UNIVERSIDADE PORTUCALENSE INFANTE D. HENRIQUE

Predictive Maintenace with LSTM autoencoder and Explainability by SHAP and LIME

Luis Miguel Vieira de Brito

Mestrado Ciência dos Dados

Supervisor: Prof. Bruno Veloso

Co-Supervisor: Prof. Dr. João Gama

October 3, 2023

Predictive Maintenace with LSTM autoencoder and Explainability by SHAP and LIME

Luis Miguel Vieira de Brito

Mestrado Ciência dos Dados

October 3, 2023

Abstract

Predictive maintenance is becoming increasingly important as machines and equipment become more complex. This is because modern machines and equipment have sensors and control systems that can generate a large amount of data. This data can be used to monitor the operating status of equipment and identify signs of failure.

The versatility of LSTM (Long Short-term memory) makes it possible to use it with quite different data sets, with results that allow anomalies to be detected in data sets with different characteristics.

This dissertation will demonstrate that it is possible to use LSTM on two data sets and successfully detect both anomalies. A first set with information from NASA bearing sensors will be used. The objective is to detect changes in the frequency of the data that could indicate a possible anomaly.

The architecture used with the NASA data set was applied to a new data set from the Porto Metro in order to understand whether it was possible to use the same architecture with positive results in different data sets.

Humans do not easily interpret the results presented in many machine learning models, so SHAP and LIME will be used to understand the model results.

SHAP is a method for explaining the predictions of machine learning models that provide a global perspective through the quantification of the contribution of each feature to the final prediction of the model. LIME is another explanation model that works by creating a simpler model that approximates the behaviour of the original model.

It will be possible to demonstrate through the results obtained that, despite being powerful tools used to understand the predictions of machine learning models, they have their limitations. In the case of SHAP, the results are not easy to interpret, unlike LIME, where the results presented are easy to interpret.

Keywords: LSTM, Anomaly Detection, SHAP, LIME, Predictive Maintenance

Resumo

A manutenção preditiva está a tornar-se cada vez mais importante à medida que as máquinas e equipamentos ficam mais complexos. Isto é porque máquinas e equipamentos modernos estão equipados com sensores e sistemas de controle capazes de gerar uma grande quantidade de dados. Esses dados podem ser utilizados para monitorizar o funcionamento do equipamento e identificar sinais de falha.

A versatilidade do LSTM (Long Short-term memory) torna possível a sua utilização com conjuntos de dados bastantes distintos, com resultados que permitem que sejam detetadas anomalias em conjuntos de dados com características diferentes.

Nesta dissertação, vai ser demonstrado que é possível utilizar o LSTM em dois conjuntos de dados e detetar as anomalias existentes nos dois com sucesso. Vai ser utilizado um primeiro conjunto com informação de sensores de rolamentos da NASA. O objetivo é detetar alterações na frequência dos dados que possam ser indicativos de uma possível anomalia.

A arquitetura utilizada com o conjunto de dados da NASA foi aplicada um um novo conjunto de dados com origem no Metro do Porto para que fosse possível compreender se em conjunto de dados diferentes era possível utilizar a mesma arquitetura com resultados positivos.

Os resultados apresentados em muitos dos modelos de machine learning não são facilmente interpretados pelo ser humano, para isso vai ser utilizado o SHAP e o LIME para que os resultados do modelo possam ser entendidos.

O SHAP é um método de explicação das previsões de modelos de machine learning que dá uma perspectiva global através da quantificação do contributo de cada característica na previsão final do modelo. O LIME é um outro modelo de explicação que funciona através da criação de um modelo mais simples que se aproxima do comportamento do modelo original.

Vai ser possível demonstrar através dos resultados obtidos que, apesar de serem ferramentas poderosas utilizadas para entender as previsões de modelos de machine learning, têm as suas limitações. No caso do SHAP, os resultados não são de fácil interpretação, ao contrário do LIME, onde os resultados apresentados são interpretados de forma fácil.

Keywords: LSTM, Deteção de Anomalias, SHAP, LIME, Manutenção Preditiva

Acknowledgements

I would like to express my gratitude to the prof. Bruno Veloso for is support and guidance throughout the elaboration of these thesis. His insights and advice were invaluable in helping me to develop this work.
I would also like to thank my colleague Ricardo Cruz for his suggestions. His insights helped me to improve my work.
Finally, I would like to thank my family for their support throughout this process. Their patience and encouragement that helped me to stay motivated and persevere.

"Believe you can and you're halfway there."

Theodore Roosevelt

"Until I began to learn to draw,
I was never much interested in looking at art."

Richard P. Feynman

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial inteligence
AMRules	Adaptive Model Rules
fp	False positive
fn	False negative
LSTM	Long Short-term memory
LIME	Local Interpretable Model-agnostic Explanations
NASA	National Aeronautics and Space Administration
PdM	Predictive maintenance
RMSE	Root Mean Square Error
SHAP	SHapley Additive exPlanations
tp	True positive
tn	true negative
XAI	Explainable artificial intelligence

Chapter 1

Introduction

With advances in industrial technology, companies are now generating large amounts of data that must be used to make daily decisions. To maximise resource utilisation and reduce equipment downtime, companies are increasingly turning to predictive maintenance as a way to reduce costs and ensure production quality.
The collection of data from the various sensors allows detection of possible malfunctions in advance, enabling the resolution of the problems before they happen and thereby reducing maintenance costs that represents 15 % to 60 % of total operating costs and improving the productivity [31].

To improve efficiency, industries are investing heavily in machine learning models that can help prevent equipment failure. This use case is known as predictive maintenance.
However, for predictive maintenance to be effective, it is important to understand why the model makes the predictions that it does. This is where explainable AI (XAI) comes in. XAI models are more transparent and explainable, which makes them ideal for understanding the predictions of machine learning models.

Van Lent first introduced the concept of explainable AI (XAI) in 2004 to describe how his proposed system could explain the behavior of AI systems in simulation games. However, while machine learning algorithms have evolved rapidly in recent years, equipment maintenance has only evolved significantly over a longer period.

Lately, interest in explainable AI (XAI) has grown rapidly, reflecting the need to better understand the decisions made by AI and machine learning algorithms [1].

1.1 Context and Motivation

There are three main types of maintenance: corrective, preventive, and predictive maintenance.
Corrective maintenance involves addressing problems after they occur. This type of maintenance is often unplanned and can lead to equipment downtime.Preventive maintenance involves taking steps to prevent problems from happening in the first place. This type of maintenance is

often scheduled and can help to reduce equipment downtime and extend the lifespan of equipment.Predictive maintenance uses data analytics to predict when problems are likely to occur. This type of maintenance can help to prevent equipment downtime and ensure that equipment is operating at peak efficiency [31].

Industrial equipment has become increasingly complex and expensive, with less tolerance to failures and performance degradation [7].

Most organizations today only become aware of equipment failures when they occur or when an alarm goes off indicating that a component is about to fail. This alarm is typically triggered only when the failure is imminent, leaving little time to plan a maintenance response [15].
The main objective of the various maintenance strategies is to extend the useful life of the equipment and reduce maintenance costs. Predictive maintenance has been capturing the industry's attention because it has proved to be the most advantageous due to its ability to optimise the equipment's use, as seen in Figure 1.1.

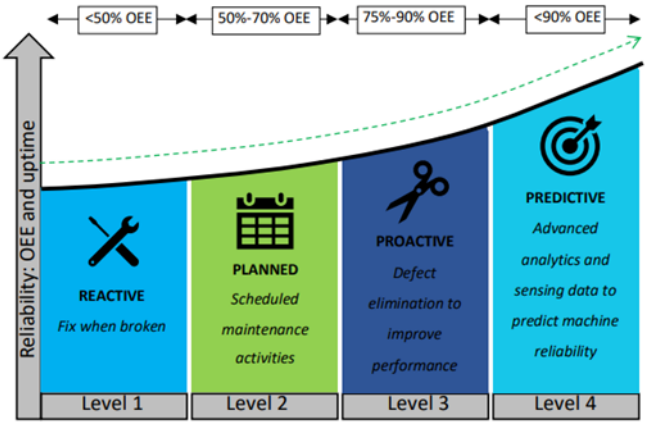


Figure 1.1: Maintenance types [6]

Predictive maintenance (PdM) can reduce maintenance costs, downtime, and the number of replacement parts required, while increasing equipment life and profits.One of the main goals of PdM is to collect data about the health of equipment, which can be used to identify faults and predict future equipment status [6].

Çınar et al. [6] says Deloitte ranks the technology that drives predictive maintenance into five categories.The categories are sensors, networks, integration, increased intelligence, and behaviour,These five categories are essential for predictive maintenance.
There is no one-size-fits-all solution to the problems encountered in the industry. For each problem, it is necessary to choose the most appropriate technique and prepare the data accordingly [6].

Predictive maintenance models are not just used in industry; they are also in high demand for transportation, especially long-haul vehicles and rail transport. This demand is driven by increasing global trade and fuel costs [15].

Railway companies need to increase their speed of operation, but without compromising the safety of their passengers and cargo. To achieve this, they are shifting from reactive to proactive maintenance [15].

One of the factors that can reduce the speed of rail services is service interruptions caused by derailments or alarms indicating a potential breakdown. Railway companies have invested heavily in implementing networks to monitor the condition of locomotives and tracks.The data collected from these sensors can be used to study mechanisms that enable predictive maintenance, preventing service interruptions. If a service interruption is necessary, it should be very short because the source of the problem has been accurately identified, making it easier to resolve [15].

The effectiveness of AI models was limited by their inability to explain their decision-making process to the users. This triggered a new research area named Interpretable Machine Learning or Explainable Artificial Intelligence(XAI), focusing on making machine learning models that can explain their rationale and characterise their strengths and weaknesses, giving an understanding of how they will behave in the future [28].

There is, however, no definition of this concept that most people accept,but according to DARPA, Explainable AI is about developing AI models that can explain themselves to humans in a way that is clear, concise, and accurate. This is important because it allows us to understand how AI models make decisions and to identify any potential biases or errors [1].

Confidence in the results of a forecasting model is essential because humans need to understand the rationale behind the model's decisions and evaluate the model before using it. Knowing how the model was generated gives confidence that it will work well with real-world data [21].

There is growing interest in explainable machine learning algorithms, with the goal of developing a comprehensive and unified framework of concepts and categories. This interest is also reflected in the recent focus on explainable artificial intelligence (XAI) from governments, particularly in the context of the European General Data Protection Regulation (GDPR). XAI is important because it can help us to understand and trust AI systems, and to identify and mitigate bias in their decisions [9].

Many companies are integrating explainability into the artificial intelligence (AI) lifecycle to ensure that their algorithms are ethical and fair for their users. This integration is important because workers need to be able to explain to customers how the AI algorithm works, and to bridge the gap between data scientists who build the models and business teams who use the models to make decisions [28]. An explanation is a way to understand the reasoning behind an AI algorithm's decision. This is important because it allows us to trust the results of AI algorithms and to use them more effectively [9].

The figure1.2 shows the expected improvements when using XAI techniques to support user decision-making.

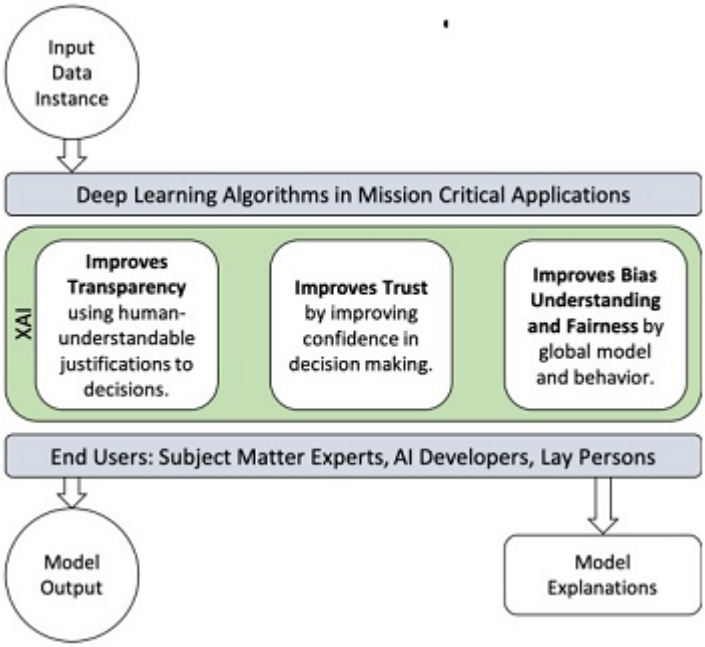


Figure 1.2: Diagram of the different explanatory options [9]

Introducing explainable algorithms into machine learning models has raised concerns about the quality of the decisions made by those models. As algorithms become more complex in order to improve performance, they also become more difficult to explain. However, with advances in explainability algorithms, it is possible to reverse this trend.

Figure 1.3 shows the margin of improvement while maintaining the gap between the explanatory of the models and the performance [2].

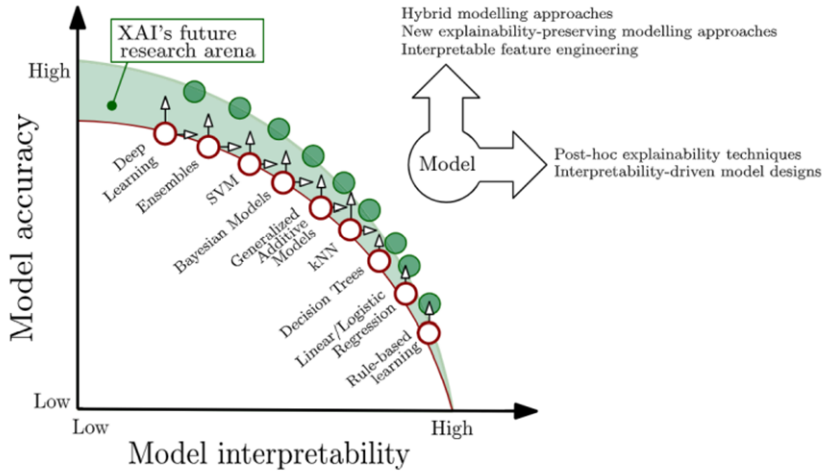


Figure 1.3: Gap between explainability and performance and improvement[2]

A prediction explanation is a textual or visual representation of the features that influenced a machine learning model's prediction. This explanation should allow humans to understand how the model made its decision and to resonate with the model and its prediction results [21].

There are some characteristics that explainable algorithms have to possess: (i) they must be easily interpreted, and (ii) they should produce a qualitative understanding between the input variables and the final result. This requirement implies that explanations should be easily understood,however the ability to interpret also depends on the target audience [21].

An explanatory model must have two important characteristics: local fidelity and global fidelity.Local fidelity means that the model must be able to give meaningful explanations for individual predictions. Global fidelity means that the model must be able to explain the overall behavior of the original model.An explanatory model must be agnostic, meaning that it should be able to explain any model, regardless of how the model works. This is important because it allows the use of explanatory models to explain black box models.Explaining predictions with a global perspective is essential to trust the model. This is because it allows us to understand how the model is making decisions overall, and to identify any potential biases or errors [21].

Global explanation methods can explain the entire model, while local explanation methods can only explain a single prediction. Posthoc explanations can be either model-agnostic, meaning they can be applied to any model as long as the features and model outputs match the selected explanation model, or model-specific, meaning they are restricted to a specific model or class of models [27].

Interpretable models are essential for high-risk environments where the model outcomes can have serious consequences. Therefore, it is crucial to analyze and explain the results obtained [28].

1.2 Objectives

This dissertation has two main objectives.

Objective 1: Develop an algorithm with the capacity to detect anomalies preventively using data from the Porto Metro and NASA, thus preventing damage and reducing the risk of accidents.

Objective 2: Provide explanations to the anomalies detected.This objective is important to ensure that operators accept and use the system. If operators need help understanding the reasons for the predictions, they may be less likely to trust the system and may not take the appropriate action. The explanation algorithm will help to address this issue by providing operators with clear and concise explanations of the predictions.

1.3 Dissertation Structure

Chapter one introduces the study's topic and provides the research's motivation and objectives.

Chapter Two is a literature review published on predictive maintenance and explainability. The focus is identifying the most used methods for detecting anomalies and explainability.

Chapter three contains the materials and methods used in this dissertation. This chapter provides an overview of the data used in this study, the methods used to detect anomalies, and the explanation method used to identify the features that contribute most to the model's prediction of an anomaly in a given record. In this chapter is also made a data analyses using some of the most common metrics.

In the chapter four are the results and discussion. This chapter presents the study's results, including the performance of the anomaly detection models and the explanation models. It also provides a general overview of the findings.

Chapter five contains the conclusions and Future Work. This chapter summarises the findings of the study and discusses the implications of these findings. This chapter will also discuss new methods that are being developed for anomaly detection and explainability. These methods may be more effective than the methods used in this study.

Chapter 2

Literature Review

The literature review for this dissertation provides an overview of some research works that have been conducted in the field of predictive maintenance and explainable algorithms, and discusses how they can be applied to real-world problems.

2.1 Predictive maintenance

As some countries revitalize the industry, it is becoming necessary to implement methods that improve efficiency. Predictive maintenance, which uses machine learning and deep learning models, has been widely applied in the industry to achieve this goal.

The implementation of machine learning models has increased the number of articles published about predictive maintenance. This interest has been increasing since 2013 [30].

According to Carvalho et al. [5], only two articles were published with the keywords "predictive maintenance" or "PDM" until 2013. However, from 2013 to 2018, an average of 11.3 articles were published per year. This growing interest in predictive maintenance is likely due to the increase in the amount of data generated by the industry and the advances in machine learning algorithms [5].

A variety of machine learning algorithms can be used for predictive maintenance.The most common algorithms used in the studies Carvalho et al. [5] analyzed were random forest (33 %), neural networks (27 %), support vector machines (25 %), and k-means (13 %) [5].

Most research articles on predictive maintenance follow a similar implementation process. The typical steps involved are data acquisition, data preprocessing, component wear identification, and failure prediction [31].

Table 2.1 lists articles where predictive maintenance was implemented using machine learning algorithms.

Article List				
Author	Task	Algorithm	DataSet	Metrics
[31]	Literature review			
[5]	Literature review			
[15]	Catastrofic alarm prediction and failure prediction	SVM and Decision tree	Rail network	
[14]	Detect wind turbine abnormalities and predict maintenance needs	Random forest,Decision tree and statistical analysis	Taipower	Accuracy
[18]	Predictive maintenance for railway points	Data cluster	Railway system	
[4]	Predict maintenance need for railway switch	Decision tree, random forest and gradient boosted tree	Railway system	F-Score,Accuracy, Misclassification, Kappa
[20]	Predict the Remaining useful life of a component	Random Forest, Beam search algorithm and Kolmogorov-Smirnov test	Logged Vehicle Data and Volvo Service Records	Accuracy
[8]	Predict the rate of degradation and remaining useful life of railcar wheel bearing	Kurtosis analysis, statistical analysis and MATLAB 2018b	Railcar system	Kustosis resul
[23]	four-layers big data architecture	Hadoop framework	Railway Interlocking System	

Table 2.1: Article list

Zhang et al. [30] describe data-based predictive maintenance as having two phases: a learning phase and a prediction phase.The learning phase involves training a model on sensor history data, while the prediction phase involves using the trained model to predict and make decisions.Each phase has three sub-processes: data acquisition, feature engineering, and model training and prediction.Traditional machine learning algorithms generally require a large amount of data from health conditions and multiple failure scenarios in order to train a model and engineer features from the time, frequency, and time-frequency domains. These features are then used to represent the device's health.On the other hand, deep learning models can learn directly from data without the need for feature engineering.This is the main difference between machine learning and deep learning algorithms.

Machine learning algorithms typically require a large amount of data from both healthy and faulty components to train the model. After training, the model can make real-time predictions about the condition of components [30].

Several challenges in creating a predictive maintenance model were identified by Zhang et al. [30] as showed in the figure 2.1.

Stages	*Challenge Descriptions*
Operational Assessment	• **time cost** (e.g., system development, debugging, and deployment) • **economic cost** (e.g., research and hardware expenditure, as well as developer cost) • **safety** (e.g., equipment operation safety and personnel safety) • **restriction** (e.g., environmental restrictions on data collection)
Data Acquisition	• **development mode** (e.g., integrated development or independent development) • **budget** (e.g., design cycle) • **conversion ability** (i.e., the ability to transform sensor data into health status information) • **system performance** (e.g., useful life and maintenance cost)
Feature Engineering	• require **expert knowledge** and **prior experience** • **signal analyzing** (e.g., time-, frequency-, and time frequency domain) • feature design, extraction, selection
Modeling	• **model selection** (based on model complexity) • **model training** (whether the data is labeled) • **model predicting** (time efficiency and accuracy requir -ement)

Figure 2.1: Challenges creating predictive maintenance [30]

Zhang et al. [30] also identifies a set of statistical methods that can be used in predictive maintenance as demonstrated in the figure 2.2.

No.	Name	Equation	No.	Name	Equation	No.	Name	Equation
1	Maximum	$s_1 = max(x_i)$	8	Mean square	$s_8 = \frac{1}{N}\sum_{i=1}^{N} x_i^2$	15	Variance frequency	$F_2 = \frac{1}{N}\sum_{j=1}^{N}(x_j - \bar{x})^2$
2	Minimum	$s_2 = min(x_i)$	9	Root mean square	$s_9 = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$	16	Standard deviation frequency	$F_3 = \sqrt{\sum_{j=1}^{N}((f_j - F_1)X_j) \Big/ \sum_{j=1}^{N} X_j}$
3	Median	$s_3 = \begin{cases} x_{(N+1)/2}, N \text{ is odd} \\ \frac{x_{N/2} + x_{(N+1)/2}}{2}, N \text{ is even} \end{cases}$	10	Skewness	$s_{10} = \frac{1}{N}\sum_{i=1}^{N} \frac{(x_i - s_5)^3}{s_7^3}$	17	Root mean square frequency	$F_4 = \sqrt{\sum_{j=1}^{N}(f_j^2 X_j) \Big/ \sum_{j=1}^{N} X_j}$
4	Peak-to-peak	$s_4 = max\lvert x_i\rvert - min\lvert x_i\rvert$	11	Kurtosis	$s_{11} = \frac{1}{N}\sum_{i=1}^{N} \frac{(x_i - s_5)^4}{s_7^4}$	18	Spectral skewness	$F_5 = \sum_{j=1}^{N}\left(\frac{f_j - F_1}{F_3}\right)^3 S(f_j)$
5	Mean	$s_5 = \frac{1}{N}\sum_{i=1}^{N} x_i$	12	Skewness factor	$s_{12} = \frac{1}{N}\sum_{i=1}^{N} x_i^3 \Big/ (\sqrt{s_8})^3$	19	Spectral kurtosis	$F_6 = \sum_{j=1}^{N}\left(\frac{f_j - F_1}{F_3}\right)^4 S(f_j)$
6	Variance	$s_6 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$	13	Kurtosis factor	$s_{13} = \frac{1}{N}\sum_{i=1}^{N} x_i^4 \Big/ (\sqrt{s_8})^4$	20	Spectral power	$F_7 = \sum_{j=1}^{N}(f_j)^3 S(f_j)$
7	Standard deviation	$s_7 = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2}$	14	Mean frequency	$F_1 = \frac{1}{N}\sum_{j=1}^{N} f_j$	21	Wavelet energy	$F_8 = \sum_{j=1}^{N} \omega t_\phi^2(j)/N$

Figure 2.2: Statistical methods [30]

For traditional machine learning algorithms, data retrieval and feature selection are useful for optimization. For deep learning algorithms, the architecture and the largest dimension of vector characteristics are more important for improving metrics.An analysis of various methods used for predictive maintenance concluded that machine learning and deep learning models are effective because they achieve results between 95.06 % and 100 %. AI algorithms have shown a superior performance advantage with the significant improvement of computational power and data volume

growth.With the continuous innovation of algorithms, research will focus on data-driven predictive maintenance applications [30].

Hsu et al. [14] used statistical methods in conjunction with machine learning models to detect and predict breakdowns in wind turbine components. To detect anomalies, they used a statistical program based on the frequency of each anomaly to elaborate a list of items to be verified whenever a problem was detected in a wind turbine. They also used scatter plots to determine the relationship between the attributes and detect anomalies.To predict future malfunctions, they implemented two machine learning algorithms: random forest and decision tree. Random forest algorithms were used to prevent over-fitting, while decision tree algorithms provide comprehensive explanations and allow for the identification of the features needed to predict maintenance needs. In both cases, the data was validated with a cross-validation k-fold.
The results of the two algorithms were similar, with the decision tree algorithm achieving an accuracy of 92.86 % in the training data and 92.62 % in the test data. After cross-validation, the average accuracy was 92.68 %. The random forest algorithm achieved an accuracy of 99.71 % in the training data and 95.34 % in the test data. After cross-validation, the average accuracy was 92.16 %. These results verified that using either model was effective and created a list of factors to consider as possible failures [14].
Statistical methods are very present in the literature regarding preventive maintenance.

Patalay [18] used the mean and signal variations of sensor data to create clusters of different failure modes. They then used these clusters to detect and identify possible failures in real time.To create the clusters, the authors simulated multi-component failures in a lab setting. This allowed them to create unique clusters for each type of failure. They then compared the real-time sensor data to these clusters to detect possible failures and identify the most likely type of failure.
Statistical methods used to perform fault testing are not confined to the most common measures such as mean, median, standard deviation and other similar methods.

Daniyan et al. [8] used kurtosis analysis with an envelope spectrum to continuously monitor and predict the degradation and remaining useful life of a railcar wheel bearing. Kurtosis analysis is a method of detecting anomalies in a signal by measuring the peakedness of the signal distribution. However, it can be difficult to detect weak features in a signal with a low signal-to-noise ratio.The envelope spectrum is a method of extracting the underlying signal from a noisy signal. It does this by removing the high-frequency components of the signal and focusing on the low-frequency components.By combining kurtosis analysis with an envelope spectrum, Daniyan et al. were able to detect even weak anomalies in the signal, which could indicate degradation of the wheel bearing.This allowed them to predict the remaining useful life of the wheel bearing and to take preventive action before it failed.

Predictive maintenance can also be used in the automotive industry, especially commercial trucks and buses. Prytz et al. [20] used data collected from scheduled maintenance of many vehicles over a three-year period to create a model that could predict the need for air compressor repair. They balanced the data to avoid bias towards the majority class, as only a small percentage of the vehicles had compressor failures.
The model was created using the Random Forest algorithm with 10-fold cross-validation and independent datasets for training and testing. The datasets were divided randomly by vehicle, ensuring that no vehicle information appeared in both the training and test datasets.The authors used the beam search algorithm and a filter based on the Kolmogorov-Smirnov test to select features. To calculate the maintenance cost, they used several K-NN algorithms, achieving accuracy greater than 75 % at all points tested.The study demonstrates the usefulness of algorithms in making predictions [20].

Predictive maintenance research in the railway industry is increasingly gaining attention. Most of the published work focuses on solutions to assess the health of infrastructure, such as railway points and interlocking systems [10].
The railway infrastructure is critical, but it is facing increasing demands and rapid deterioration. Budget cuts have limited maintenance, making it difficult to keep up with the needs of the network [4].

Bukhsh et al. [4] identified a need for more tools and models to support decision-making in the railway industry to help plan maintenance more effectively and efficiently. To address this gap, they proposed decision tree, random forest, and gradient-boosted tree models to predict the need for maintenance on rail switches.The authors used four different data sources and performed some data preprocessing, class labeling, and approach definition. They then created the decision tree, random forest, and gradient-boosted tree models using hyperparameters. They found that the gradient-boosted tree algorithm achieved the best results on the test data, with an accuracy of 86.2 %. The random forest algorithm achieved an accuracy of 85.7 %.The authors evaluated the models using 10-fold stratified cross-validation. The gradient-boosted tree algorithm achieved an accuracy of 92.3 %, and the random forest algorithm achieved an accuracy of 91.3 % [4].

One of the problems in predictive prevention for the railway infrastructure is the amount of data generated by the most varied components. However, it needs an efficient way to collect this data. Salierno et al. [23] proposed a four-tier architecture to cover all the fundamental data collection steps. The four layers of the proposed architecture consisted of a storage layer, a processing layer, a service layer, and an integration layer.The architecture implementation used the Hadoop framework that allows the processing of large volumes of data by distributing this data across multiple clusters of computers [23].

IBM and the American railway company US Class I and Li et al. [15] developed an analytical model of failures based on data collected from multiple systems. The model consists of five main steps: feature extraction, data size reduction, model training, prediction, and confidence level and finally, a simplification of the rules. The features were extracted using statistical methods such as mean, maximum value and 95 percentile. For the training model that aimed to predict whether

a bearing would trigger a level 1 alarm in the next 3 to 7 days, a variant of the support vector machine (SVM) algorithm was used with a 5-fold cross-validation using real data to evaluate the accuracy [15].
To evaluate the SVM model, the authors compared it to a baseline decision tree algorithm. The SVM model outperformed the decision tree model on both tests.For true positives, the SVM model achieved an accuracy of 97.59 %, while the decision tree model achieved an accuracy of 91.55 %. For false positives, the SVM model achieved an accuracy of 0 %, while the decision tree model achieved an accuracy of 6.85%.The authors used the same data modeling process to create a model to predict bad trucks or wheels three months in advance based on data from three months in the past.They created a decision tree model, splitting the data into 80 % for training and 20 % for testing. The model achieved a true positive rate of 97 % in both the training and test data, and a false positive rate of 0.20 % in the training data and 0.23 % in the test data.The authors were able to extract human-interpretable rules from the model, as each leaf corresponds to a primitive rule [15].

2.2 Previous Works in Explainable Artificial Intelligence

While it is possible to interpret decision tree models, it is difficult to understand why a particular decision was made. An explainable algorithm is a way to explain the reasoning behind a machine learning model's decision [9].

Open Source XAI Platforms (Explainable Artificial Intelligence Platforms) have been developed to help build trust and transparency in AI. These platforms provide tools and techniques to help users understand how AI models work and why they make the decisions they do. IBM developed AI Fairness 360, an open-source toolkit that can help detect and remove bias in machine learning models. Microsoft developed the Microsoft Model interpretability in Azure ML, a tool that helps users understand what AI models are doing from functional and ethical aspects. Google developed the What If Tool, an interactive visual interface designed to probe the models. H20.ai has the H2O Driverless AI, a platform that has the capability of explaining machine learning algorithms using a unique combination of techniques and methodologies such as LIME, SHAP, and decision trees in an interactive dashboard to explain the results and the Oracle Skater that is a unified framework to enable Model Interpretation for all forms of models. These platforms can be used by a variety of stakeholders, including data scientists, machine learning engineers, and business users, to better understand and explain AI models. This can help to build trust in AI systems and ensure that they are used in a fair and responsible manner [24].

Siddavatam et al. [25] proposed using a decision tree with Monte Carlo simulation to try to replicate the results of a trained and highly accurate model.Decision trees are among the most understandable machine learning algorithms, making them a good choice for this purpose. A person can simply follow a path through the tree to see how the algorithm determines the output. Additionally, decision trees can be tailored to provide as much or as little explanation as needed.However, using a decision tree to mimic a black box model may not perfectly represent

the black box. The quality of the database used to train the decision tree will have a significant impact on the accuracy of the mimicking model. Additionally, because this method is more of a simulation than a model itself, it is expected that the accuracy of the mimicking model will be lower than the accuracy of the black box model [25].

LIME (Local Interpretable Model-Agnostic Explanation) is a popular algorithm that provides interpretability for black-box machine learning models. It does this by creating a local interpretable model that explains the predictions of the black-box model for a specific instance.LIME is model-agnostic, meaning that it can be used to explain any black-box model, regardless of how it was trained.
LIME works by creating a set of synthetic data points around the instance to be explained. These synthetic data points are created by perturbing the features of the original instance. LIME then trains a simple interpretable model, such as a linear regression model, on the synthetic data points. This interpretable model is then used to explain the prediction of the black-box model for the original instance.
LIME is a powerful tool for understanding how black-box machine learning models work. It can be used to identify the features that are most important for the model's predictions and to understand how changes in the features will affect the model's output.

Garreau and Luxburg [13] analyzed LIME theoretically and concluded that it can be used for both regression and classification tasks. They also found that LIME is able to discover interesting features that are important for the model's predictions, but it may also forget some crucial features.

Patel and Shanbhag [19] used SHAP (Shapley Additive exPlanations) to interpret a machine learning model that predicts machine failure from sensor data. SHAP is a game-theoretic approach that measures the contribution of each feature to the model's prediction. This is done by calculating the change in the prediction when a single feature is removed. The higher the SHAP value of a feature, the more important it is to the prediction. This information can be used to understand the general behavior of the model and to ensure that it is functioning correctly. Patel also used SHAP to generate bar plots that show the absolute SHAP values for all features over all predicted data points. These plots can be used to identify the components of the machine that are failing and the components that are likely to cause failure.
SHAP is a preferable way of interpreting machine learning models over other methods, such as feature importance. This is because SHAP allows us to understand the rationale behind the model's predictions. This is important for predictive maintenance applications, where we need to understand which components of the machine are likely to fail [19].

Situ et al. [26] proposed a Learning to Explain (L2E) approach to explain black-box text classification models. L2E uses a deep neural network (DNN) to learn the behavior of an explanation algorithm, such as LIME. Once trained, the DNN can be used to explain new instances much faster than the original explanation algorithm.L2E takes two inputs: a black-box text classification model and an explanation algorithm. It then trains the DNN to predict the explanation that the explanation algorithm would generate for a given input text. This process is called refining the explanation algorithm into an explanatory model. Situ et al. [26] evaluated L2E on three text clas-

sification tasks and found that it generated more stable explanations than six popular explanation algorithms. L2E also explained instances much faster than the other algorithms, by a factor of 5 to 7,5x10^4 times [26].
The L2E approach has the potential to improve the interpretability of black-box text classification models and make them more useful for real-world applications [26].

De et al. [11] proposed a hybrid approach to explainable artificial intelligence that combines two prior methods: clustering of a network hidden layer and the TREPAN decision tree algorithm. This approach allows for the visualization of the flow of information within a deep neural network and the generation of human-interpretable explanations for each result at an individual level.The TREPAN algorithm works by training a decision tree to approximate the function of a trained neural network. A unique feature of TREPAN is that it draws a new sample while the tree grows based on the minimum sample criteria for each node.De et al. [11] combined clustering of hidden layer outputs with the TREPAN decision tree to develop an interpretation of the neural network classifier. This approach is more appropriate than approximating a network with a decision tree at an overall level.After implementing the proposed approach, the authors found that the decision path of each leaf node was logical and sensible. Figure 2.3 shows the analysis of two clusters made by De et al. [11].

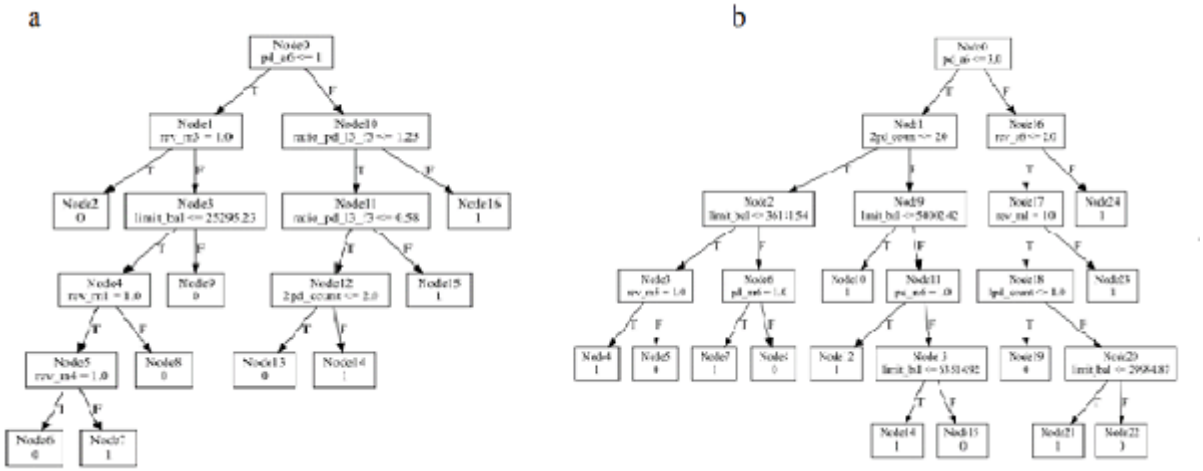


Figure 2.3: a) TREPAN Decision Tree for cluster 9;b) TREPAN Decision Tree for cluster 4 [11]

The same analysis was made to all other clusters, and it was found that TREPAN generated meaningful human interpretable reason codes to explain the predictions given by the Neuronal Network model. To compare the results of his proposal, he compared the TREPAN model with the results of LIME. LIME focuses on training local substitution models to explain individual predictions in a human-interpretable manner [11].

Comparison between TREPAN,LIME	
TREPAN	LIME
TREPAN gives a set of reason codes that combined form a decision rule or path to explain the prediction outcome	Reason codes are discontinuous, imprecise and could be ambiguous.LIME gives a list of reason codes that can explain the predicted class or explain the other class or classes
The rules are formed based on the most significant features that depict the flow or sequence of information used by the predictive model, and the decision rules are human-interpretable	Not all the reason codes are completely human-interpretable and may or may not be satisfied by the customer under consideration
It does not need human intervention to draw an explanation and explain a predicted outcome. It gives a single human interpretable decision rule	LIME gives a list of reason codes, and the user has to select the best to explain the prediction outcome for that instance.

Table 2.2: Comparison TREPAN and LIME[11]

De et al. [11] concluded that his approach that combines clustering of relationships learned by a neuronal network and the application of TREPAN decision tree to extract reason codes or decision rules at cluster level outperformed the "Global TREPAN" in fidelity terms and LIME in comprehensibility, generating better-quality reason codes that provide concise human interpretable explanations.

Bukhsh et al. [4] used LIME to explain their model predictions by analyzing feature importance and per-case details. LIME shows how the contribution of each feature to the final prediction varies from case to case, compared to feature importance analysis, which only provides a global view. Some features may have a significant impact on the model prediction for a single case, as shown in the instance graphs. However, the same feature may have a lower feature importance score globally because it is fragmented based on its values deep in a tree [4].

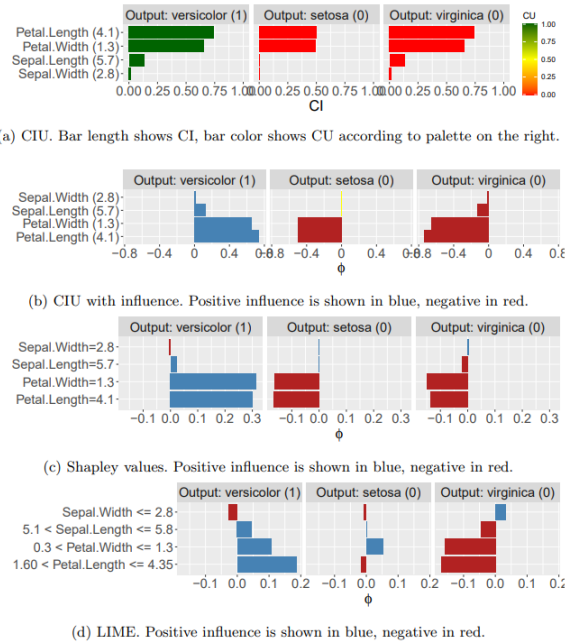
Matzka [16] created an explainable model and exploratory interface using a decision tree algorithm. The model was trained on 15 decision trees, each with a maximum of 4 nodes. This was done to make the trees easy to interpret by humans. Each decision tree used only 4 of 6 available features. The model worked by providing a group of data points to the 15 decision trees. If more than one decision tree returned a positive result, the tree using the highest feature sum of the predictor importance was chosen as an explanation.
Matzka [16] found that the explainable decision trees had some limitations. First, they were not able to detect any failures in the data. This was because the bagged trees used to train the explainable decision trees also had difficulty detecting failures. Second, the explainable decision trees were only able to detect one type of failure, called HDF, in 3 out of 5 cases. The authors believe that this is because they used a low number of nodes in the decision trees.For the remaining data points, the explainable decision trees provided partially valuable explanations in 2 cases and helpful explanations in 9 cases. Matzka [16] concluded that the explanations provided by the decision

trees tend to be of high quality, but in many cases, the trees did not provide any explanation at all [16].

Främling et al. [12] compared Contextual Importance and Utility (CIU) with LIME and SHAP. CIU is a model-agnostic explanation method that is fundamentally different from LIME and SHAP. CIU does not estimate feature influence like most other explanation methods. Instead, it estimates contextual importance and contextual utility. The two core concepts of CIU are importance and utility, which are derived from multi-attribute utility theory. LIME and SHAP do not have a concept of utility and typically use the term importance, or influence in the case of SHAP. The influence of a feature depends on both the feature's importance and the utility of the current feature value. A feature with zero importance for the output will have zero influence on the final result, regardless of its value utility [12]. For Främling et al. [12], the CIU core concept definitions are:

- Importance: The feature importance for a factor in a singular context impacting a particular decision
- Utility: The match of feature values in the same context outcome expectations.
- Influence: The combination of utility and importance value represents a particular decision's positive or negative impact.

CIU is limited to the interval [0,1] by definition. In classification tasks, the output is transformed into utility values. For regression tasks, the outputs must be mapped into utility values using a utility function [12].
Främling et al. [12] used the Iris dataset to compare the outputs of the three methods. Figure 2.4 is possible to see the explanations generated by all the methods for a 'Versicolor' flower with a random forest model.



(a) CIU. Bar length shows CI, bar color shows CU according to palette on the right.

(b) CIU with influence. Positive influence is shown in blue, negative in red.

(c) Shapley values. Positive influence is shown in blue, negative in red.

(d) LIME. Positive influence is shown in blue, negative in red.

Figure 2.4: Explanations for 'versicolor' of Iris dataset [12]

Främling et al. [12] concluded that the output generated by CIU with influence and SHAP are almost identical and differ significantly from LIME. He also claims that the LIME results tend to change from one run to another [12].

The comparison was also made using a regression task. [12] used the Boston Housing dataset with a Gradient Boosting Machine model for that comparison. Figure 2.5 shows the result obtained by the three methods, for instance, 370. The results show that CIU and SHAP obtain similar values and are very different from the LIME results.

(a) CIU.

(b) CIU with influence.

(c) Shapley values.

(d) LIME.

Figure 2.5: Explanations for a instance of Boston dataset [12]

Främling et al. [12]compared Contextual Importance and Utility (CIU) with LIME and SHAP and found that CIU is a promising new alternative to these two popular explanation methods.

The results indicate that CIU explanations may be more rational and faithful to the underlying model than LIME explanations. CIU and SHAP produced similar results on both datasets, which is reassuring.
CIU is more lightweight than SHAP because it only modifies the values of one input feature at a time, requiring fewer samples. However, the number of samples required by CIU still needs to be improved, as does the estimation accuracy. This makes it difficult to make a fair comparison between CIU and other explanation methods [12].

Adaptive Model Rules (AMRules) is a rule-based model for regression problems that can be used to learn complex relationships between attributes and to make accurate predictions. AMRules can also generate explanations for its predictions, making it a valuable tool for interpretable machine learning models.In AMRules, a rule's antecedent can be a combination of attribute values and the result of another rule.The consequence of a rule can be a constant or a linear combination of several attributes. A constant consequent means that the rule always predicts the same value.A linear combination of attributes means that the rule predicts a value based on a weighted sum of the attribute values [22].
AMRules can improve the explicability of machine learning models because they are typically simple and easy to understand and can generate counterfactual explanations for the predictions. Counterfactual explanations describe how the prediction changes if the input features change.
AMRules implements three prediction strategies for each rule:

1. The average of the attribute covered by the rule. This strategy predicts the average value of the attribute for all examples that satisfy the rule's antecedent.

2. A linear combination of independent attributes. This strategy predicts a value based on a weighted sum of the attribute values for all examples that satisfy the rule's antecedent. The weights are learned using a linear regression algorithm.

3. An adaptive strategy that chooses the first two strategies based on which one has the lower MAE. The MAE is a measure of the error between the predicted values and the actual values.

The adaptive strategy ensures that each rule is using the best prediction strategy for the data [22].

Ribeiro et al. proposed combining AMRules with the Chebyshev inequality method to ensure that almost all values in any distribution probability are close to the mean.
The Chebyshev inequality is a mathematical theorem that states that for any probability distribution, almost all values will be within k standard deviations of the mean. This means that for any dataset, almost all values will be close to the mean unless the dataset is skewed or has outliers.
Ribeiro et al. proposed using the Chebyshev inequality to filter the data before training an AMRules model. This helps to ensure that the AMRules model is not biased by outliers or skewed data.

Two sampling methods use the Chebyshev approach:

1. ChebyOS: This under-sampling method selects examples for the training model where a randomly generated number is greater than the Chebyshev probability. This helps to reduce the number of outliers in the training set

2. ChebyOS, i.e. Chebyshev-based Over-Sampling: This method balances the data stream in rare cases of unbalanced data streams. This helps to ensure that the training set contains a representative sample of the data, even if the data is imbalanced.

By combining AMRules with the Chebyshev inequality method, Ribeiro et al. developed a more robust and accurate learning algorithm.
Using his strategy to filter the data before training the models, Ribeiro et al. developed models with lower RMSE (root mean squared error) that could produce more reliable explanations for failures. This is because the Chebyshev inequality helps to ensure that the models are not biased by outliers or skewed data [22].

2.3 Summary

Predictive maintenance studies have shown that random forests and decision trees are the most popular and effective algorithms.Random forests are particularly well-suited for predictive maintenance because they are both accurate and interpretable. This means that they can make predictions with high accuracy and also provide explanations for their predictions.SHAP is one of the most popular algorithms for explaining machine learning models. It has a high success rate in explaining the predictions of machine learning models.

This dissertation will use an LSTM autoencoder to predict failures in Metro do Porto railcars and NASA bearing failure. The LIME and SHAP algorithms will be used to explain the predictions made by the LSTM autoencoder.LSTM autoencoders are a type of neural network that is well-suited for predicting time series data. They will be used to learn the sensor data patterns associated with failures.

The LIME algorithm is a local interpretability method that can be used to explain the predictions made by any machine learning model. It generates a surrogate model that is locally similar to the original model.

The SHAP algorithm is a global interpretability method that can be used to explain the predictions made by any machine learning model. It works by assigning a score to each feature in the model. The score indicates how much the feature contributes to the prediction.

I propose an unsupervised anomaly detection model for predictive maintenance. Unsupervised models are less commonly used in predictive maintenance than supervised models because they are more difficult to verify. However, we will show that unsupervised models can be just as effective as supervised models for detecting anomalies.

I will use two explanation methods, LIME and SHAP, to explain the predictions of our model. We want to compare the effectiveness of these two methods and assess their ability to present results in a way that is easy for humans to understand. More research is needed to compare the effectiveness of LIME and SHAP for explaining predictions using the same dataset.

In section 3, data analysis and pre-processing will be done.

Chapter 3

Methodology

Anomalies are data points that deviate significantly from the expected patterns in a dataset. Anomaly detection is important in real-world scenarios because it can help prevent losses and improve efficiency.
Recent advances in machine learning and statistical analysis have led to the development of more effective anomaly detection techniques. In this work, I propose a neural network-based anomaly detection model that can be used to detect anomalies in two very different datasets with different data and features.

To better understand the reasons why some data points are marked as anomalous, I need to explain the anomalies. This involves identifying the factors that contribute to the unusual or unexpected nature of these data points.

For both datasets, I will explain which features have the greatest contribution to the classification of data points as anomalies.

3.1 Models

An LSTM (Long Short-Term Memory) autoencoder network will be used to detect possible anomalies in the dataset.

An LSTM autoencoder is a neural network with an input layer and an output layer that learns the best way to encode and decode the input data. The input data is encoded and compressed into a latent space, and then the decoder layer decompresses the data. The output is compared with the original data, and the errors are backpropagated through the network to update the weights [17]. The autoencoder is trained by minimizing the reconstruction error [17].

LSTM autoencoders are commonly used for time-series anomaly detection because they can learn the underlying patterns in the data and identify outliers. This can save a lot of time because it makes it much easier to find the anomalies in the dataset.

The development of the model was made using a NASA-bearing dataset.

The dataset was split into two parts: a training set and a test set. The training set consisted of the first month of data, which did not contain any anomalies. The data was preprocessed using a technique called StandardScaler, which standardizes the features by removing the mean and scaling the data to unit variance. This type of preprocessing is often used for continuous numerical features, especially when the features have different scales. This was the only preprocessing performed on the data.

The LSTM model consists of an input layer, a dense layer, a repeat vector layer, a decoder layer, and a TimeDistributed layer.
The input layer receives the raw data and passes it to the next layer.
The dense layer is a very important layer in the neural network because it connects every neuron in the layer to every neuron in the previous layer. This allows the network to learn complex relationships between the features in the data.
The repeat vector layer creates new copies of the input data, which are then fed to the decoder layer.
The decoder layer reconstructs the input data into its original format. This is an essential component in autoencoders, as it allows the network to learn the underlying patterns in the data.
The TimeDistributed layer is a type of layer that is often used in networks that deal with data sequences. It applies the same operation to all elements of a sequence, which allows the network to learn sequential patterns in the data.

The 95th percentile was used as a threshold to identify anomalies in the data. This means that 95% of the loss MAE values in the data are less than or equal to the threshold value. Any data points with a loss MAE value greater than the threshold value are considered anomalies.Loss MAE is a common metric used in anomaly detection models. It measures the average difference between the predicted and actual values for each data point. Higher loss MAE values indicate that the model is less accurate in predicting the actual values.The 95th percentile threshold is a simple and effective way to identify anomalies in data. It is often used in anomaly detection models because it is easy to implement and interpret.

SHAP (SHapley Additive exPlanation) is a powerful framework for explaining the predictions of machine learning models. It provides insights into how individual features contribute to the predictions. To explain the predictions of the anomaly detection model, I used the KernelExplainer with the first 200 occurrences of the training set, which did not contain any anomalies. I then fed a set of data points that the model predicted as anomalies to the explainer to determine the SHAP value for each feature. SHAP values can be visualized to better understand how each feature contributed to the predictions. This is very useful because it allows us to identify the features that are most important for detecting anomalies.

LIME (Local Interpretable Model-agnostic Explanations) is another popular technique for explaining the predictions of machine learning models. It works by approximating the model's behavior with a simpler, more interpretable model. LIME then provides feature importance scores that indicate how much each feature contributes to the prediction.

An explanation model is trained on the first 200 occurrences of the training data to interpret the model's predictions. This model is then used to explain a single occurrence.

The LSTM model trained on the NASA dataset will be applied to the Metro Porto dataset to assess its generalizability. If the model can detect existing anomalies in the Metro Porto dataset with the same settings, then it suggests that the model can be applied to any dataset where anomalies may exist.

3.2 Datasets

3.2.1 Nasa dataset

´

The NASA dataset contains vibration sensor readings from four bearings that were run to failure under constant load over multiple days. The data is collected every 10 minutes and consists of 1-second vibration signal snapshots. Each file contains 20,480 sensor data points per bearing, obtained by reading the bearing sensors at a sampling rate of 20kHz

3.2.1.1 Data Analysis

In Table 3.1, it summarize the data in the NASA dataset. This information is useful for understanding the type, variability, and characteristics of the data, which is essential for making informed decisions about how to use the data.

Information Resume				
Type	Bearing 1	Bearing 2	Bearing 3	Bearing 4
mean	0,080951	0,078543	0,081351	0,04783
std	0,0402	0,011789	0,011607	0,009549
min	0,001168	0,000767	0,000716	0,001699
25%	0,060773	0,07424	0,076829	0,043951
50%	0,062021	0,075206	0,078187	0,044524
75%	0,083277	0,077458	0,080575	0,04813
max	0,453335	0,161016	0,151299	0,119047

Table 3.1: Information Resume NASA

Table 3.2 shows the kurtosis and skewness values of the bearing sensors in the NASA dataset. These values indicate whether there are any points that deviate from the normal distribution, which could be caused by numerous or extreme values.High kurtosis values indicate a large concentration of data points in the tails, indicating the presence of outliers. Positive kurtosis can indicate a distribution that is more peaked than a normal distribution.Positive skewness values mean that the distribution is extended to the right, with most of the data concentrated on the left side of the distribution. Positive skewness can indicate the presence of extreme values or outliers.

The presence of outliers can indicate the presence of anomalies in the data.

Distribution of bearing sensors		
Sensor	Kurtosis	Skewness
Bearing 1	24,687155	4,184574
Bearing 2	19,700267	3,103132
Bearing 3	15,232293	2,483783
Bearing 4	20,295604	3,725114

Table 3.2: Kustosis and Skewness values NASA

An LSTM autoencoder model was developed to detect anomalies in the data from the NASA dataset.The Table 3.3 shows the architecture of the model.

Model architecture				
Encoder	Bottleneck	Repeat	Decoder	Output
LSTM neurons=4 (inputs)	Dense neurons = 2 activation='sigmoid' (encoder)	RepeatVector (bottleneck)	LSTM neurons=4 return sequence=True (repeat)	TimeDistributed (Dense layer)(decoder)

Table 3.3: Model architecture NASA

1. LSTM - This is a definition of an LSTM layer with 4 neurons that receives the input data.

2. Dense - A dense layer, also known as a fully connected layer, is a neural network layer where each neuron is connected to every neuron in the previous layer. In this case, the dense layer has 2 neurons and is connected to all neurons in the encoder.The dense layer is also known as a "bottleneck layer" because it reduces the dimensionality of the data. This means that the dense layer compresses the data into a smaller representation, while still preserving the most important information.The sigmoid activation function is used in the dense layer to squash the output values into the range [0, 1]. This activation function is often used in the bottleneck layer of autoencoders to produce values that can be interpreted as probabilities.

3. RepeatVector -The RepeatVector layer is a neural network layer that repeats the input data n times, where n is a parameter that is specified during training. In other words, the RepeatVector layer replicates the output of the bottleneck layer multiple times.The RepeatVector layer is often used to prepare the output of the bottleneck layer for further processing,

especially when the goal is to generate sequences or perform tasks where the same compressed information needs to be applied at multiple time steps.The RepeatVector layer is a powerful tool that is often used in autoencoders, sequence-to-sequence models, and other types of neural networks that need to process or generate sequences of data.

4. LSTM - The LSTM layer is a type of recurrent neural network layer that is well-suited for processing sequential data. In this case, the LSTM layer has 4 neurons and the return sequence argument is set to True. This means that the LSTM layer will produce output sequences for each time step of the input sequence, rather than just a single output for the entire sequence.This setting is useful for tasks such as sequence generation, machine translation, and anomaly detection.If the LSTM layer is used to detect anomalies in a sequence of data, then the output sequence will be a sequence of scores that indicate the likelihood of an anomaly at each time step.

5. TimeDistribuited - The TimeDistributed layer is a wrapper layer that allows a layer to be applied to each time step of a sequence independently. In this case, the TimeDistributed layer is wrapping a Dense layer. This means that the TimeDistributed layer will apply the Dense layer to each time step of the input sequence, producing a sequence of outputs.The TimeDistributed layer is often used to apply a layer to the output of a recurrent neural network layer, such as an LSTM layer. This is because recurrent neural network layers typically produce sequences of outputs, and the TimeDistributed layer allows us to apply the same layer to each time step of the sequence.

3.2.2 Metro Porto - Data Description

A public transportation service in Porto, Portugal collected a dataset in 2022 between January and June. The dataset contains over 10 million rows of data on analogue signals (pressure, temperature, current consumption), digital signals (control and discrete signals), and GPS information (latitude, longitude, speed).
The Table A.1 shows all sensors in the dataset, along with their functions.
The dataset contains real data, and most of the anomalies are known from the Metro do Porto maintenance reports. To evaluate the model's ability to detect anomalies in this dataset, we will only use the data from the analog sensors.

3.2.2.1 Data Analysis

Table 3.4 shows a summary of the digital sensors. Some sensors may have a large amplitude between the minimum and maximum values. This large amplitude may indicate information that is outside of the normal range and may be considered outliers. Outliers are often a warning sign that an anomaly may be occurring.

Information Resume								
Type	TP2	TP3	H1	DV pressure	Reservoirs	Oil temperature	Flowmeter	Motor current
mean	1,15218	8,97461	7,75142	-0,02454	1,56505	67,30720	20,39515	2,38318
std	3,07530	0,70070	3,05145	0,14866	0,09016	5,38385	3,74361	2,19338
min	-0,03000	0,00600	-0,03400	-0,03800	1,35000	13,87500	18,83472	-0,01250
25%	-0,00800	8,48400	8,23200	-0,03200	1,47000	63,67500	19,01225	0,00250
50%	-0,00800	8,98400	8,74600	-0,02800	1,59000	68,32500	19,04028	3,70500
75%	-0,00600	9,49200	9,29000	-0,02600	1,63800	71,07500	19,25519	3,83750
max	10,87600	10,40800	10,41400	8,32600	2,05400	97,90000	43,07241	9,68500

Table 3.4: Information Resume Metro digital sensors

Table 3.5 shows the kurtosis and skewness values of the analog sensors. The very high kurtosis values in some of the sensors indicate that there are data points with very large peaks compared to the normal distribution. The elevated kurtosis values of TP3, DV pressure, Reservoirs, and Oil Temperature suggest that the information from these sensors may have an anomaly.The high skewness values in some sensors also indicate that there may be information about an anomaly in these sensors. The kurtosis and skewness values suggest that there may be anomalies in the data from the TP3, DV pressure, Reservoirs, and Oil Temperature sensors.

Distribuition of analogic sensors		
Sensor	Kurtosis	Skewness
TP2	3,699159	2,336042
TP3	86,866325	-7,416115
H1	2,673242	-2,080311
DV pressure	299,751634	-3,666856
Reservoirs	247,002506	-15,179812
Oil temperature	109,933743	-8,386621
Flowmeter	5,779556	1,790458
Motor current	-1,281359	-0,045124

Table 3.5: Kustosis and Skewness values Metro Digital sensors

Table 3.6 shows the kurtosis and skewness values of the digital sensors. These sensors only assume binary values, so they will not be considered in the model for detecting anomalies, even though they have relatively high kurtosis and skewness values.

Distribuition of digital sensors		
Sensor	Kurtosis	Skewness
COMP	2,7778	-2,1858
DV eletric	2,9740	2,2303
Towers	9,9058	-3,4505
MPG	2,7778	-2,1858
LPS	157,2456	12,6193
Pressure switch	0,0000	0
Oil level	3659845	1913,0726
Caudal impulses	679,3841	26,1033

Table 3.6: Kustosis and Skewness values Metro analogue sensores

The architecture of the model used for the Metro dataset is the same as the one used for the NASA dataset, with the exception of the number of neurons in the input and output layers. The NASA dataset used 4 neurons in each layer, while the Metro dataset used 8 neurons.

Model architecture				
Encoder	Bottleneck	Repeat	Decoder	Output
LSTM neurons=8 (inputs)	Dense neurons = 2 activation='sigmoid' (encoder)	RepeatVector (bottleneck)	LSTM neurons=8 return sequence=True (repeat)	Time- Distributed (Dense layer) (decoder)

Table 3.7: Model architecture Metro

In the next section, I will present the results of the model we created on the two datasets. I will also use SHAP and LIME to explain the predictions identified as anomalies, and I will analyze whether the explanations given by the two methods for the same point coincide. I will present the results in a clear and concise manner, using tables, graphs, and figures as needed.

3.3 Evaluation protocol

To ensure the accuracy of the model's results, it is necessary to evaluate the model. This can be done using an evaluation protocol, which is a set of rules that specify how to train and test the model, and how to interpret the test results. Evaluation protocols also help to ensure that the evaluation results are meaningful and can be used to improve the model.

One simple but effective evaluation protocol is the holdout method. This is a type of train-test split where the data is divided into two sets: the training set and the test set. The training set is used to train the model, and the test set is used to evaluate the model.

The two datasets used in the dissertation had different sizes, so the way the data was split for testing and training was also different.
For the NASA dataset, the first three days of data were used for training, and the remaining data were used for testing.

The Metro dataset contained a larger volume of data than the NASA dataset, so a different approach was used to separate the data for training and testing. The first month of data was used for training, and the remaining data was used for testing.

There are several evaluation measures for different types of models. Measures such as accuracy, F1-score, and recall are used for classification models, such as the one implemented in this dissertation.

Confusion matrices can also be used to evaluate models. A confusion matrix is a table that summarizes the performance of a classification model. It shows the number of instances that were correctly classified and the number of instances that were incorrectly classified.

A confusion matrix is a table that shows the performance of a classification model. It has two rows and two columns. The rows represent the actual class of the data points, and the columns represent the predicted class of the data points.The confusion matrix can be used to calculate a number of different performance metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare the performance of different models or to track the performance of a model over time.

Confusion Matrix	
True Positives	False Positives
False Negatives	True Negatives

Accuracy is a performance metric for classification models. It is calculated as the number of correct predictions divided by the total number of predictions.Accuracy is a simple and intuitive metric, but it is important to note that it can be misleading in some cases,for example, if a dataset is imbalanced, meaning that there are many more instances of one class than another, then a model can achieve high accuracy by simply predicting the majority class for all instances.

In such cases, it is more informative to use other metrics, such as precision, recall, and F1 score, which take into account the imbalance in the dataset.
Accuracy is a widely used metric for evaluating the performance of classification models, but it is important to be aware of its limitations.

Accuracy = (True Positives + True Negatives) / (Total Instances)

F1 score is a measure of a model's accuracy that considers both the precision and recall of the model. Precision is the fraction of positive predictions, while recall is the fraction of actual positives that were predicted positive. The F1 score is calculated as the harmonic mean of precision and recall. It is a more balanced measure of accuracy than either precision or recall alone.F1 score takes into account both how well the model is able to identify positive instances and how well it is

able to avoid identifying negative instances as positive. This makes it a more informative metric for evaluating the performance of a model, especially when the dataset is imbalanced.
The F1 score ranges from 0 to 1, with a higher score indicating better performance. A perfect score of 1 means that the model is able to identify all positive instances without any false positives or false negatives. The F1 score is a widely used metric for evaluating the performance of classification models in a variety of fields, including machine learning, natural language processing, and computer vision.

F1 = 2 * (precision * recall) / (precision + recall)

Recall is calculated as the number of true positives divided by the total number of positive instances. In other words, it is the proportion of all actual positives that were correctly identified by the model.Recall is an important metric for evaluating the performance of classification models, especially when it is important to identify all positive instances, even if it means making some false positives.Recall is often used in conjunction with other metrics, such as precision and F1 score, to provide a more comprehensive evaluation of the performance of a classification model.

Recall = True Positives / (True Positives + False Negatives)

The metrics of accuracy, precision, and recall can only be used to evaluate the performance of the model on the Metro dataset. This is because the Metro dataset contains information about the moments when anomalies occurred, which can be used to create labels for the records. This allows for the effective validation of the model's performance.

The NASA dataset does not include information about anomalies. This means that it is impossible to create labels for the records, making it impossible to use the metrics of accuracy, precision, and recall to evaluate the model's performance.

Chapter 4

Results and Discussion

4.1 Results

At the end of this chapter, three questions must have a response:

1. Is it possible to detect anomalies in two different datasets?
2. The anomalies detected can be explained?
3. Different explaining methods can give the same result?

4.1.1 Case study 1 - NASA Bearing dataset

4.1.1.1 Model training

To develop the model, the dataset was split into training and test sets. The training set was used to train the model, and the test set was used to evaluate the model's performance. The training set represents the normal operating conditions of the sensors, while the test set contains the sensor readings that lead to bearing failure.The train-test split helps to ensure that the model is not overfitting to the training data. Overfitting occurs when the model learns the training data too well and is unable to generalize to new data. By evaluating the model on the test set, we can get a more accurate estimate of how well the model will perform in real-world conditions.

4.1.1.2 Nasa Model training

Figure 4.1 shows the sensor data from the four sensors under normal operating conditions. It is not possible to identify any anomalous changes in the data that could indicate the presence of an anomaly.

Figure 4.1: Training data NASA

Figure 4.2 shows the sensor data from the four sensors under test conditions. In this data sample, it is possible to see a change in the data. Several factors could cause this change, such as a change in the environment, a change in the equipment, or a human error.

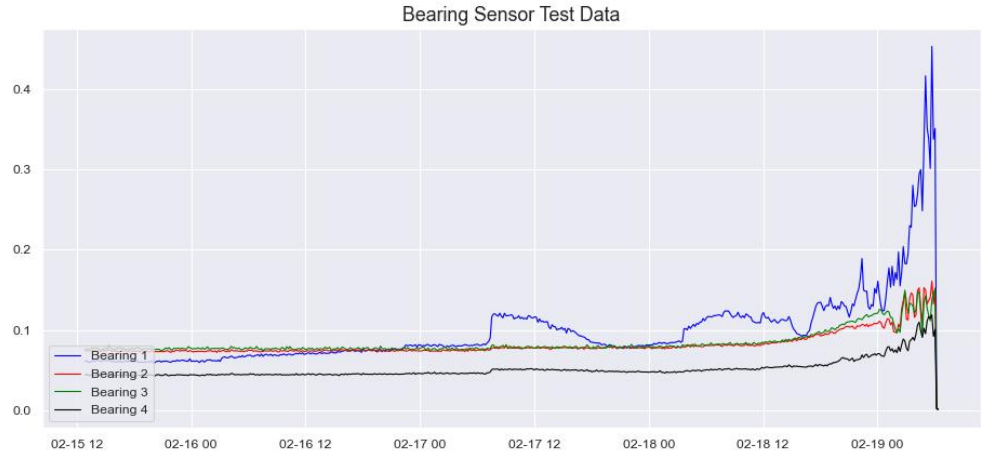


Figure 4.2: Test data

In Figure 4.3 is possible to see the frequency range of the bearing sensors when they are working in normal conditions. The frequency range of the bearing sensors is typically 20 kHz. The lower frequencies are associated with the rotating components of the bearing, while the higher frequencies are associated with the vibrations caused by defects in the bearing.

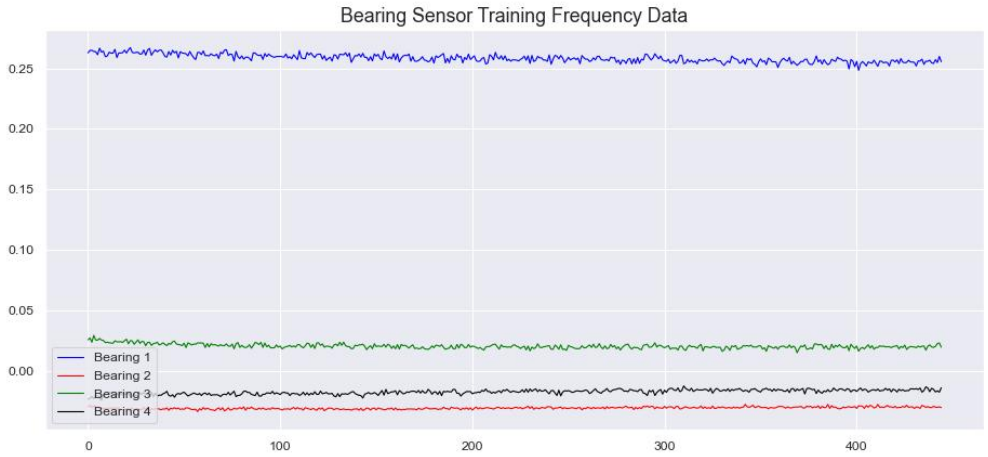


Figure 4.3: Training frequency NASA

The frequency data in Figure 4.4 shows a frequency change, which indicates that there is an anomaly in the data. The sudden increase in frequency suggests that the bearings may be starting to fail. This increase in frequency is caused by the defects in the bearings, which are causing the bearings to vibrate at a higher frequency.

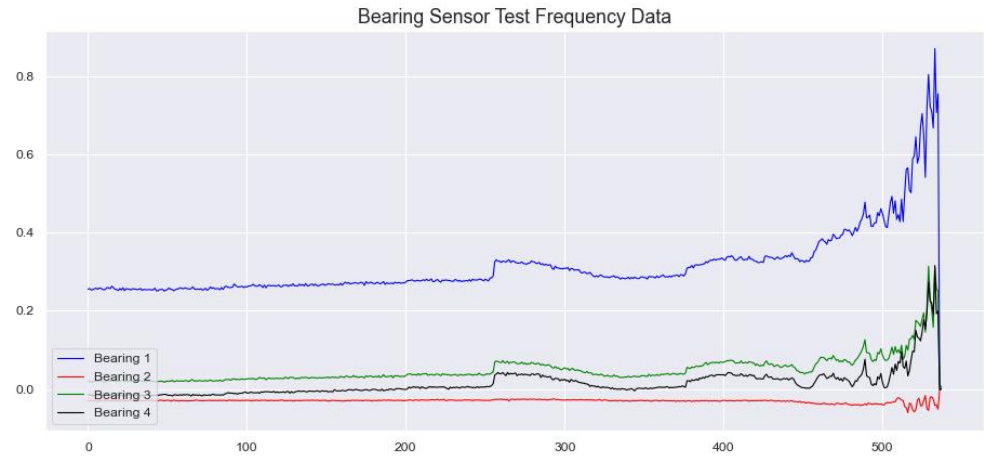


Figure 4.4: Test frequency NASA

The LSTM model is trained to learn how the bearing sensors behave in normal conditions using the training data. This knowledge is then used to detect anomalies in the test data.The model first learns the normal behavior of the bearings by being fed the training data of the normal working conditions. The model learns to identify the patterns associated with normal working conditions.Once the model has learned the normal patterns of the bearings, it can be used to detect anomalies in the test data. When the test data is fed to the model, the model predicts whether the

data is normal or an anomaly.
In the training phase, the model achieved an accuracy of 94.95 % and a loss value of 5 %. A high accuracy and a low loss indicate that the model is performing well in the training phase, meaning that it is learning to make accurate predictions and is not overfitting the data.

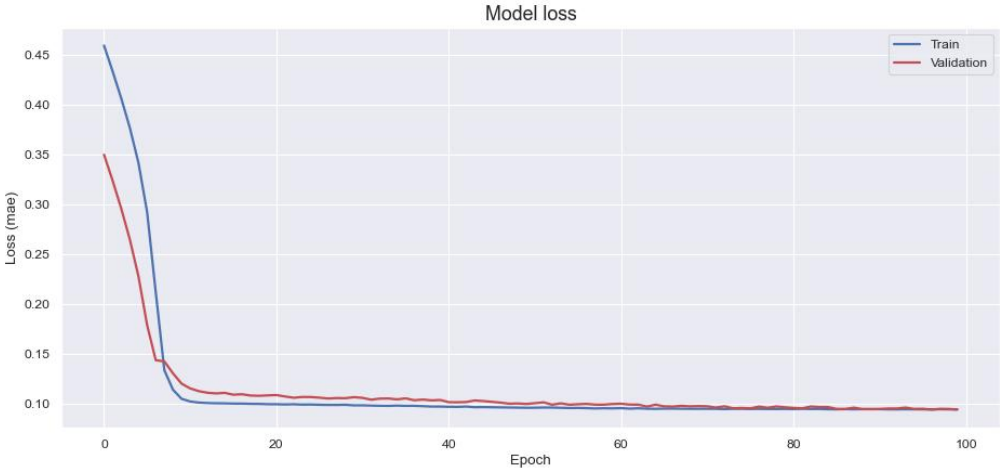


Figure 4.5: Model Loss NASA

Figure 4.6 shows the change in the accuracy of the model on the training and test data over time. The model's accuracy increases on both datasets, indicating that it is not overfitting the training data. The results demonstrate that the model performs well and does not overfit the training data.

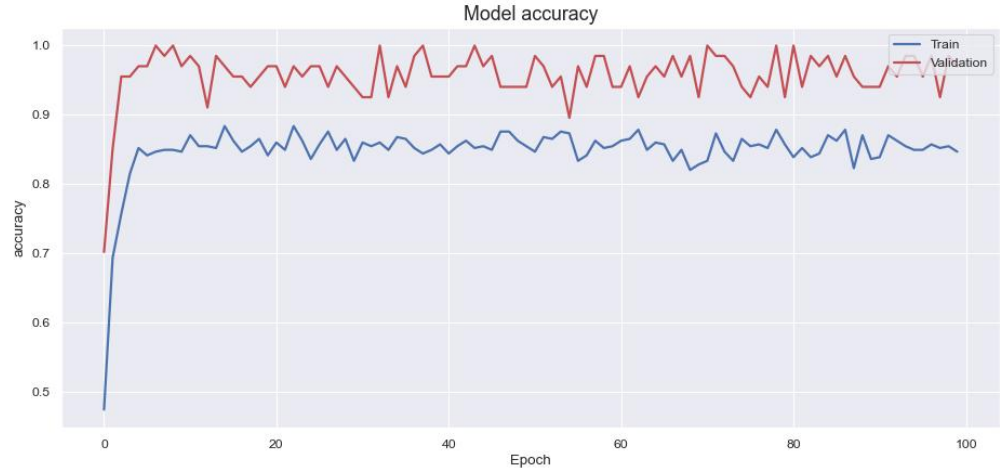


Figure 4.6: Model accuracy NASA

Figure 4.7 shows how the loss distribution changes during model training. The loss values vary during training, but as the model learns to make better predictions, the loss distribution becomes narrower.

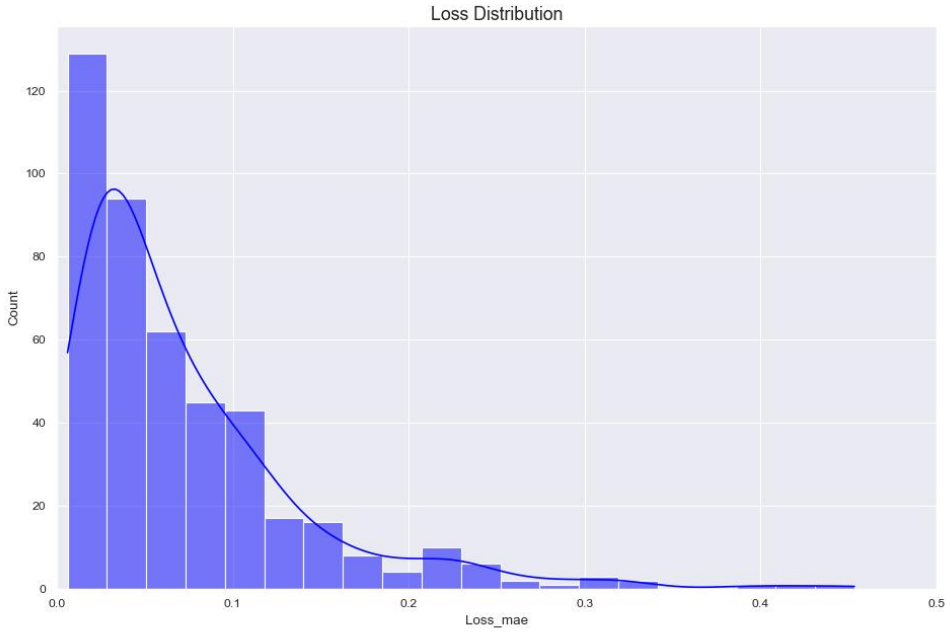


Figure 4.7: Loss distribution NASA

Figure 4.8 shows the results of the model. The model was trained successfully and was able to detect the anomaly when the normal operation of the sensors was interrupted. The maximum value of the mean absolute error (MAE) loss was used to determine when the anomaly in the data started. In this model design, the value was 0.475538.

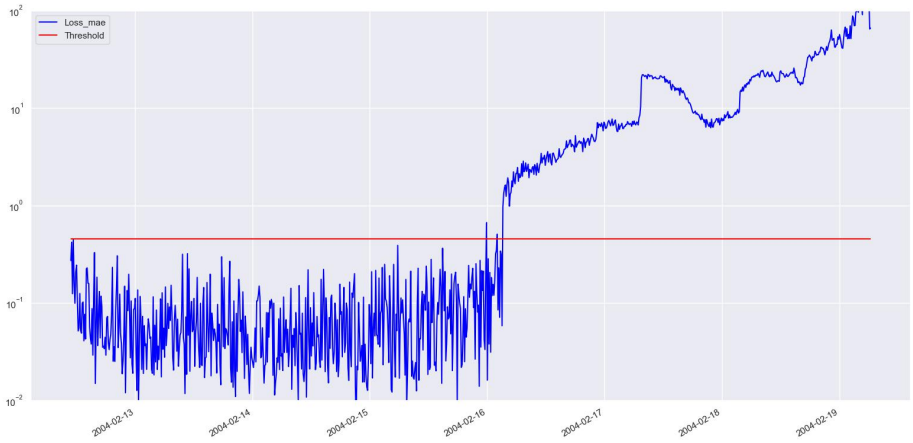


Figure 4.8: Anomaly detection NASA

Tables A.3 and A.4 show the variation in the frequency of data from all components. Table A.3 shows the data frequency when all components are functioning perfectly, while Table A.4 shows the data frequency and the moments that the model considers anomalies.

Table 4.1 shows the reconstruction errors for each feature. The reconstruction error is a measure of how well the model can reconstruct the original data from the latent representation. A lower reconstruction error indicates that the model is able to learn a more accurate representation of the data.

Reconstruction error			
Bearing 1	Bearing 2	Bearing 3	Bearing 4
306.867978	23.898428	25.355984	176.139333

Table 4.1: Reconstruction error NASA

The model evaluation metrics for this dataset are not available because the available information does not identify the anomalies in the dataset. This means that it is not possible to quantify the performance of the model or to identify any areas where it needs to be improved.
Without model evaluation metrics, it is difficult to know how well the model will generalize to new data or to assess its reliability in real-world applications. Additionally, without ground truth labels, it is impossible to determine whether the model is actually detecting anomalies or simply flagging normal data as anomalous.

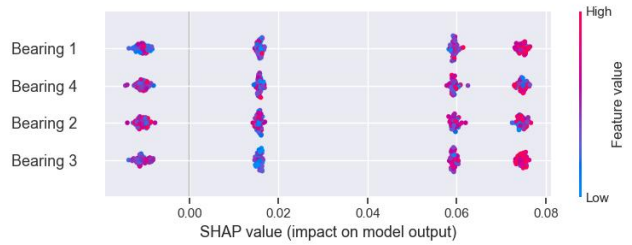
4.1.1.3 Explainability of case study 1

SHAP (SHapley Additive exPlanations) and LIME are two methods that can be used to determine the feature importance in a test model. SHAP is a more general method that can be used to explain how features affect the model's predictions for any type of machine learning model.

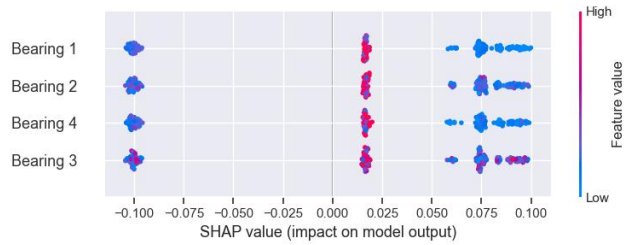
SHAP is a more general method than LIME, and it can be used to explain any type of machine learning model. LIME is specifically designed for explaining local behavior, which means that it can be used to explain individual predictions but not overall model behavior.
Both SHAP and LIME are valuable tools for understanding machine learning models and for identifying the features that are most important for making accurate predictions. The best method to use will depend on the specific application and the desired level of explanation.

Figures 4.9a and 4.9b show the importance of each feature in the model's prediction, according to SHAP. SHAP is a method for explaining how features affect the model's predictions.

Before the anomaly, all features had similar importance. This means that the model was using a combination of all features to make predictions. However, during the anomaly, Bearing 3 became the most important feature. This means that the model was relying heavily on Bearing 3 to detect the anomaly.This is likely because Bearing 3 was the feature that changed the most during the anomaly. The model learned that changes in Bearing 3 are a strong indicator of an anomaly, so it gave Bearing 3 more importance in its predictions.



(a) NASA summary feature importance by SHAP before anomaly



(b) NASA feature importance by SHAP in the anomaly

Figure 4.9: NASA summary feature importance by SHAP

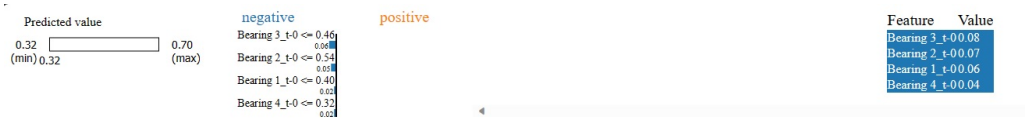
Analysis of a moment that was not classified as an anomaly (Figure 4.10a and Figure 4.10b) shows that there is no significant difference between the importance of the features in the model's decision-making. Similarly, analysis of a moment in which an anomaly was identified (Figure 4.9a and Figure 4.9b) also does not allow to accurately state which feature most contributes to the identified anomaly.
The fact that it is not possible to accurately state which feature most contributes to an anomaly in either of these cases suggests that the anomalies are caused by a combination of factors. It is also possible that the anomalies are caused by features that are not included in the model.

(a) NASA normal feature importance by SHAP force

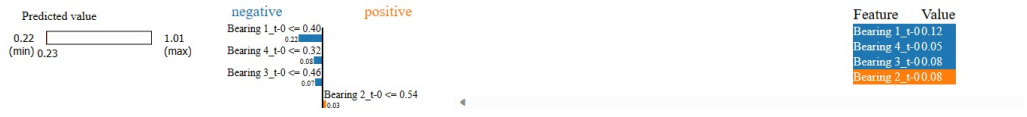
(b) NASA anomaly feature importance by SHAP force

Figure 4.10: NASA feature importance by SHAP

Before the anomaly occurs, LIME and SHAP both give equal importance to all features. However, when analyzing a moment identified as an anomaly, LIME indicates that Bearing 2 is the feature that most contributes to the model's prediction, while SHAP does not. This suggests that LIME and SHAP may have different strengths and weaknesses in explaining anomaly detection models.The results suggest that it is important to use multiple methods to explain anomaly detection models. This can help to get a more complete understanding of how the model is making predictions and to identify the features that are most important for anomaly detection.



(a) NASA feature importance by LIME before anomaly



(b) NASA feature importance by LIME in anomaly

Figure 4.11: NASA feature importance by LIME

4.1.2 Case study 2 - Porto Metro dataset

In 2022, a dataset was collected to evaluate machine learning methods for online anomaly detection and failure prediction. The dataset contains analogue sensor signals (pressure, temperature, and current consumption), digital signals (control signals and discrete signals), and GPS information (latitude, longitude, and speed). The data was logged at 1 Hz from the APU unit by an onboard device and sent to a remote server every 5 minutes using a GSM network.

Only the digital sensor data from the Metro dataset will be used in the anomaly detection model. This is because digital sensor data is more reliable and consistent than analog sensor data. Digital sensor data is also easier to process and provides more information about the system being monitored, which is important for anomaly detection models.

4.1.2.1 Model Training

Figure 4.12 shows the digital sensors working in normal conditions. The sensor data is consistent, indicating that the sensors are providing reliable and accurate information.The consistency of the sensor data is important for anomaly detection. If the sensors are providing consistent data, then any significant changes in the data can be flagged as anomalies. This can help to identify potential problems with the system being monitored.

Figure 4.12: Metro training data

Figure 4.13 shows two disruptions in the data. These disruptions may indicate that the data contains one or more anomalies. Anomalies can be caused by a variety of factors, such as system failures, data corruption.

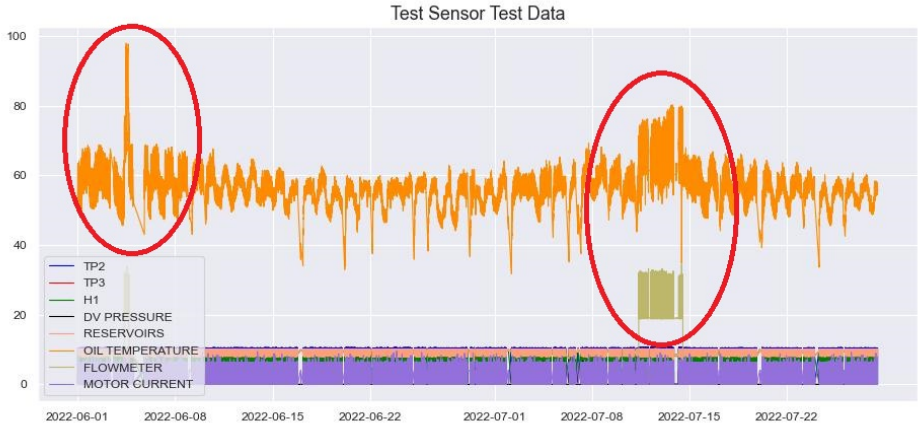


Figure 4.13: Metro test data

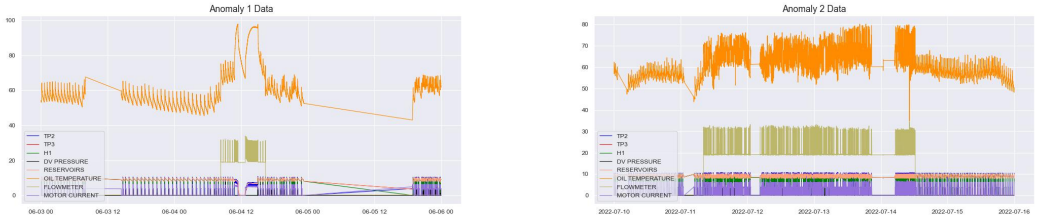
Figure 4.14 shows that the frequency of the data is relatively constant when the sensors are working in normal conditions. This means that the sensors are not producing any unusual readings.

Figure 4.14: Metro train frequency data

The two disruptions in the data identified in Figure 4.13 can also be seen in Figure 4.15. One of the disruptions is smaller than the other. The first disruption lasts a few hours or perhaps a day, while the second disruption lasts more than a day.

Figure 4.15: Metro test frequency data

Figures 4.16a and 4.16b show the variation in the data from normal to abnormal working conditions. The data in Figure 4.16a shows a sudden increase in frequency, while the data in Figure 4.16b shows a sudden decrease in frequency. These changes in frequency could be caused by a variety of factors, such as system failures, data corruption



(a) Metro first anomaly

(b) Metro second anomaly

Figure 4.16: Metro anomalies

The LSTM model used to identify the two disruptions in the NASA dataset (Table 3.7) was adapted to the Metro dataset by changing the number of neurons in the input and dense layers to account for the different number of features in the two datasets. The model achieved over 90 % accuracy and less than 1 % loss on the training set, and over 84 % accuracy and less than 1 % loss on the test set. This indicates that the model is able to learn to identify anomalies in the METRO dataset with high accuracy.

The fact that the model achieved high accuracy and low loss on both the training and test sets indicates that the model is able to learn to generalize to new data. This is important for anomaly detection models, as they need to be able to detect anomalies in data that they have never seen before.

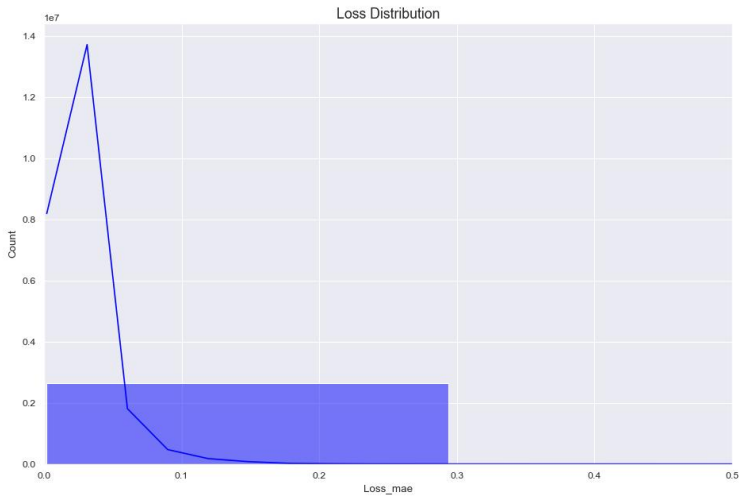


Figure 4.17: Metro loss distribution

The 95th percentile threshold was used to detect possible anomalies in the data. This means that only data points that are more extreme than 95 % of the data are considered anomalies. This approach helps to reduce the risk of false positives (identifying normal data as anomalies) and false negatives (failing to identify anomalies).

Figure 4.18 shows the threshold used to detect anomalies in the Metro data. The two disruptions shown in Figure 4.13 are the data points that are above the threshold. This means that the threshold is correctly identifying the two disruptions as anomalies.

The first anomaly corresponds to a shorter period, lasting only a few hours. It is visible as a sharp spike in the frequency data. The second anomaly has a much longer period, lasting several days. This suggests that the first anomaly may have been caused by a temporary event.The second anomaly may have been caused by a more permanent problem, such as a hardware failure.
The fact that the threshold is correctly identifying the two disruptions as anomalies suggests that the threshold is well-tuned and that the anomaly detection model is effective

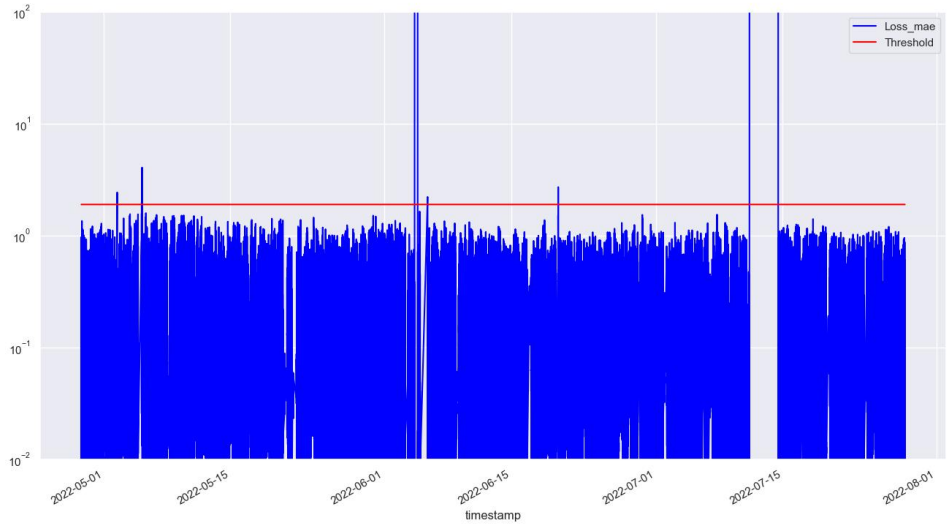


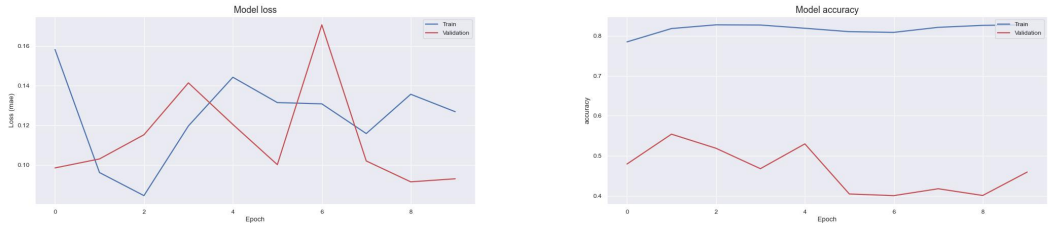
Figure 4.18: Metro anomaly detection

Table A.5 shows an example of the sensors working in perfect conditions. Table A.6 shows information about each sensor classified as an anomaly in the model. The first four lines refer to the first anomaly, and the rest to the second anomaly.The model predicted that the first fault started on June 4th and lasted for 8 hours, 3 minutes, and 6 seconds. The model predicted that the second anomaly occurred over three days and 4 hours, 11 minutes, and 46 seconds, starting on July 11th and ending on July 14th.Comparing the values of the sensors in the two tables with regard to the first anomaly, it is possible to see that there is a great difference in the data for the TP2 sensor. The TP2 sensor is the sensor that registers the compressor pressure. Variations in the data recorded in sensors connected to air pressure may indicate an air leak or a malfunction in one of the components connected to the airflow.

The existence of a change in the data recorded by the Oil temperature. This sensor records oil temperature information. Analysing the change in data recorded by the sensor, it can be seen that this second anomaly is linked to air pressure. The pressure difference could be due to an air leak. This information is available in the appendix on the table A.5 and on the table A.6.

The anomaly detection model was trained using a Long Short-Term Memory (LSTM) neural network. An LSTM neural network is a type of recurrent neural network (RNN) that is well-suited for tasks such as anomaly detection, which involve learning long-term dependencies in the data.

The training data consisted of 2,658,868 records from the first month of operation, with eight features and no anomalies. This means that the model was trained only on normal data, which is important for anomaly detection models.Figures 4.19a and 4.19b show the evolution of the model during the training and validation process. The training loss and validation loss gradually decrease over time, while the training accuracy and validation accuracy gradually increase over time. This suggests that the model is learning to identify anomalies in the data..



(a) Metro model loss

(b) Metro model accuracy

Figure 4.19: Metro model history

The anomaly detection model uses a threshold to distinguish between normal and anomalous data points. The threshold is set to the 95th percentile of the loss value, which means that only data points with a loss value greater than 95 % of the other data points are considered anomalies.In this dataset, the threshold was calculated to be 1432.459645. This means that any data point with a loss value greater than 1432.459645 is considered an anomaly.newline

The reconstruction error value was also calculated for each of the features. This is a measure that can also be used to detect anomalies. Data points with a high reconstruction error are considered to be anomalies.Reconstruction error is a measure of how well a machine learning model can reconstruct the input data. It is calculated as the difference between the original input data and the reconstructed output data.In the context of anomaly detection, reconstruction error can be used to identify anomalies by flagging data points that have a high reconstruction error. This is because anomalies are more likely to be different from the rest of the data, and therefore have a higher reconstruction error.

Reconstruction error							
TP2	TP3	H1	DV pressure	Reservoirs	Oil temperature	Flowmeter	Motor current
2.36145	4.47880	6.30866	1.220247	4.478699	1.066828	943107.1	1.963598

Table 4.2: Reconstruction error Metro

The anomaly detection model is evaluated using standard metrics for anomaly detection, such as accuracy, precision, and recall. The table shows that the model has a very high accuracy in detecting the anomalies in the dataset, meaning that it is able to correctly identify most of the anomalies.Existing anomalies in the dataset have been identified in other anomaly detection works that used the same data.The created model detects anomalies two hours before they were initially documented and continues to detect them up to two hours after. This means that the model is able to detect anomalies early, and that it is able to continue to detect them even after they have been initially documented.

Model evaluation formulas					
Accuracy	Precision	Recall	f1 score	false positive rate	false negative rate
(tp+tn)/ (tp+fp+fn+tn)	tp/ (tp +fp)	tp /(tp+fn)	((precision* recall)/ (precision+recall)) *2	fp/(fp+tn)	fn/(fn+tp)

Table 4.3: Model evaluation Metro

Model evaluation					
Accuracy	Precision	Recall	f1 score	false positive rate	false negative rate
0.989676	0.878654	0.9206746	0.899174	0.006692	0.079326

Table 4.4: Model evaluation Metro

The confusion matrix in Figure 4.20 shows that the model accurately detects anomalies. A confusion matrix is a table that shows the performance of a classification model. It compares the model's predictions to the actual labels of the data.
In the confusion matrix the number of true positives and true negatives is much higher than the number of false positives and false negatives. This means that the model is able to correctly identify most of the anomalies in the data, and that it is also able to correctly identify most of the normal data.

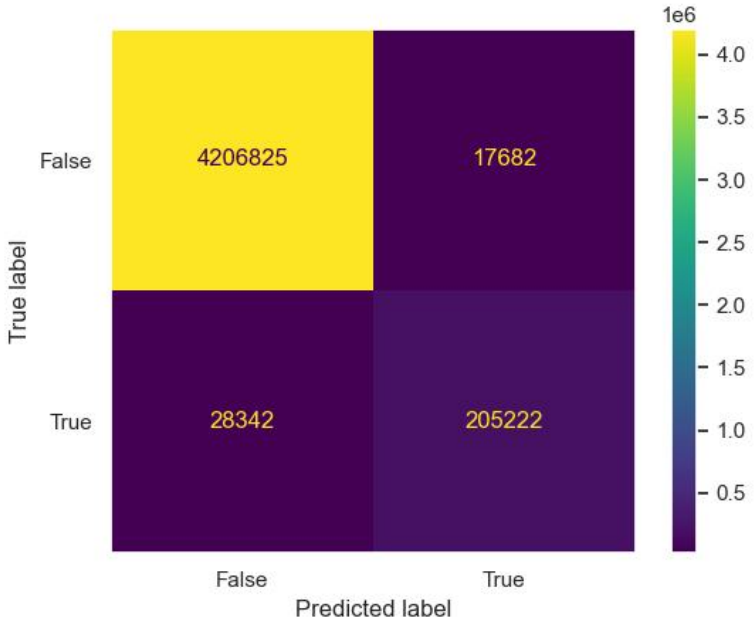


Figure 4.20: Metro confusion matrix

4.1.2.2 Explainability of case study 2

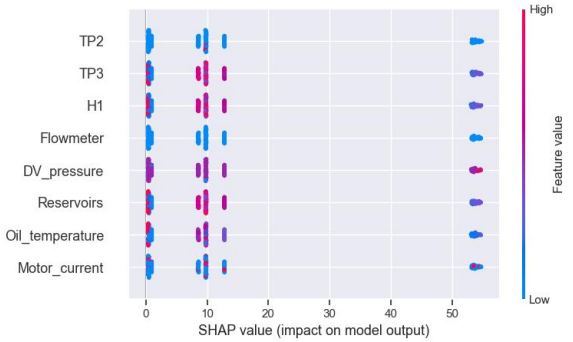
The importance of each feature in the model's decision-making process on a data set will be analysed. The analysis will be performed on the importance of SHAP and LIME attributes to each feature for moments before the first anomaly is detected, during the first anomaly, and after the anomaly.

SHAP, or Shapley Additive explanations, is a method for explaining the predictions of machine learning models. SHAP works by calculating the contribution of each feature to the model's prediction.SHAP calculates the SHAP value of each feature. The SHAP value is the sum of its marginal contribution over all possible feature values.

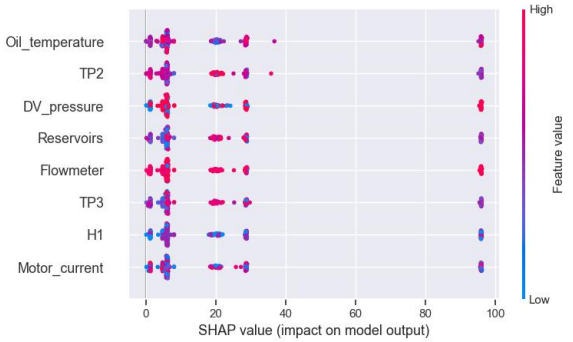
LIME, or Local Interpretable Model-Agnostic Explanations, is a technique for explaining the predictions of black box machine learning models. LIMe works by creating a simple, interpretable model that approximates the behaviour of the black box model around a specific data point. LIME starts by creating a simple model, such as a linear regression model, that approximates the behaviour of the black box model. LIME calculates how much the feature contributes to the simple model's prediction for each feature. These contributions are called attribution scores.

In all the model explanation, two hundred records were used. This was done to provide more information to the explainer, giving more support for the feature measure.Anomaly detection models are often trained and evaluated on large datasets. However, when explaining the model to a human, it can be helpful to use a smaller subset of the data. This is because it can be easier for a human to understand the model's predictions when they are presented with a smaller number of data points.Using two hundred records for the model explanation is a good compromise between providing enough information to the explainer and keeping the explanation concise and easy to understand.

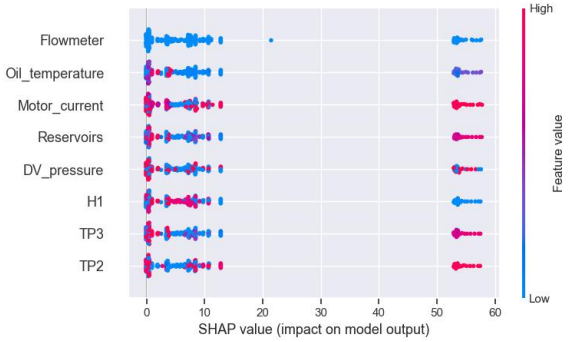
Before the first anomaly 4.21a, the DV pressure and Reservoirs sensors are the most important for SHAP. During the anomaly 4.21b, H1 becomes the most important sensor, followed by DV pressure, Oil temperature, and Reservoirs. After the anomaly 4.21c, H1 remains the most important sensor, followed by Motor current.



(a) Metro feature importance before the first anomaly



(b) Metro feature importance in anomaly first anomaly

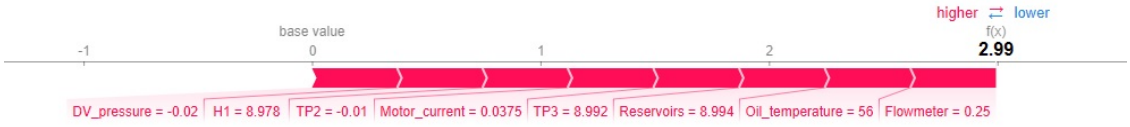


(c) Metro feature importance after first anomaly

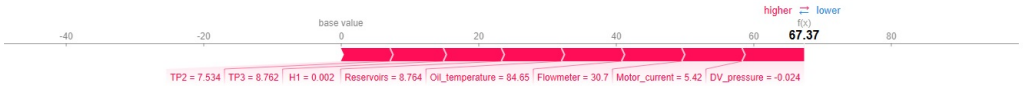
Figure 4.21: Feature importance on anomaly first anomaly

The importance of features can change over time, so it is important to analyse the importance of each feature when a single data point is selected from each of the three moments that were previously analysed.

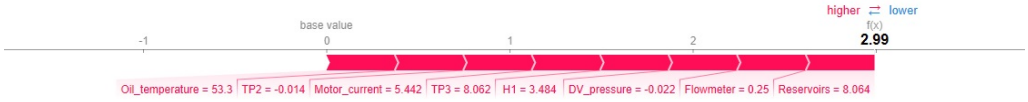
According to the SHAP analysis, all features contribute equally to the model's result before the first anomaly occurs. However, during the anomaly, the Oil temperature is the most important feature, followed by the flowmeter and the reservoirs. After the anomaly ends, the importance of each feature for the model's decision changes. When the anomaly ends, the most important feature becomes DV pressure, followed by TP2.



(a) Metro feature importance before the first anomaly



(b) Metro feature importance in anomaly first anomaly



(c) Metro feature importance an anomaly first anomaly

Figure 4.22: Feature importance by SHAP in the first anomaly

Before the first anomaly occurs, the LIME analysis of the same data points as the SHAP analysis indicates that Reservoirs is the feature that has the highest positive contribution to the model's predictions. This means that the model is using the Reservoirs feature to predict that the data points are normal.When the anomaly is being predicted, the feature with the most positive contribution to the forecast is TP2, followed by Reservoirs. The feature that has the most negative contribution to the forecast is the Oil Temperature. This means that the model is using the TP2 and Reservoirs features to predict that the data point is anomalous, and the Oil Temperature feature to predict that the data point is normal.After the first anomaly, the LIME analysis of the data at that time indicated that Motor current was the feature that had the most positive contribution to the model's prediction. In contrast, TP2 had the greatest negative contribution. This means that the model is using the Motor current feature to predict that the data point is anomalous, and the TP2 feature to predict that the data point is normal.

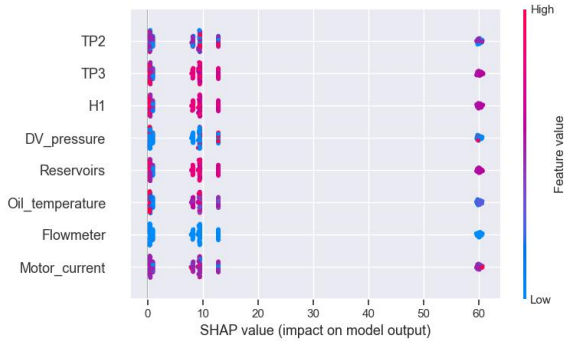
(a) Metro feature importance before the first anomaly

(b) Metro feature importance in anomaly first anomaly

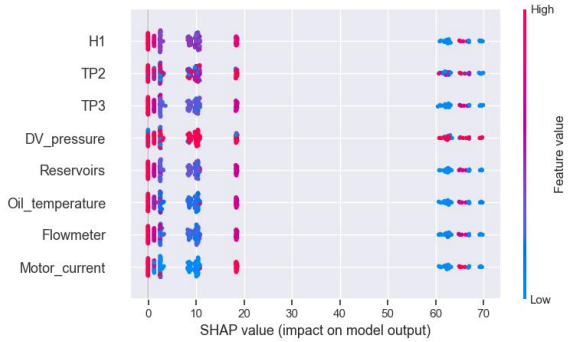
(c) Metro feature importance after the first anomaly

Figure 4.23: Feature importance by LIME in the first anomaly

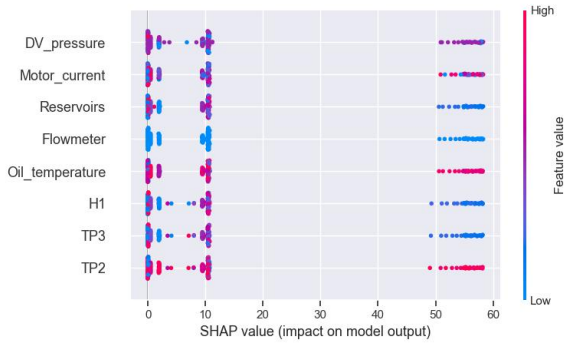
Before the second anomaly, the most important feature for the model's prediction, according to SHAP calculations, was Reservoirs, and the least important was H1. TP3 and flowmeter are the features that SHAP considers to be the most important for the model's predictions. After the date of the second failure, the most important feature for the model's prediction is the Motor current. The importance of each of the features varies over time. The fact that the importance of each feature varies over time suggests that the model is able to learn new features over time. This is an important property for anomaly detection models, as it allows them to adapt to changes in the data and to continue to be effective even as the system evolves.



(a) Metro feature importance before the second anomaly by SHAP



(b) Metro feature importance in the second anomaly by SHAP

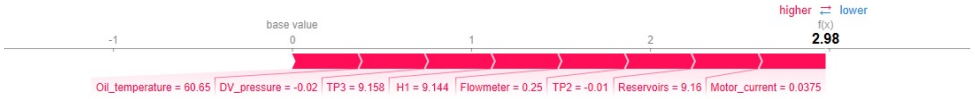


(c) Metro feature importance after second anomaly

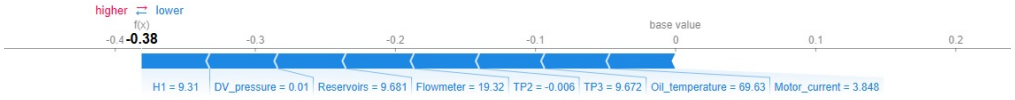
Figure 4.24: Feature importance by SHAP in the second anomaly

By analyzing a specific moment during the second anomaly, we can see that before the anomaly was detected, SHAP considered the Reservoirs feature to be the most important feature for the model's predictions. At one point during the second anomaly, the TP2 feature was considered to have the greatest influence on the model's predictions. After the fault finished and one of the moments was evaluated, it was verified that H1 was the most important feature.

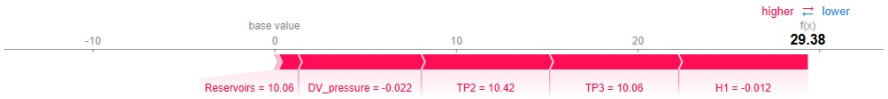
The model was able to learn new features over time. During the second anomaly, the model learns that the TP2 feature is important for anomaly detection. After the anomaly, the model learns that the H1 feature is important for anomaly detection. This suggests that the model is able to adapt to changes in the data, and that it can continue to be effective even as the system evolves.



(a) Metro feature importance before the second anomaly by SHAP



(b) Metro feature importance in the second anomaly by SHAP

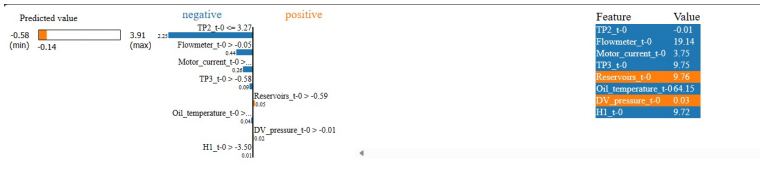


(c) Metro feature importance after second anomaly

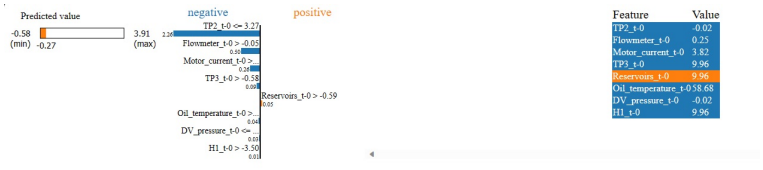
Figure 4.25: Feature importance in a moment by SHAP on the second anomaly

Before the second anomaly was detected in the dataset, LIME analysis showed that Reservoirs had the greatest positive contribution to the model's prediction. This means that the model was using the Reservoirs feature to predict that the data points were normal.When the model predicted a failure, Reservoirs and DV pressure were the features that contributed positively to this prediction. This means that the model was using the Reservoirs and DV Pressure features to predict that the data point was anomalous.When the anomaly was no longer detected, Reservoirs again became the feature that contributed positively to the predictions generated. This means that the model was once again using the Reservoirs feature to predict that the data points were normal.

(a) Metro feature importance before the second anomaly by LIME



(b) Metro feature importance in the second anomaly by LIME



(c) Metro feature importance after second anomaly

Figure 4.26: Feature importance by LIME in the second anomaly

4.2 Discussion

LSTM autoencoders can be used for anomaly detection. They work by learning the normal patterns in a time series and then detecting anomalies as data points deviate from these patterns. The main advantage of using LSTM autoencoders for anomaly detection is the capability of capturing long-term dependencies in time series data. The LSTM autoencoders are relatively easy to train and can be used with various time series. Because of its characteristics, LSTM is a powerful tool for anomaly detection and is well-suited for detecting anomalies that occur over time. It can be used with a large variety of time series data, but it can be computationally expensive to train and sensitive to the choice of hyperparameters.

The use of LSTM autoencoders was able to detect anomalies in two different datasets, even though they had different data types. The only necessary adjustment was to consider the number of features in each dataset.

The successful detection of anomalies in two different datasets answers whether using the same architecture to detect anomalies in different datasets is possible. The results suggest that it is possible to do so, with only minor adjustments to the architecture depending on the specific datasets.

SHAP and LIME are two of the most popular methods for explaining the predictions of a machine-learning model. Both methods create a simplified model that approximates the behaviour of the original model. SHAP uses a game-theoretic approach to calculate the contribution of each feature in the model prediction. LIME, on the other hand, creates a simplified model by perturbing the values of the features and observing how the model prediction changes.

SHAP is considered to be more accurate than LIME, but SHAP can be more difficult to interpret, especially for complex models. LIME is easier to interpret but may not be as accurate as SHAP.

Using two methods to explain the model's predictions allowed us to identify the feature that contributed the most to the model's prediction. Both methods allowed us to identify the most important feature at each moment of analysis. The analysis focused on the moments before and after each anomaly was detected and the importance of each feature during the period the model detected each anomaly.

In the case of the NASA dataset, SHAP indicates that the anomaly is related to Bearing 3, while LIME indicates that the anomaly originates from Bearing 2. However, more information is needed to confirm which methods are correct.

Before the first anomaly in the Metro dataset, the flowmeter and DV Pressure are the most important features for the anomaly detection model. After the second anomaly, the flowmeter loses its importance, and TP2 and TP3 become the most important features. In the second anomaly, TP2 and flowmeter are the most important features, and Oil Temperature and DV pressure are the most important features in the first anomaly.
It was possible to explain which component is behaving abnormally by identifying the feature that most contributes to the model's identification of a given moment as an anomaly. This allows

me to state that there is an explanation for the identified anomaly. Therefore, I can answer the second research question. The methods used to produce explanations produce similar results. The importance of each feature is similar in both methods, LIME is easier to interpret than SHAP. In response to the third question, the explanations given by each of the methods are very similar.

The biggest limitation of this dissertation was the lack of documentation for the NASA dataset. Documentation is important for understanding the data, as it provides information about the features, the data collection process, and any known biases in the data.Without documentation, it is difficult to be sure that the data is accurate and reliable. It is also difficult to understand how the data was collected and what the features represent.This can make it difficult to train and evaluate machine learning models on the data, as it is difficult to know what to expect from the data.In this case, the lack of documentation for the NASA dataset made it difficult to fully understand the data and to train and evaluate the anomaly detection model.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This dissertation's main objective was to use the LSTM autoencoder to detect anomalies in time series data and SHAP and LIME to explain the predictions of the LSTM model. The results showed that LSTM could detect anomalies in the data with high accuracy. SHAP and LIME could explain the predictions of the LSTM model, and LIME was easier to interpret than SHAP.

LSTM detected anomalies in two different datasets, with only minor adjustments to the architecture to account for the number of features in each dataset. This dissertation demonstrated that LSTM is a highly effective tool for anomaly detection. Their learning capacity partly determines the effectiveness of anomaly detection algorithms.

The use of LIME and SHAP to explain the predictions of machine learning models for anomaly detection was very useful because it allowed us to identify the features that most influenced the model's prediction at each moment. Identifying the most important feature at each moment can help understand which component or components may be behaving abnormally. This abnormal behaviour could be an indication that an anomaly is occurring. The results showed that the two methods can produce similar results, but LIME is easier to interpret than SHAP.

This work used two explainability methods to understand the origins of the anomalies detected in two datasets.

In the case of the NASA dataset, the results were inconclusive because SHAP and LIME identified different characteristics as being the most important.

In the case of the Metro dataset, SHAP could not identify the most important characteristic or characteristics in each fault. However, it was possible to identify that some characteristics stood out. For example, in the case of Metro anomaly two, the characteristics that stood out most were linked to the compressed air flow. For anomaly one, one of the most important characteristics was the oil pressure.

5.2 Future Work

For future work, it is necessary to develop new methods that are more robust to noise and outliers or methods that combine the prediction of multiple algorithms.

Many anomaly detection algorithms are computationally demanding and challenging to apply to large datasets. Therefore, developing more efficient algorithms that can analyze large datasets is essential.Anomaly detection algorithms are normally applied to time series data. However, it is important to understand if they can also be applied to other data types, such as images and text.

This was the approach that Yueyue Yao and Weiting Zhang published in 2023. Yao et al.presented a new approach to anomaly detection that captures patterns based on a statistical analysis of the input sequences' discrete wavelet transform (DWT) coefficients. The latent spaces are restricted to reflect the unique patterns of normal sequences in both the time and frequency domains. Additionally, a weight controller is designed to calculate sample-adaptive regularization weights to utilize the regularization effect fully.

Dai and Gao introduced a different approach to anomaly detection called MST-GAT. MST-GAT is a multimodal spatial-temporal graph attention network that uses a multimodal graph network and a convolution network to capture temporal and spatial dependencies. MST-GAT is formulated as a graph structure, with each sensor represented as a node. MST-GAT optimizes reconstruction and prediction targets and identifies anomalies using anomaly scores.

Fuzzy rules are a type of explainability model that is human-friendly, efficient, and can be applied to complex models. Fuzzy logic is a mathematical framework that deals with partial truths and vagueness. It is based on the observation that humans make decisions based on imprecise and non-numerical information. Fuzzy sets are a way of representing this type of information
This system is applied in industrial manufacturing, automatic control, automobile production, banks, hospitals, libraries and academic education Bai and Wang.

Developing new methods for explaining the predictions of anomaly detection algorithms is essential. This would help users understand why an algorithm has flagged a particular data point as anomalous, which could be useful for debugging the algorithm or taking corrective action.
To better understand the inner workings of a machine learning model and the logical reasoning behind its predictions, it is essential to explore other models of explainability and make them more interpretable.

Appendix A

Appendix

Analog Sensors	
Sensor	Description
TP2	Pressure on the compressor (bar)
TP3	Pressure generated at the pneumatic panel (bar)
H1	Valve that is activated when the pressure read by the pressure switch of the command is above the operating pressure of 10.2 bar (bar)
DV pressure	Pressure exerted due to pressure drop generated when air dryers towers discharge the water. When it is equal to zero, the compressor is working under load (bar)
Reservoirs	Pressure inside the air tanks installed on the trains (bar)
Oil Temperature	Temperature of the oil present on the compressor (°C)
Flowmeter	Airflow was measured on the pneumatic control panel (m3/h)
Motor Current	Motor's current, which should present the following values: (i) close to 0 A when the compressor turns off; (ii) close to 4 A when the compressor is working offloaded; and (iii) close to 7 A when the compressor is operating under load (A)
Digital Sensors	
COMP	Electrical signal of the air intake valve on the compressor. It is active when there is no admission of air on the compressor, meaning that the compressor turns off or working offloaded
DV electric	Electrical signal that commands the compressor outlet valve. When it is active, it means that the compressor is working under load; when it is not active, it means that the compressor is off or offloaded
TOWERS	Signal that defines which tower is drying the air and which tower is draining the humidity removed from the air. When it is not active, it means that tower one is working; when it is active, it means that tower two is working
MPG	Is responsible for activating the intake valve to start the compressor under load when the pressure in the APU is below 8.2 bar. Consequently, it will activate the sensor COMP, which assumes the same behaviour as the MPG sensor
LPS	Signal activated when the pressure is lower than 7 bars
Pressure switch	Signal activated when pressure is detected on the pilot control valve
Oil Level	The oil level on the compressor is active (equal to one) when the oil is below the expected values
Caudal impulses	Signal produced by the flowmeter indicating the existence of the flow of air per second
GPS Information	
gpsLong	Longitude position (°)
gpsLat	Latitude position (°)
gpsSpeed	Speed (km/h)
gpsSpeed	Speed (km/h)

Table A.1: Metro sensors

Normal working conditions				
timestamp	Bearing 1	Bearing 2	Bearing 3	Bearing 4
2004-02-12 11:02	0.06146	0.07384	0.08446	0.04508
2004-02-12 11:12	0.06136	0.07561	0.08284	0.04512
2004-02-12 11:22	0.06166	0.07328	0.08488	0.04417
2004-02-12 11:32	0.06194	0.07459	0.08263	0.04466
2004-02-12 11:42	0.06123	0.07417	0.08202	0.04384

Table A.2: NASA Normal working conditions data

Normal working conditions				
timestamp	Bearing 1	Bearing 2	Bearing 3	Bearing 4
2004-02-12 11:02	0.06146	0.07384	0.08446	0.04508
2004-02-12 11:12	0.06136	0.07561	0.08284	0.04512
2004-02-12 11:22	0.06166	0.07328	0.08488	0.04417
2004-02-12 11:32	0.06194	0.07459	0.08263	0.04466
2004-02-12 11:42	0.06123	0.07417	0.08202	0.04384

Table A.3: NASA Normal working conditions data

Anomaly working conditions				
timestamp	Bearing 1	Bearing 2	Bearing 3	Bearing 4
2004-02-17 11:42	0,11517	0,07839	0,07934	0,05185
2004-02-17 11:52	0,11509	0,07753	0,07742	0,05119
2004-02-17 12:02	0,11640	0,07768	0,07743	0,05038
2004-02-17 12:12	0,11624	0,07767	0,07825	0,05178
2004-02-17 12:22	0,10965	0,07785	0,07997	0,05091

Table A.4: NASA anomaly working conditions data

Normal working conditions								
timestamp	TP2	TP3	H1	DV pressure	Reservoirs	Oil temperature	Flowmeter	Motor current
2022-06-04 08:05:43.782	-0,012	9,190	9,176	-0,022	9,194	57,275	0,250	0,040
2022-06-04 08:05:44.773	-0,012	9,190	9,176	-0,022	9,194	57,225	0,250	0,040
2022-06-04 08:05:45.764	-0,014	9,188	9,174	-0,020	9,192	57,200	0,250	0,040
2022-06-04 08:05:46.754	-0,012	9,188	9,176	-0,022	9,192	57,275	0,250	0,038
2022-06-04 08:05:47.745	-0,012	9,188	9,174	-0,020	9,192	57,200	0,250	0,040
2022-07-10 20:36:09,888	-0,012	8,074	8,062	-0,020	8,074	52,700	0,250	0,038
2022-07-10 20:36:10,879	-0,012	8,072	8,060	-0,018	8,074	52,725	0,250	0,038
2022-07-10 20:36:11,870	-0,012	8,072	8,060	-0,020	8,072	52,675	0,250	0,038
2022-07-10 20:36:12,861	-0,012	8,070	8,060	-0,020	8,070	52,725	0,250	0,038
2022-07-10 20:36:13,851	-0,012	8,070	8,058	-0,020	8,072	52,700	0,250	0,038

Table A.5: Metro Normal working conditions data

Anomaly example								
timestamp	TP2	TP3	H1	DV pressure	Reservoirs	Oil temperature	Flowmeter	Motor current
2022-06-04 10:53:00.427	9,03	8,61	0,00	-0,03	8,60	81,78	31,26	5,70
2022-06-04 10:53:01.417	9,04	8,61	0,00	-0,02	8,61	81,75	31,35	5,56
2022-06-04 10:53:02.408	9,04	8,63	0,00	-0,02	8,62	81,88	31,55	5,60
2022-06-04 10:53:03.399	9,04	8,61	0,00	-0,02	8,61	81,78	31,60	5,71
2022-06-04 10:53:04.390	9,04	8,62	0,00	-0,02	8,62	81,85	31,71	5,62
2022-06-04 10:53:05.381	9,05	8,62	0,00	-0,02	8,63	81,93	31,52	5,58
2022-07-11 10:13:40.034	-0,006	9,200	8,922	0,050	9,204	67,900	19,283	3,780
2022-07-11 10:13:41.025	-0,006	9,198	8,922	0,050	9,192	67,825	19,283	3,830
2022-07-11 10:13:42.016	-0,008	9,196	8,920	0,030	9,204	67,875	19,283	3,888
2022-07-11 10:13:43.008	-0,008	9,194	8,918	0,050	9,203	67,975	19,274	3,778
2022-07-11 10:13:43.999	-0,008	9,192	8,918	0,030	9,185	68,000	19,283	3,858
2022-07-11 10:13:44.990	-0,006	9,192	8,914	0,050	9,202	68,075	19,274	3,873

Table A.6: Metro Anomaly data

References

[1] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.

[2] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

[3] Ying Bai and Dali Wang. *Fundamentals of Fuzzy Logic Control — Fuzzy Sets, Fuzzy Rules and Defuzzifications*, pages 17–36. 01 2007. ISBN 978-1-84628-468-7. doi: 10.1007/978-1-84628-469-4_2.

[4] Zaharah Allah Bukhsh, Aaqib Saeed, Irina Stipanovic, and Andre G Doree. Predictive maintenance using tree-based classification techniques: A case of railway switches. *Transportation Research Part C: Emerging Technologies*, 101:35–54, 2019.

[5] Thyago P Carvalho, Fabrízzio AAMN Soares, Roberto Vita, Roberto da P Francisco, João P Basto, and Symone GS Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, 137:106024, 2019.

[6] Zeki Murat Çınar, Abubakar Abdussalam Nuhu, Qasim Zeeshan, Orhan Korhan, Mohammed Asmael, and Babak Safaei. Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0. *Sustainability*, 12(19):8211, 2020.

[7] Xuewu Dai and Zhiwei Gao. From model, signal to knowledge: A data-driven perspective of fault detection and diagnosis. *IEEE Transactions on Industrial Informatics*, 9(4):2226–2238, 2013.

[8] IA Daniyan, K Mpofu, and AO Adeodu. Development of a diagnostic and prognostic tool for predictive maintenance in the railcar industry. *Procedia CIRP*, 90:109–114, 2020.

[9] Arun Das and Paul Rad. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*, 2020.

[10] Narjes Davari, Bruno Veloso, Gustavo de Assis Costa, Pedro Mota Pereira, Rita P Ribeiro, and João Gama. A survey on data-driven predictive maintenance for the railway industry. *Sensors*, 21(17):5739, 2021.

[11] Tanusree De, Prasenjit Giri, Ahmeduvesh Mevawala, Ramyasri Nemani, and Arati Deo. Explainable ai: a hybrid approach to generate human-interpretable explanation for deep learning prediction. *Procedia Computer Science*, 168:40–48, 2020.

[12] Kary Främling, Marcus Westberg, Martin Jullum, Manik Madhikermi, and Avleen Malhi. Comparison of contextual importance and utility with lime and shapley values. In *International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*, pages 39–54. Springer, 2021.

[13] Damien Garreau and Ulrike Luxburg. Explaining the explainer: A first theoretical analysis of lime. In *International Conference on Artificial Intelligence and Statistics*, pages 1287–1296. PMLR, 2020.

[14] Jyh-Yih Hsu, Yi-Fu Wang, Kuan-Cheng Lin, Mu-Yen Chen, and Jenneille Hwai-Yuan Hsu. Wind turbine fault diagnosis and predictive maintenance through statistical process control and machine learning. *Ieee Access*, 8:23427–23439, 2020.

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

[16] Stephan Matzka. Explainable artificial intelligence for predictive maintenance applications. In *2020 Third International Conference on Artificial Intelligence for Industries (AI4I)*, pages 69–74. IEEE, 2020.

[17] H Du Nguyen, Kim Phuc Tran, Sébastien Thomassey, and Moez Hamad. Forecasting and anomaly detection approaches using lstm and lstm autoencoder techniques with the applications in supply chain management. *International Journal of Information Management*, 57: 102282, 2021.

[18] Sandeep Patalay. Predictive maintenance of railway points.

[19] Krish Patel and Aakash Shanbhag. Exploring ml for predictive maintenance using imbalance correction techniques and shap. In *2022 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, pages 1–10. IEEE, 2022.

[20] Rune Prytz, Sławomir Nowaczyk, Thorsteinn Rögnvaldsson, and Stefan Byttner. Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data. *Engineering applications of artificial intelligence*, 41:139–150, 2015.

[21] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[22] Rita P Ribeiro, Saulo Martiello Mastelini, Narjes Davari, Ehsan Aminian, Bruno Veloso, and João Gama. Online anomaly explanation: A case study on predictive maintenance. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 383–399. Springer, 2022.

[23] Giulio Salierno, Sabatino Morvillo, Letizia Leonardi, and Giacomo Cabri. An architecture for predictive maintenance of railway points based on big data analytics. In *International Conference on Advanced Information Systems Engineering*, pages 29–40. Springer, 2020.

[24] Bibhudhendu Shukla, Ip-Shing Fan, and Ian Jennions. Opportunities for explainable artificial intelligence in aerospace predictive maintenance. In *PHM Society European Conference*, volume 5, pages 11–11, 2020.

[25] Irfan Siddavatam, Ashwini Dalvi, Viraj Thakkar, Aditya Vedpathak, Smit Moradiya, and Apoorva Jain. Explainability using decision trees and monte carlo simulations. *Available at SSRN 3868707*, 2021.

[26] Xuelin Situ, Ingrid Zukerman, Cecile Paris, Sameen Maruf, and Gholamreza Haffari. Learning to explain: Generating stable explanations fast. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5340–5355, 2021.

[27] Simon Vollert, Martin Atzmueller, and Andreas Theissler. Interpretable machine learning: A brief survey from the predictive maintenance perspective. In *2021 26th IEEE international conference on emerging technologies and factory automation (ETFA)*, pages 01–08. IEEE, 2021.

[28] Chathurika S Wickramasinghe, Kasun Amarasinghe, Daniel L Marino, Craig Rieger, and Milos Manic. Explainable unsupervised machine learning for cyber-physical systems. *IEEE Access*, 9:131824–131843, 2021.

[29] Yueyue Yao, Jianghong Ma, and Yunming Ye. Regularizing autoencoders with wavelet transform for sequence anomaly detection. *Pattern Recognition*, 134:109084, 2023.

[30] Weiting Zhang, Dong Yang, and Hongchao Wang. Data-driven methods for predictive maintenance of industrial equipment: A survey. *IEEE Systems Journal*, 13(3):2213–2227, 2019.

[31] Tiago Zonta, Cristiano André Da Costa, Rodrigo da Rosa Righi, Miromar Jose de Lima, Eduardo Silveira da Trindade, and Guann Pyng Li. Predictive maintenance in the industry 4.0: A systematic literature review. *Computers & Industrial Engineering*, 150:106889, 2020.