall:** As previously mentioned, in the context of this problem, both specificity and recall are crucial metrics. According to the results, there appears to be a trade-off between recall and specificity. Recall tends to be lower when specificity is high, and vice versa. It's critical to strike a balance between these two metrics to minimize both the risk of false alarms and unnecessary maintenance actions and the risk of missing critical issues. Focusing on models that achieve a good balance between these metrics would be advantageous given the significance of recall and specificity in identifying railway faults. For instance, LightGBM and XGBoost classifiers seem to perform well in this regard when paired with the RF Regressor.

Following all these considerations, although not a clear winner, a conclusion can be made that the pairing of a Random Forest Regressor and a GBM Classifier shows the most promise for real-world applications. Another reason for this conclusion includes the fact that RF Regressors are much less prone to overfitting when compared to Neural Networks. Additionally, RF regressors are known for their interpretability compared to some other complex models like Neural Networks. Combined with GBMs, which also provide some level of interpretability, this pairing might be easier to understand and explain which could be a desirable aspect, especially in safety-critical applications like railway fault identification. Another important aspect is efficiency. RF and GBMs are both efficient algorithms, with LightGBM being particularly notable, especially when compared to more complex models like Neural Networks. This efficiency can be advantageous when dealing with large datasets or deploying the model in real-time environments. Finally, the RF regressor paired with LightGBM, for example, has a very good balance between different performance metrics, making it a practical option for real-world deployment. While other combinations might excel in specific metrics, the overall balance and consistency of this pairing could make it a more versatile and promising choice.

5.1.5 Additional Tests

For the purpose of extracting more conclusions, some additional tests were made. For this test, instead of concatenating all the datasets into one, two splits were made. The first split includes datasets 1, 2, 4, and 5 and the second split consists of datasets 3, 6, 7, and 8. The first split covers datasets predominantly non-faulty, i.e. datasets high a very low number of faults while the second split covers the datasets predominantly faulty, i.e. datasets with a higher number of faults. These tests were made for both Method 2 and Method 3 and for the sample size of 25%. The results are represented in tables 5.13 and 5.14. The results of Method 2 are in the row where the classifier used is the threshold and the results of Method 3 are the remaining rows.

Table 5.13: Results for the first split

Regressor Model	**Classifier Model**	**Accuracy [%]**	**Precision [%]**	**Recall (TPR) [%]**	**Specificity (TNR) [%]**
RF	Thresholds	14.1	5.28	90.14	9.86
RF	LightGBM	87.75	2.12	2.92	92.48
RF	XGBoost	87.83	2.11	2.88	92.56
RF	NN	85.86	2.48	4.38	90.40
RF	RF	87.86	2.16	2.94	92.59
NN	Thresholds	17.55	5.31	86.94	13.69
NN	LightGBM	94.72	0	0	100
NN	XGBoost	94.72	0	0	100
NN	NN	94.42	2.23	0.13	99.68
NN	RF	94.72	0	0	100

Table 5.14: Results for the second split

Regressor Model	**Classifier Model**	**Accuracy [%]**	**Precision [%]**	**Recall (TPR) [%]**	**Specificity (TNR) [%]**
RF	Thresholds	66.19	68.75	92.69	8.88
RF	LightGBM	59.71	68.39	76.39	23.63
RF	XGBoost	60.34	68.37	87.42	21.79
RF	NN	63.77	68.39	78.16	12.61
RF	RF	60.39	68.39	78.23	21.83
NN	Thresholds	66.11	68.94	91.79	10.58
NN	LightGBM	67.54	68.38	97.73	2.25
NN	XGBoost	66.97	68.29	96.5	3.10
NN	NN	65.29	68.26	92.02	7.49
NN	RF	64.16	68.44	88.3	11.94

The results for the first split, that have a very low number of faults, clearly show a bad performance, especially using the NN Regressor. The results between Method 2 and 3 are the completely opposite, only showing similar results in the precision metric. Method 2 shows a really high TPR and low performance on the remaining metrics while Method 3 shows really high accuracy and TNR. From these results, it's possible to conclude that neither method works properly displaying a huge disparity between metrics.

The outcomes for the second split, which have a higher number of faults but have more balanced distributions between classes, indicate better performance. The results are much more balanced than the ones on the first split, exhibiting moderate accuracy and precision, high recall, and low specificity. Here, the results between Method 2 and 3 are very similar, unlike what was achieved using the first split. However, it is still evident a high unbalance between the TPR and the TNR.

From these results, it's possible to conclude that the models perform a lot better in an environment where there are a lot of faults. The results from the first split could have also been interfered with by the fact that these datasets are much more unbalanced than those of split 2. This difference between the splits can be observed in table 4.2. However, this discrepancy between classes in the first split can be seen as a good thing because, in a real-world scenario, the most normal case is to find that a given railway has a percentage of faulty sections under 1%. The models could work better in the first split if the distribution of the classes was similar to the second split, but the testing data would not be simulating a real-world scenario.

5.2 Discussion on the effectiveness and limitations of the proposed system

Starting with Method 1, the results were very poor, indicating difficulty in classifying faults (low recall). As previously forecast in section 4.1.5, the accelerations showed a very small correlation with the theoretical labels, meaning that using the accelerations by themselves to directly predict new labels of unseen data did not seem feasible. Such a situation has been proven by the results of Method 1.

Moving on to Method 2, the results demonstrate some strengths in achieving high recall for both RF and NN Regressors. This indicates that Method 2 is fairly good at capturing a significant proportion of actual faults, minimizing the risk of missing critical issues. However, the main limitation of Method 2 is the relatively low specificity. This means that the method might be prone to generating more false alarms and unnecessary maintenance actions, which is a concern.

Lastly, Method 3 which used a Transfer Learning approach with Regressors feeding into Classifiers fell short of expectations. The results showed medium accuracy, precision, recall, and specificity. Although the results were not particularly exciting, they did show a more balanced ratio between the TPR and the TNR, which is a good indicator. This could mean that further research and improvement on the models can make it succeed in its task. Especially, the pairings of RF Regressor with LightGBM or XGBoost Classifiers exhibit competitive results, striking a good balance between the TPR and the TNR. This ensures that a considerable number of actual faults are captured while minimizing false alarms.

In spite of all these considerations, the overall performance of Method 2 can be considered better than the performance of Method 3.

Chapter 6

Conclusions and Future Work

6.1 Summary of the research findings

In this section, a summary of the key research findings derived from the extensive investigation and analysis conducted in this thesis is presented. The following discoveries represent the fundamental outcomes that address the research objectives established at the outset of this study.

The main research objective aimed to build a working AI model that had the ability to predict railway faults based solely on Acceleration data. For this purpose, three different methods were developed and evaluated. One of the most significant findings was that Acceleration data is not enough to directly classify and predict faults, as observed in the results section of the first Method developed. This discovery meant that other ways of predicting the faults had to be implemented. So, two more Methods were developed, that used the Accelerations to predict Irregularities which is an indicator that is directly correlated with the existence of faults in the rails. These were called Method 2 and Method 3.

Both Method 2 and 3 presented strengths and limitations. Each of these two methods offers unique advantages, and their choice should be carefully considered based on the specific application's requirements. Method 2's strengths lie in high recall and interpretability but may suffer from lower specificity, leading to more false positives. Method 3 demonstrates balanced performance with moderate accuracy.

In conclusion, Method 2 emerged as the most effective among the proposed methods due to its high recall rate and proficient fault capture. However, it is essential to consider the trade-off between recall and specificity in practical applications. Depending on the specific requirements and tolerances of the system, either Method 2 or Method 3 with potential for improvement may be the preferred choice.

6.2 Limitations of the study and recommendations for future research

In this study, some limitations were encountered that impacted the scope and outcomes of the research. Lack of computing power was a considerable restriction, especially when working with 100% of data. The use of Google Colab for building the models sometimes resulted in prolonged processing times and insufficient RAM even though programs ran with access to a maximum of 35.2 GB of RAM. For example, using 100% of the data, it was only possible to create RF Regressors with 30 estimators as using more than that would consume all the RAM available and shut down the program. These reasons prevented thorough testing of methods like k-fold cross-validation which could have increased some parameters on the models. This method, which divides the data into smaller groups for iterative training and testing, might have produced more accurate assessments and enhanced model performance. Even with the use of accelerating hardware like the GPUs Nvidia A100, Nvidia V100 or Nvidia T4 in Colab, sometimes the computing power was an issue.

The study's sole reliance on simulated data, and lack of any real-world data, was an additional limitation. At first, real-world data was expected to be included, but obtaining it was not possible for the time being. Since simulations can't completely replicate the complexities and variations present in real-world railway conditions, the models may not have performed as well as they could have.

Despite these limitations, the study identified promising avenues for future research. One such approach was the utilization of Regressors to predict the Irregularity indicators (Method 2 and 3). Although the results were not optimal, fine-tuning and further improvements to these techniques could enhance the model's performance. Exploring alternative architectures or pre-trained models may produce better outcomes. Additionally, techniques to reduce overfitting in the NN models like Regularization or Dropout and the already mentioned k-fold cross-validation could all improve performances.

Future research should also incorporate real-world data to increase the AI models' applicability in the real world. The models can develop a more thorough understanding of rail fault prediction, capturing the complexity of real operational conditions, by combining simulated data with actual railway data. In order to improve the research, there are also opportunities to include different variables, approaches, and data sources. Incorporating weather information, such as temperature, humidity, and precipitation, for example, could reveal information about how these factors affect rail faults. To validate the findings, field studies and long-term data collection on operational railway tracks are crucial. The accuracy and efficiency of the AI models should be evaluated by deploying them in real-world railway systems and continuously gathering data there.

Addressing these limitations and pursuing these recommendations can significantly contribute to enhancing the performance and practicality of AI-driven railway fault prediction systems in the future.

Appendix A

Graphs for Irregularities

A.1 Irregularities for Damaged Scenario 2

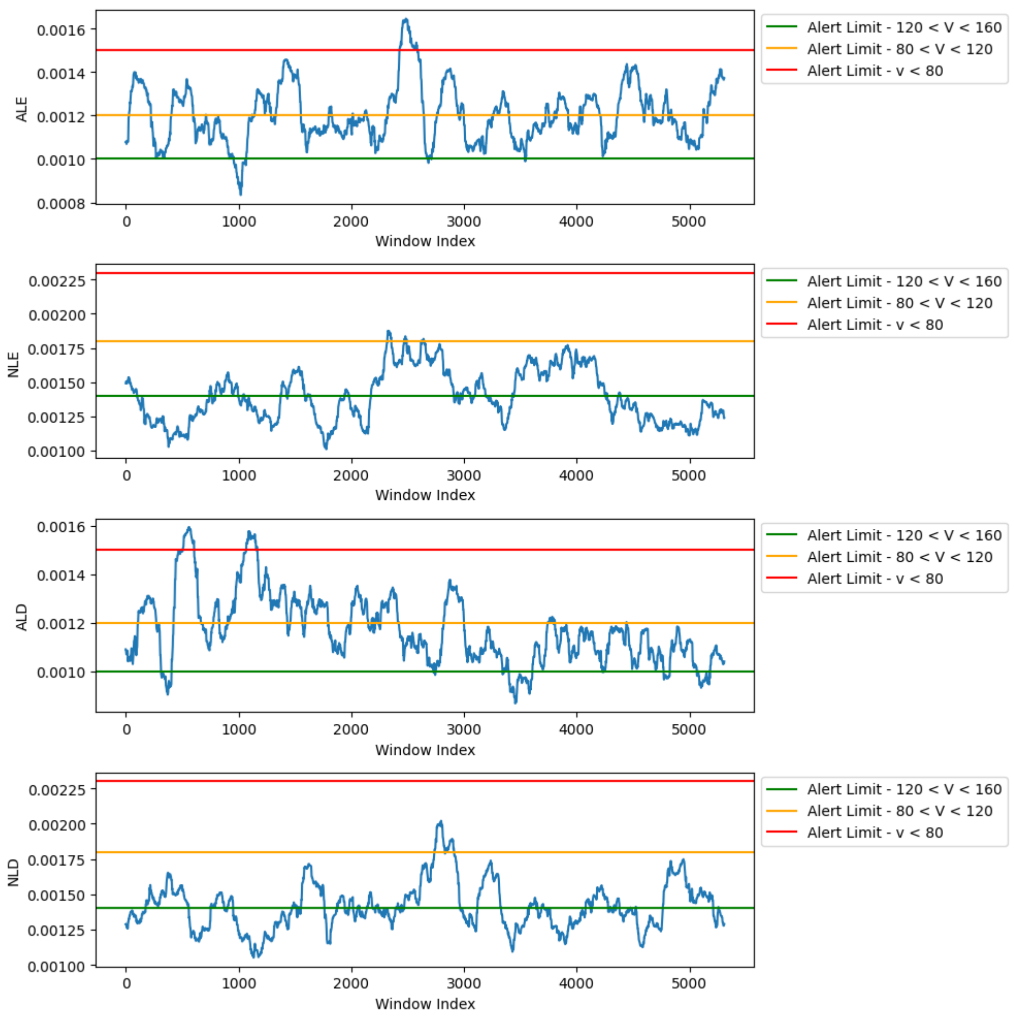


Figure A.1: Irregularities for Damaged Scenario 2

A.2 Irregularities for Damaged Scenario 3

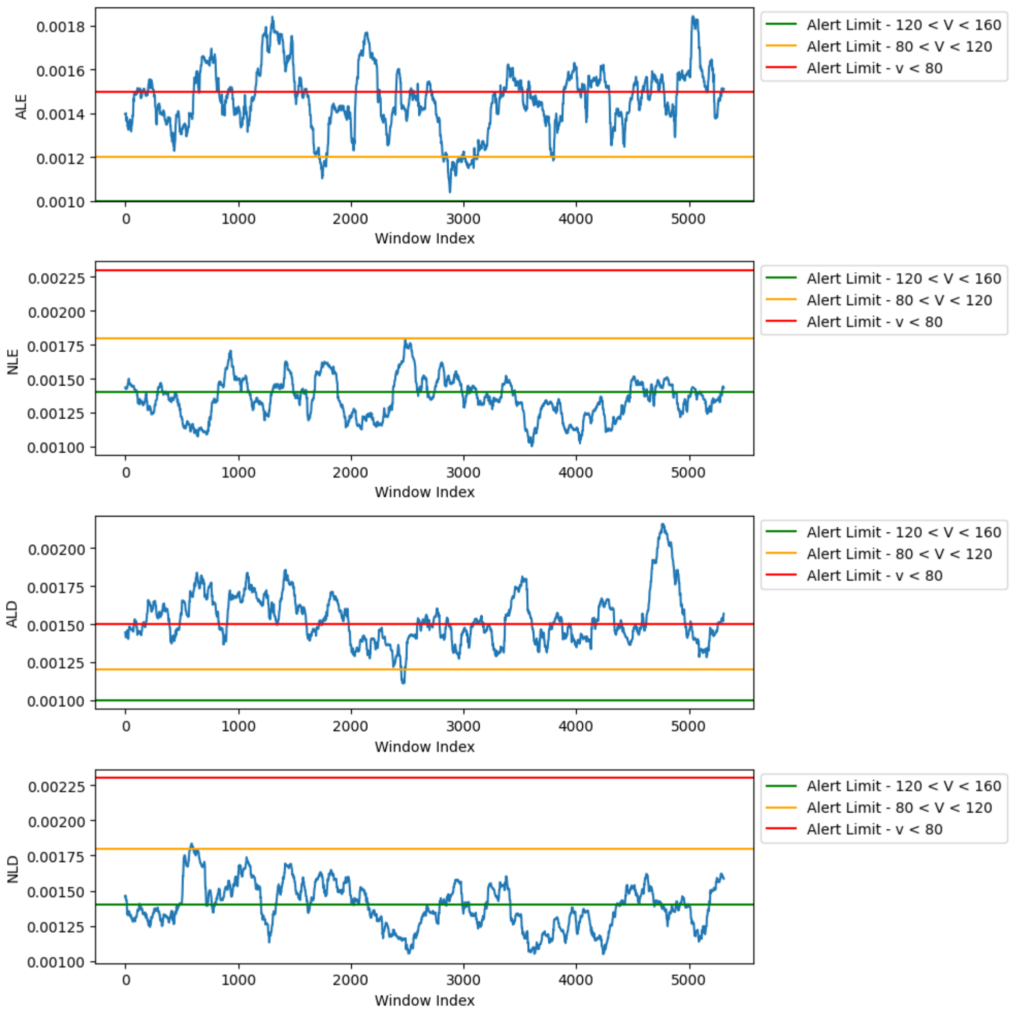


Figure A.2: Irregularities for Damaged Scenario 3

A.3 Irregularities for Damaged Scenario 4

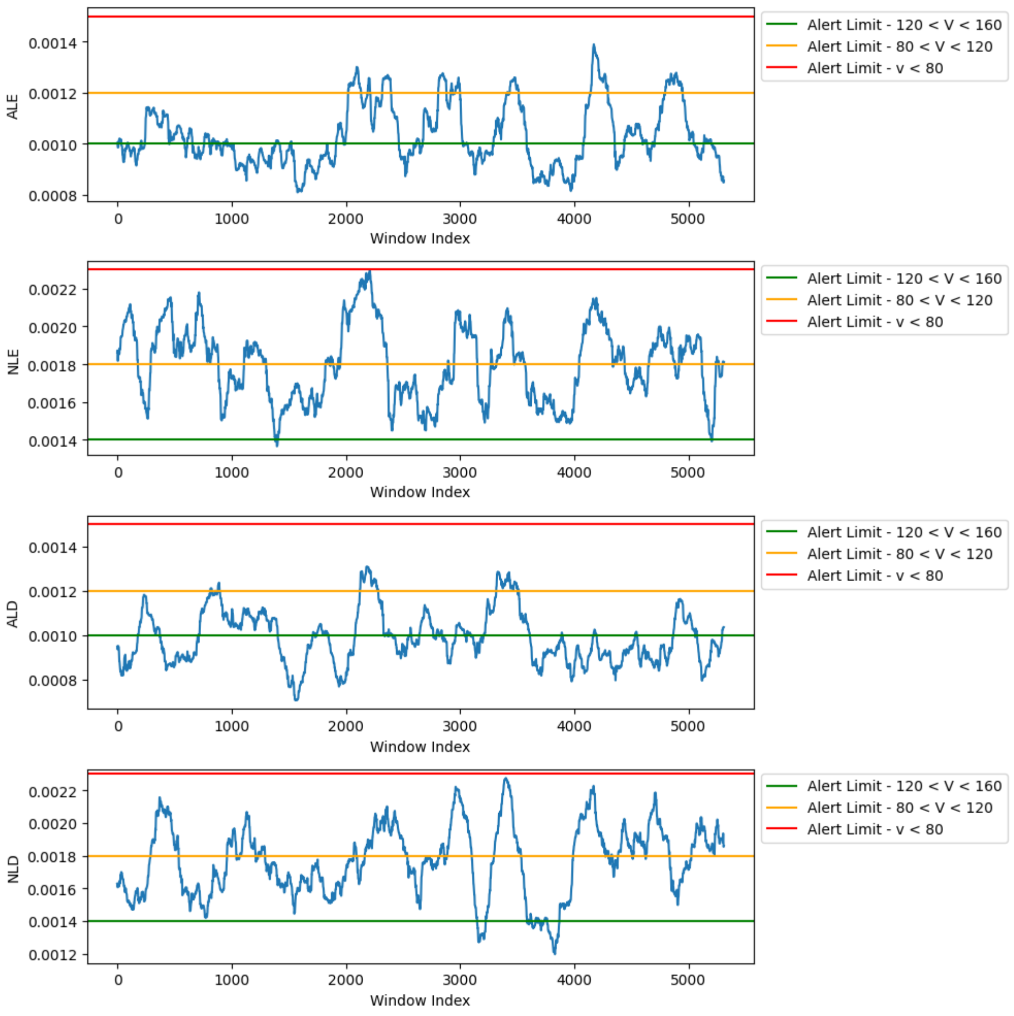


Figure A.3: Irregularities for Damaged Scenario 4

A.4 Irregularities for Damaged Scenario 5

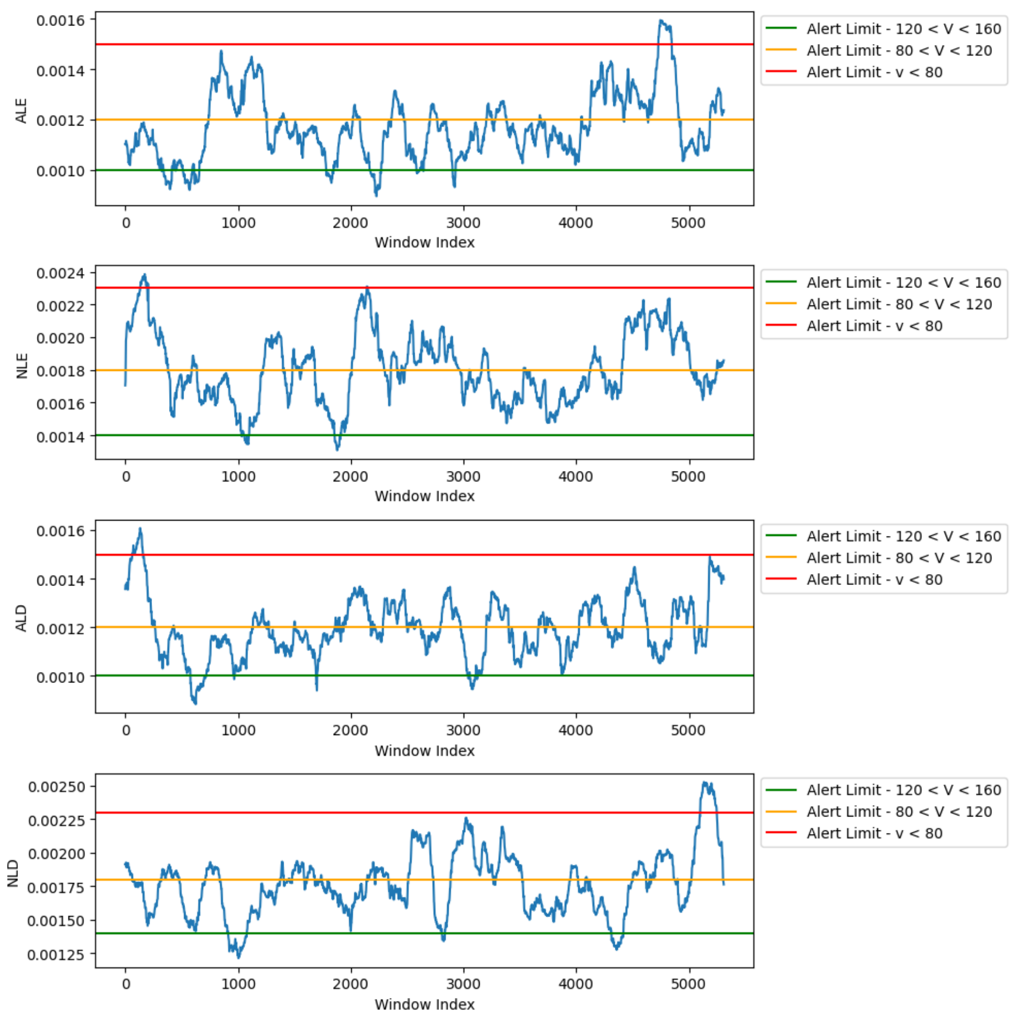


Figure A.4: Irregularities for Damaged Scenario 5

A.5 Irregularities for Damaged Scenario 6

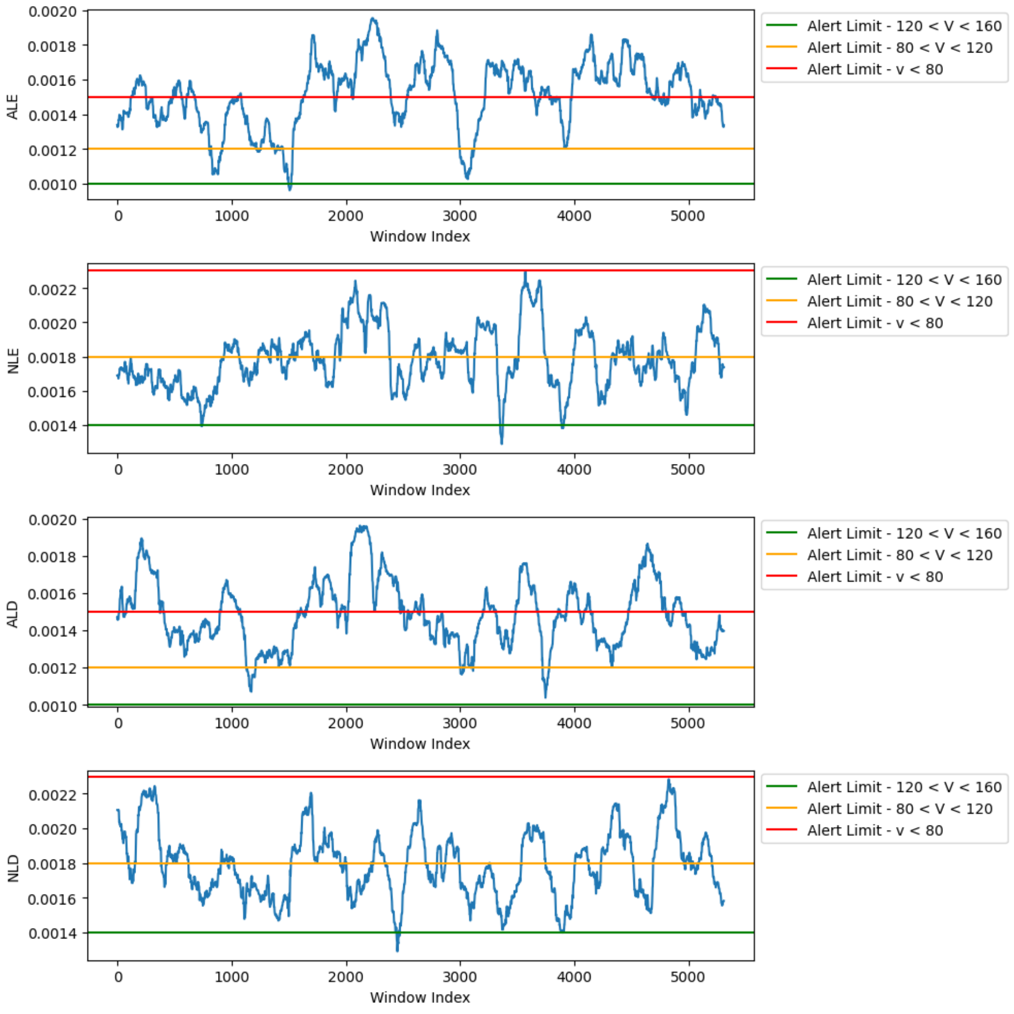


Figure A.5: Irregularities for Damaged Scenario 6

A.6 Irregularities for Damaged Scenario 7

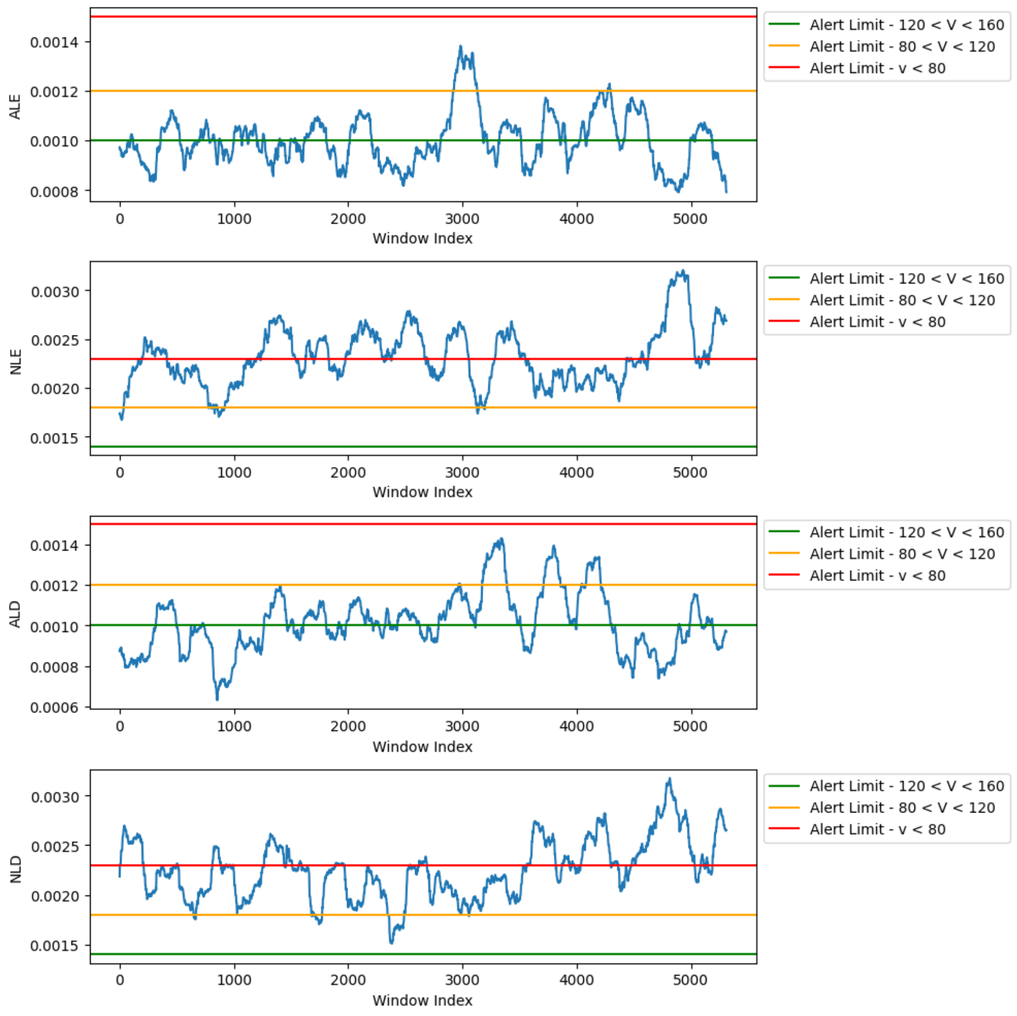


Figure A.6: Irregularities for Damaged Scenario 7

A.7 Irregularities for Damaged Scenario 8

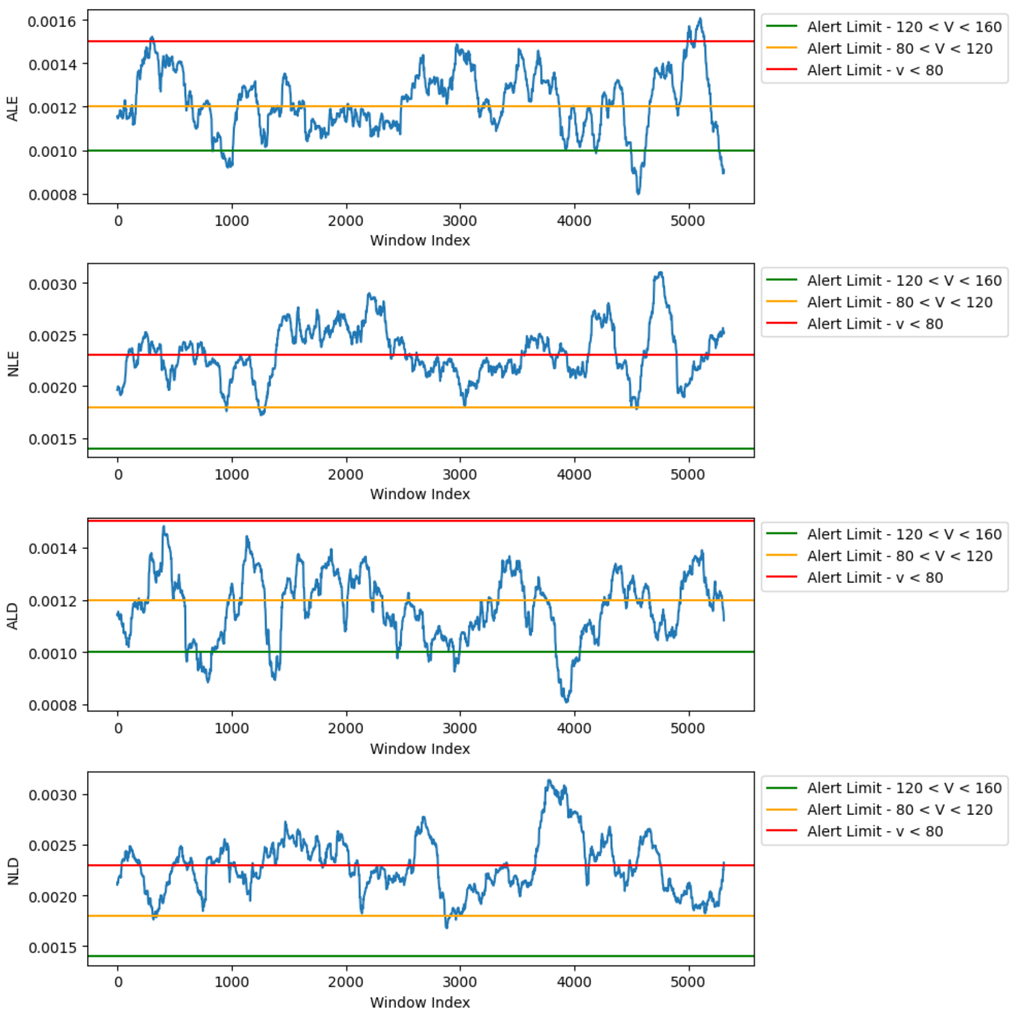


Figure A.7: Irregularities for Damaged Scenario 8

A.8 Irregularities for Damaged Scenario 9

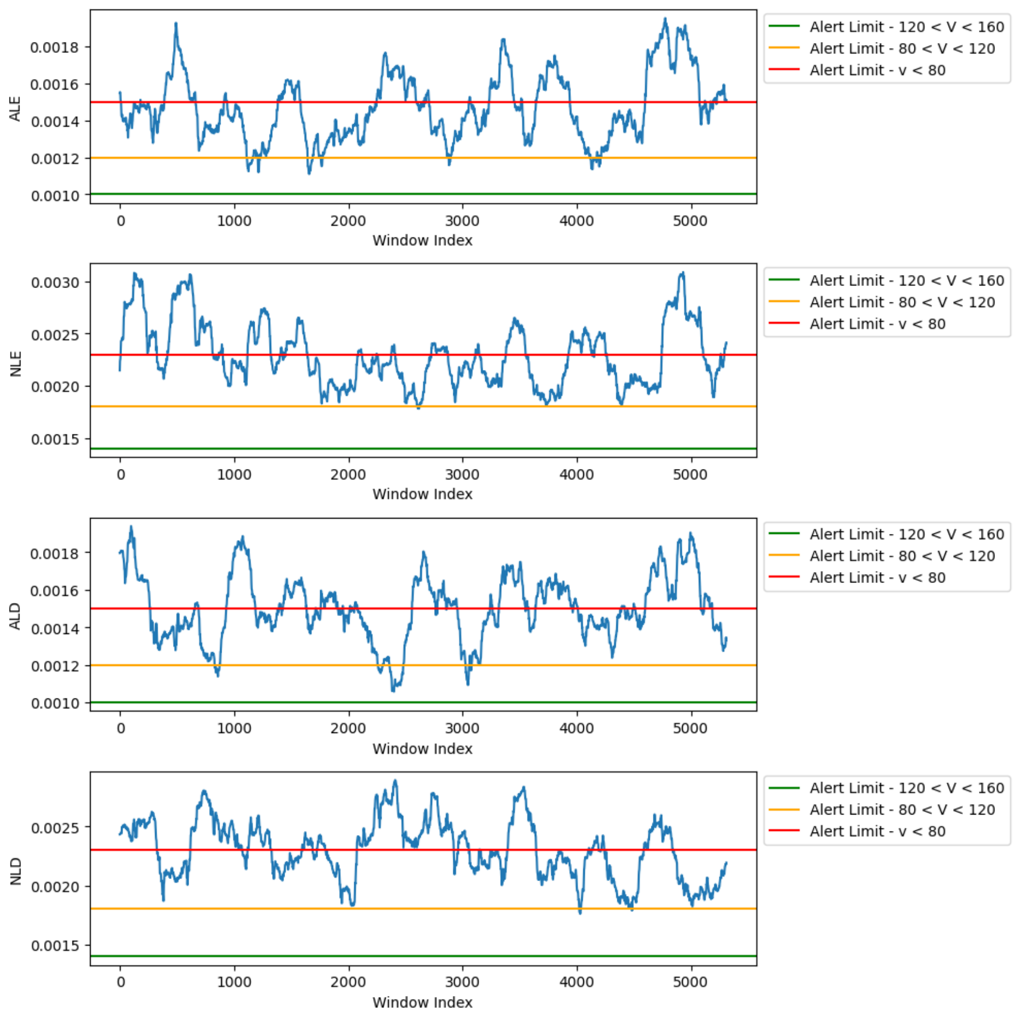


Figure A.8: Irregularities for Damaged Scenario 9

Appendix B

Graphs for Accelerations in Damaged Scenario 1

B.1 Front Left Wheel Horizontal Acceleration - S2_y1

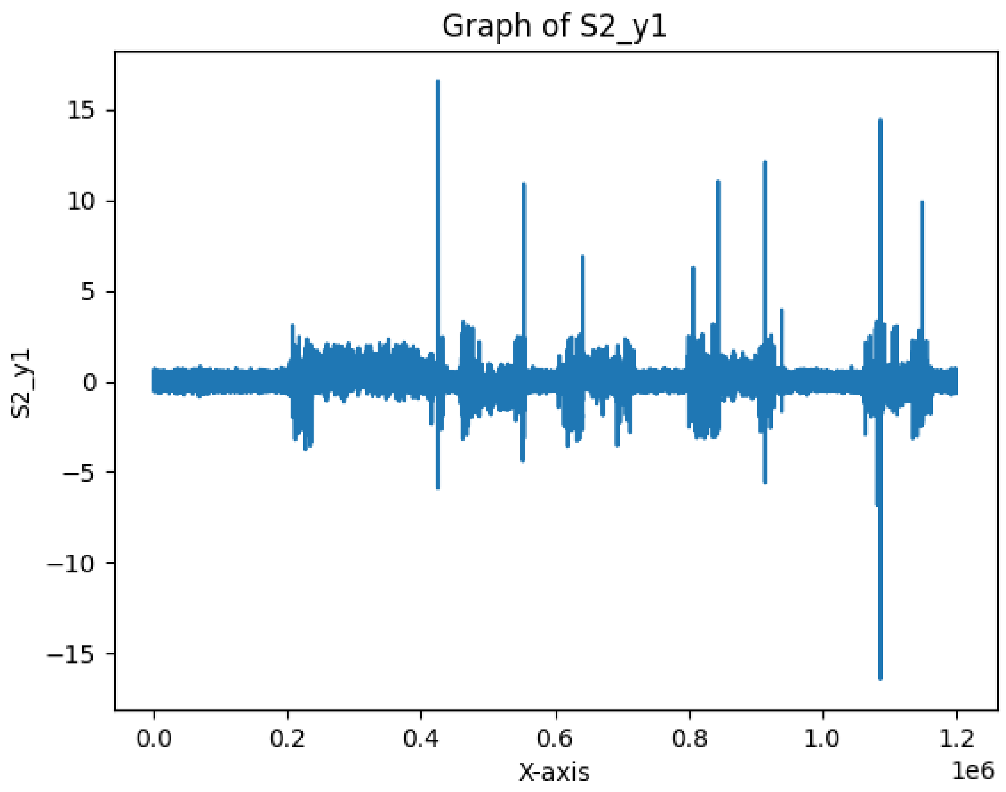


Figure B.1: Front Left Wheel Horizontal Acceleration

B.2 Front Right Wheel Vertical Acceleration - S3_z1

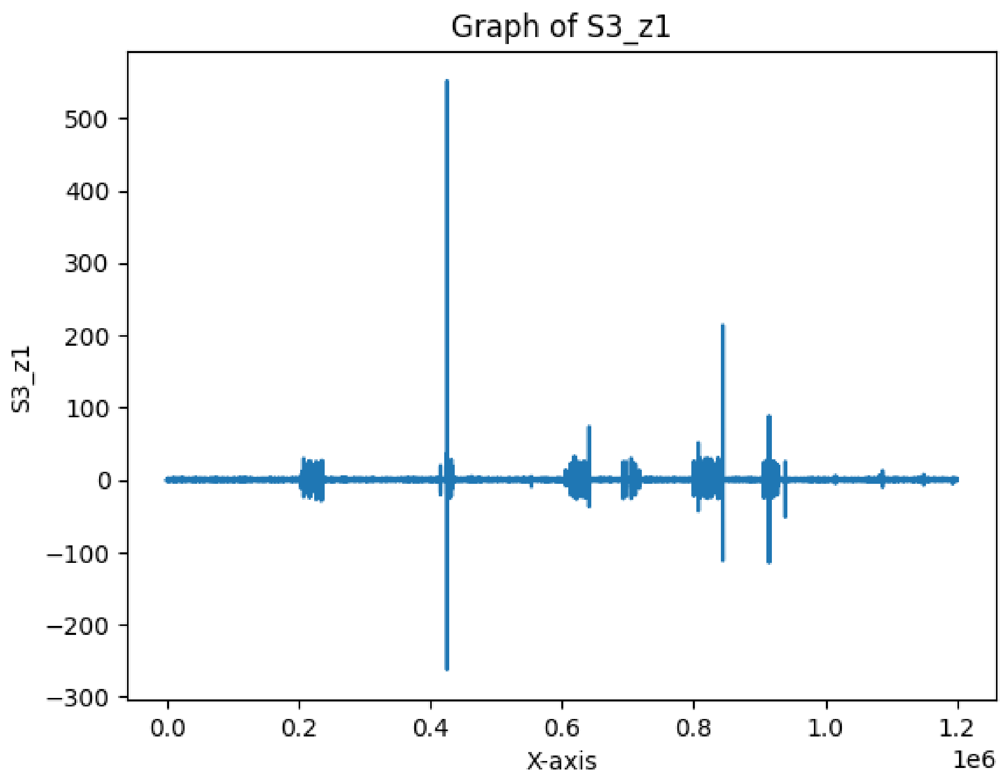


Figure B.2: Front Right Wheel Vertical Acceleration

B.3 Front Right Wheel Horizontal Acceleration - S3_y1

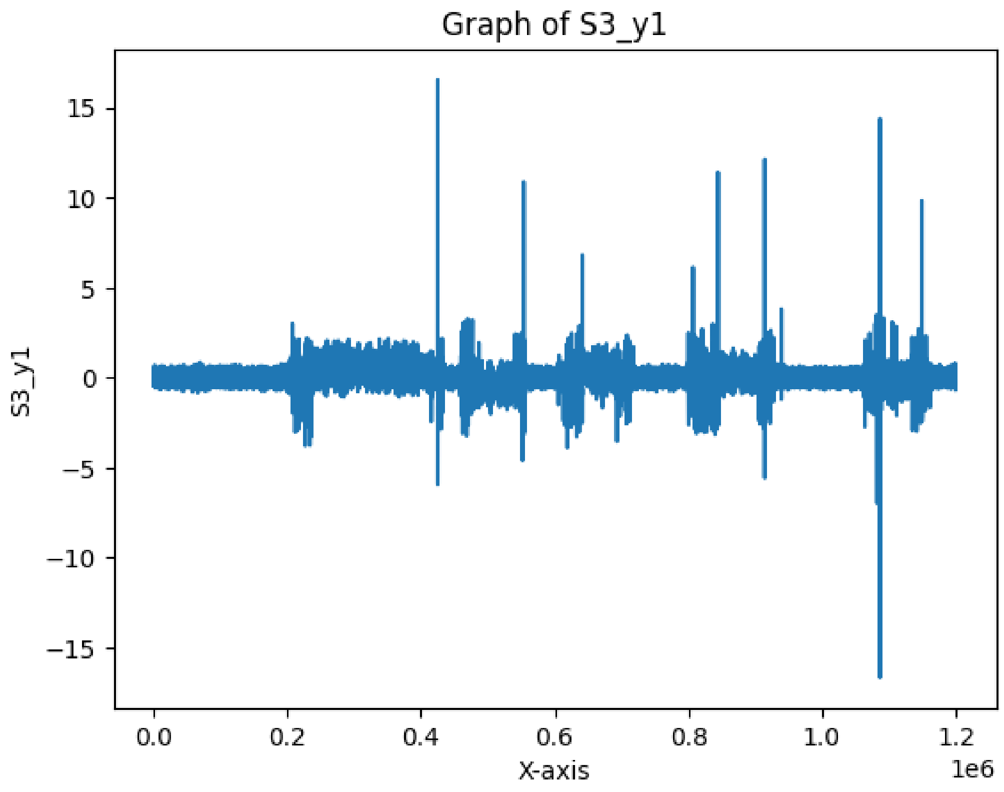


Figure B.3: Front Right Wheel Horizontal Acceleration

B.4 Rear Left Wheel Vertical Acceleration - S4_z1

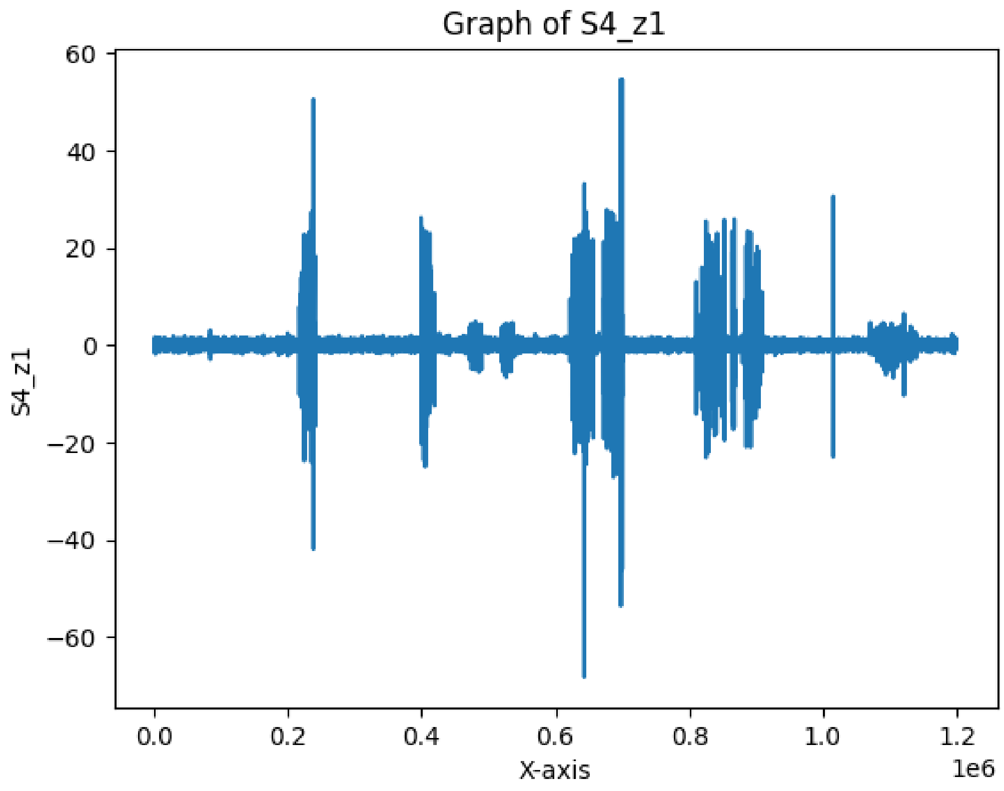


Figure B.4: Rear Left Wheel Vertical Acceleration

B.5 Rear Left Wheel Horizontal Acceleration - S4_y1

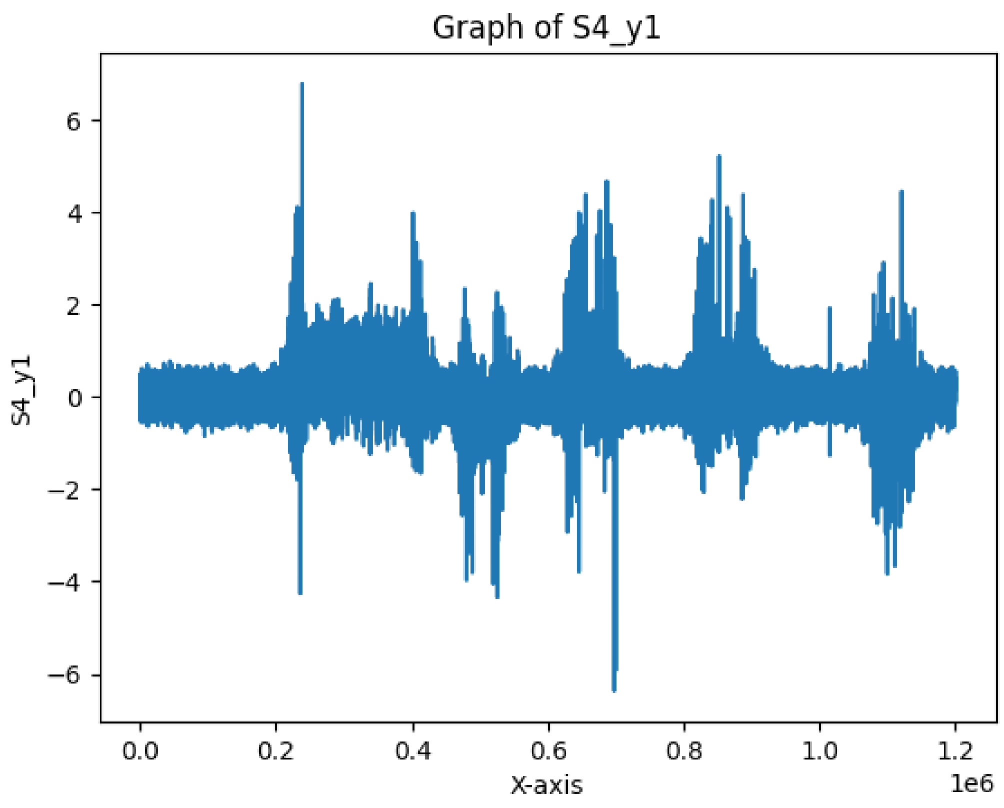


Figure B.5: Rear Left Wheel Horizontal Acceleration

B.6 Rear Right Wheel Vertical Acceleration - S1_z1

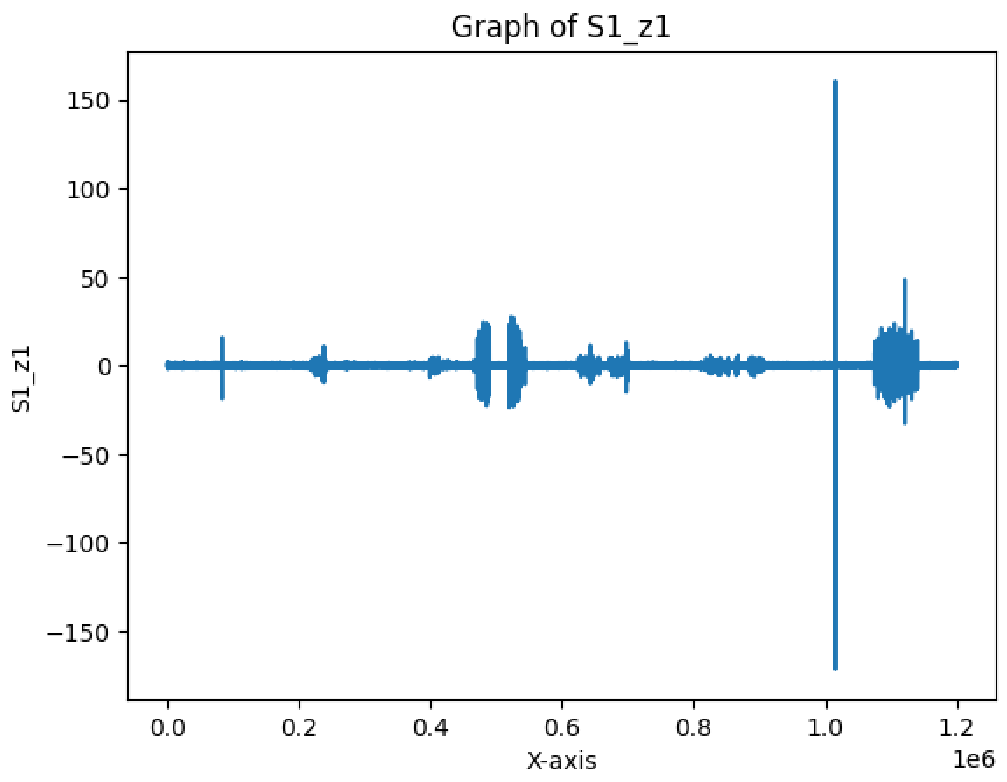


Figure B.6: Rear Right Wheel Vertical Acceleration

B.7 Rear Right Wheel Horizontal Acceleration - S1_y1

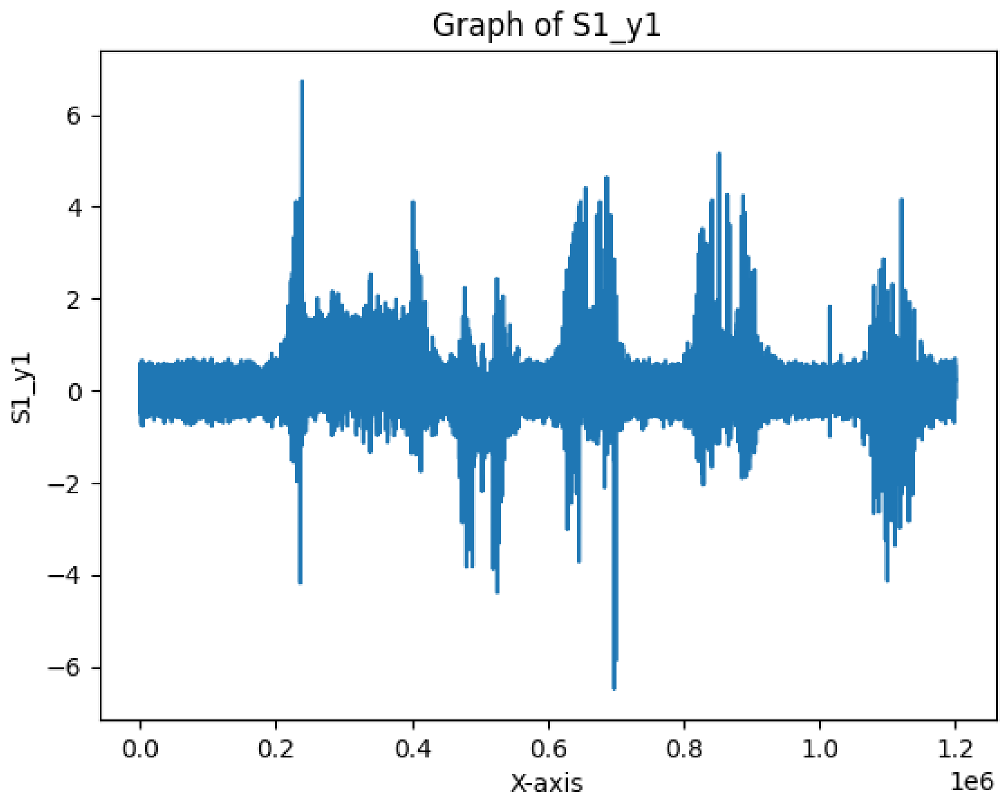


Figure B.7: Rear Right Wheel Horizontal Acceleration

References

[1] Inspeção e diagnóstico ferroviário. https://www.infraestruturasdeportugal.pt/pt-pt/inspecao-diagnostico-ferroviario.

[2] Ferrovia 4.0. http://ferrovia40.pt/, Jul 2021.

[3] Mariam Aboelwafa, Karim Seddik, Mohamed Eldefrawy, Yasser Gadallah, and Mikael Gidlund. A machine learning-based technique for false data injection attacks detection in industrial iot. *IEEE Internet of Things Journal*, PP:1–1, 05 2020.

[4] Ilhan Aydin, Erhan Akin, and Mehmet Karakose. Defect classification based on deep features for railway tracks in sustainable transportation. *Applied Soft Computing*, 111, 11 2021.

[5] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[6] Costa-D. Goebel M. Robinson P. Breed, G. Electronic tracking tag programming is critical to data collection for behavioral time-series analysis. *Ecosphere*, 2:art10, 2011.

[7] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[8] T. Bruin, K. Verbert, and R. Babuska. Railway track circuit fault diagnosis using recurrent neural networks. *IEEE Trans. Neural Netw. Learning Syst.*, 28(3):523–533, 2017.

[9] D. Cantero and B. Basu. Railway infrastructure damage detection using wavelet transformed acceleration response of traversing vehicle. *Struct. Control Health Monit.*, 22(1):62–70, 2014.

[10] Stojkovic-I. Obradovic Z. Cao, X. A robust data scaling algorithm to improve classification accuracies in biomedical data. *BMC Bioinformatics*, 17, 2016.

[11] Thyago P. Carvalho, Fabrízzio A.A.M.N. Soares, Roberto Vita, Roberto da P. Francisco, João P. Basto, and Symone G.S. Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers and Industrial Engineering*, 137, 11 2019.

[12] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, et al. Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4):1–4, 2015.

[13] Xiaoping Chen, Qiang Li, and Bo Zhang. Noise-resistant support vector machines. *Pattern Recognition*, 38(2):283–287, 2005.

[14] Davide Chicco, Matthijs J Warrens, and Giuseppe Jurman. The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation. *PeerJ Computer Science*, 7:e623, 2021.

[15] Carlos Cipriano. Alfa pendular sÓ acelera ao máximo em 23 da viagem entre lisboa e porto. https://www.publico.pt/2019/06/30/economia/noticia/alfa-pendular-so-acelera-maximo-23-viagem-lisboa-porto-1878136, Jun 2019.

[16] Liwen Cui, Xiaolei Xie, Zuojun Shen, Rui Lu, and Haibo Wang. Prediction of the healthcare resource utilization using multi-output regression models. *IISE Transactions on Healthcare Systems Engineering*, 8(4):291–302, 2018.

[17] Jovani Dalzochio, Rafael Kunst, Edison Pignaton, Alecio Binotto, Srijnan Sanyal, Jose Favilla, and Jorge Barbosa. Machine learning and reasoning for predictive maintenance in industry 4.0: Current status and challenges, 12 2020.

[18] N. Davari, B. Veloso, G. Costa, P. Pereira, R. Ribeiro, and J. Gama. A survey on data-driven predictive maintenance for the railway industry. *Sensors*, 21(21):5739, 2021.

[19] Comboios de Portugal. Como viajar nos comboios cp: Cp - comboios de portugal. https://www.cp.pt/passageiros/pt/como-viajar.

[20] Comboios de Portugal. Frota de material circulante: Cp - comboios de portugal. https://www.cp.pt/institucional/pt/cultura-ferroviaria/frota-material-circulante.

[21] Gopi Krishna Durbhaka and Barani Selvaraj. Predictive maintenance for wind turbine diagnostics using vibration signal analysis based on collaborative recommendation approach. pages 1839–1842. Institute of Electrical and Electronics Engineers Inc., 11 2016.

[22] Norma Europeia. Normal portugues: Aplicações ferroviárias - via - qualidade da via, parte 5 - níveis de qualidade da geometria da via.

[23] Shahrzad Faghih-Roohi, Siamak Hajizadeh, Alfredo Nunez, Robert Babuska, and Bart De Schutter. Deep convolutional neural networks for detection of rail surface defects. volume 2016-October, pages 2584–2589. Institute of Electrical and Electronics Engineers Inc., 10 2016.

[24] Abbasi R. Awais M. Imran M. Ning H. Szathmary L. Ghori, K. Performance analysis of different types of machine learning classifiers for non-technical loss detection. *IEEE Access*, 8:16033–16048, 2020.

[25] M. Hamadache, S. Dutta, O. Olaby, R. Ambur, E. Stewart, and R. Dixon. On the fault detection and diagnosis of railway switch and crossing systems: An overview. *Applied Sciences*, 9(23):5129, 2019.

[26] V. Hodge, S. O'Keefe, M. Weeks, and A. Moulds. Wireless sensor networks for condition monitoring in the railway industry: A survey. *IEEE Trans. Intell. Transport. Syst.*, 16(3):1088–1106, 2015.

[27] Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1, 2015.

[28] Nick Hotz. Crisp-dm. https://www.datascience-pm.com/crisp-dm-2/, Nov 2022.

[29] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin. Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning. *IEEE Trans. Veh. Technol.*, 69(12):14413–14423, 2020.

[30] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[31] Camilo Laiton-Bonadiez, John W. Branch-Bedoya, Julian Zapata-Cortes, Edwin Paipa-Sanabria, and Martin Arango-Serna. Industry 4.0 technologies applied to the rail transportation industry: A systematic review, 4 2022.

[32] Ahmed Lasisi and Nii Attoh-Okine. Principal components analysis and track quality index: A machine learning approach. *Transportation Research Part C: Emerging Technologies*, 91:230–248, 6 2018.

[33] J. Lee, H. Choi, D. Park, Y. Chung, H. Kim, and S. Yoon. Fault detection and diagnosis of railway point machines by sound analysis. *Sensors*, 16(4):549, 2016.

[34] Y. Liao, L. Han, H. Wang, and H. Zhang. Prediction models for railway track geometry degradation using machine learning methods: a review. *Sensors*, 22(19):7275, 2022.

[35] Filip Miljković. *Chemoinformatics-Driven Approaches for Kinase Drug Discovery*. PhD thesis, 01 2020.

[36] Y. Mohapatra and M. Ray. Software fault prediction based on gso-ga optimization with kernel based svm classification. *IJIES*, 11(5):152–161, 2018.

[37] Araliya Mosleh, Andreia Meixedo, Diogo Ribeiro, Pedro Montenegro, and Rui Calçada. Automatic clustering-based approach for train wheels condition monitoring. *International Journal of Rail Transportation*, 2022.

[38] Avinash Navlani. Multi-layer perceptron neural network using python - machine learning geek. https://machinelearninggeek.com/multi-layer-perceptron-neural-network-using-python/, Apr 2021.

[39] Tomović-S. Radusinovic I. Nikolić, N. A comparative study of deep learning and decision tree based ensemble learning algorithms for network traffic identification. *Telfor J*, 14:61–66, 2022.

[40] Mai ODASHIMA, Shohei AZAMI, Yasukumi NAGANUMA, Hirotaka MORI, and Hitoshi TSUNASHIMA. Track geometry estimation of a conventional railway from car-body acceleration measurement. *Mechanical Engineering Journal*, 4:16–00498–16–00498, 2017.

[41] Zhao-X. Pan X. Ye W. Peng, P. Gas classification using deep convolutional neural networks. *Sensors*, 18:157, 2018.

[42] Nguyen-V. Le A. Bui V. Pham, T. Udcats: a comprehensive unsupervised deep learning framework for detecting collective anomalies in time series. 2022.

[43] Bernhard Schölkopf, Alexander J Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Advances in kernel methods*, pages 327–352. MIT Press, 1999.

[44] R. Shafique, H. Siddiqui, F. Rustam, S. Ullah, M. Siddique, E. Lee, and S. Dudley. A novel approach to railway track faults detection using acoustic analysis. *Sensors*, 21(18):6221, 2021.

[45] Adrian Stetco, Fateme Dinmohammadi, Xingyu Zhao, Valentin Robu, David Flynn, Mike Barnes, John Keane, and Goran Nenadic. Machine learning methods for wind turbine condition monitoring: A review, 4 2019.

[46] Mingjian Sun, Yan Wang, Xin Zhang, Yipeng Liu, Qiang Wei, Yi Shen, and Naizhang Feng. Feature selection and classification algorithm for non-destructive detecting of high-speed rail defects based on vibration signals. pages 819–823. Institute of Electrical and Electronics Engineers Inc., 2014.

[47] Gian Antonio Susto, Andrea Schirru, Simone Pampuri, Seán McLoone, and Alessandro Beghi. Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Transactions on Industrial Informatics*, 11:812–820, 6 2015.

[48] H. Tsunashima. Condition monitoring of railway tracks from car-body vibration using a machine learning technique. *Applied Sciences*, 9(13):2734, 2019.

[49] Hitoshi Tsunashima. Condition monitoring of railway tracks from car-body vibration using a machine learning technique. *Applied Sciences (Switzerland)*, 9, 7 2019.

[50] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science Business Media, 2013.

[51] Auch-M. Doblander C. Mandl P. Jacobsen H. Weber, M. Transfer learning with time series data: a systematic mapping study. *IEEE Access*, 9:165409–165432, 2021.

[52] B. Williams, J. Carver, S. Matsumoto, T. Kakimoto, and K. Matsumoto. The effects of over and under sampling on fault-prone module detection. 2007.

[53] G. Yan, Y. Bai, C. Yu, and C. Yu. A survey on fault diagnosis approaches for rolling bearings of railway vehicles. *Processes*, 10(4):724, 2022.

[54] Chunsheng Yang, Yanmin Sun, Chris Ladubec, and Yan Liu. Article developing machine learning-based models for railway inspection. *Applied Sciences (Switzerland)*, 11:1–15, 1 2021.

[55] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

[56] Zhong Zheng, Haoyang Qi, Li Zhuang, and Zijun Zhang. Automated rail surface crack analytics using deep data-driven models and transfer learning. *Sustainable Cities and Society*, 70, 7 2021.

[57] S. Zhou, M. Xiao, K. Valaskova, M. Filip, and G. Geng. Remaining useful life prediction and fault diagnosis of rolling bearings based on short-time fourier transform and convolutional neural network. *Shock and Vibration*, 2020:1–14, 2020.