CAMILA APARECIDA DA SILVA ALMEIDA

DEVELOPMENT OF A MACHINE LEARNING MODEL TO PREDICT CONSTRUCTION MACHINERY GEARBOXES' HEALTH STATUS

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA MECÂNICA
2024

CAMILA APARECIDA DA SILVA ALMEIDA

DEVELOPMENT OF A MACHINE LEARNING MODEL TO PREDICT CONSTRUCTION MACHINERY GEARBOXES' HEALTH STATUS

Final Paper submitted to the Faculty of Mechanical Engineering from Federal University of Uberlândia in partial fulfillment of the requirements for the degree of Aeronautical Engineer. *FINAL VERSION*

Advisor: Prof. Dr. Higor Luis Silva

Uberlândia - MG
2024

CAMILA APARECIDA DA SILVA ALMEIDA

DESENVOLVIMENTO DE UM MODELO DE APRENDIZADO DE MÁQUINA PARA PREVER O ESTADO DE SAÚDE DAS TRANSMISSÕES DE MÁQUINAS DE CONSTRUÇÃO

Trabalho de Conclusão de Curso apresentado à Faculdade de Engenharia Mecânica da Universidade Federal de Uberlândia como parte dos requisitos para obtenção do título de Bacharel em Engenharia Aeronáutica. *VERSÃO REVISADA*

Orientador: Prof. Dr. Higor Luis Silva

Uberlândia - MG
2024

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to those who have been by my side throughout my academic journey. Firstly, I would like to thank my parents and grandmother for all the unconditional support they have given me throughout my degree. Your love and encouragement were fundamental to me reaching this moment. Writing these thanks took a while, as I was afraid to face the reality of my father's absence in this achievement. Although he is no longer with us to share this moment, I am sure that, wherever he is, he is deeply proud of all my achievements. He, after my mother, was my greatest motivator, and I am immensely grateful for the precious moments we shared.

I would also like to express my thanks to my friends who have been by my side throughout this journey. They patiently helped me understand difficult concepts, and laughed and cried with me. Special thanks to Arnaldo, Milena, Nicolle, Yasmin, Natália, Alfredo and Nuno. Growing up alongside you has been a real honor.

Finally, I would like to thank ZF for the opportunity to do this work. To my supervisor Isabela, who supported and guided me throughout the process, discussing results and pointing out improvements. I am also grateful to my coworkers, who offered me all the support I needed during my time at the company, and to my professor Higor, for his guidance.

This moment is the result of the joint efforts of so many special people in my life. To all of you, my deep and sincere thanks.

“Slow down, you crazy child
You’re so ambitious for a juvenile
But then if you’re so smart
Tell me why are you still so afraid ?”
(Billi Joel)

RESUMO

ALMEIDA, C.A.S. **Desenvolvimento de um modelo de aprendizado de máquina para prever o estado de saúde das transmissões de máquinas de construção**. 2024. 65 p. Trabalho de Conclusão de Curso (Graduação em Engenharia Aeronáutica) – Faculdade de Engenharia Mecânica, Universidade Federal de Uberlândia, Uberlândia - MG, 2024.

Com o aumento da dependência de abordagens orientadas por dados em estratégias de manutenção industrial, a necessidade de técnicas avançadas de manutenção preditiva torna-se cada vez mais evidente. Este estudo concentra-se na aplicação de algoritmos de Machine Learning (ML) para aprimorar a confiabilidade e eficiência dos sistemas de transmissão de veículos, com foco no monitoramento da saúde das transmissões powershift para máquinas de construção na ZF Friedrichshafen, uma empresa global de tecnologia automotiva. Ao utilizar algoritmos de ML, este projeto visa identificar padrões em dados operacionais, facilitando a previsão de possíveis falhas e permitindo intervenções oportunas. O objetivo central do estudo foi comparar os algoritmos de ML, Floresta Aleatória (Random Forest), K Vizinhos Mais Próximos (KNN) e Máquinas de Vetores de Suporte (SVM), para identificar o mais eficaz na previsão de falhas de transmissão. A pesquisa envolveu etapas que abrangeram desde a coleta e pré-processamento de dados até a otimização e avaliação dos modelos, utilizando métricas de aprendizado de máquina. Os resultados mostraram que o algoritmo de Floresta Aleatória superou os outros em termos de precisão, recall e pontuação F_1, para a aplicação em estudo. Sua capacidade robusta em classificar com precisão instâncias positivas e negativas foi especialmente significativa na análise de dados operacionais de veículos.

Palavras-chave: Algoritmos de aprendizado de máquina, manutenção preditiva, sistemas de transmissão de veículos, análise de dados operacionais, random forest.

ABSTRACT

ALMEIDA, C.A.S. **Development of a machine learning model to predict construction machinery gearboxes' health status**. 2024. 65 p. Trabalho de Conclusão de Curso (Graduação em Engenharia Aeronáutica) – Faculdade de Engenharia Mecânica, Universidade Federal de Uberlândia, Uberlândia - MG, 2024.

As industries increasingly turn to data-driven methods in their maintenance strategies, the need for advanced predictive maintenance techniques becomes increasingly evident. This study focuses on the application of Machine Learning (ML) algorithms to enhance the reliability and efficiency of vehicle transmission systems, with a focus on monitoring the health of powershift transmissions for construction machinery, at ZF Friedrichshafen, a global automotive technology company. By utilizing ML algorithms, this project aims to identify patterns in operational data, facilitating the prediction of potential failures and enabling timely interventions. The goal of the study was to compare the ML ML algorithms, Random Forest, K Nearest Neighbors (KNN), and Support Vector Machines (SVM) to identify the most effective in predicting transmission failures. The research involved stages ranging from data collection and pre-processing to model optimization and evaluation, using machine learning metrics. The results revealed that the Random Forest algorithm outperformed the others in terms of precision, recall, and F_1 score for the studied use case. Its robust ability to accurately classify positive and negative instances was particularly significant in the analysis of vehicle operational data.

Keywords: Machine learning algorithms, predictive maintenance, vehicle transmission systems, operational data analysis, random forest.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AUC	Area Under Curve
CV	Cross-validation
CVT	Continuously Variable Transmission
EEPROMs	Electrically Erasable Programmable Memory
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GSA	Grid Search Algorithm
JSON	JavaScript Object Notation
k-NN	k-Nearest Neighbors
KS	Kolmogorov-Smirnov
ML	Machine Learning
MPC	Maintenance Planning and Control
PdM	Predictive Maintenance
ROC	Receiver Operating Characteristic
SVM	Support Vector Machines
TCU	Transmission Control Unit)
TN	True Negative
TP	True Positive
TPR	True Positive Rate

LIST OF SYMBOLS

D	— Training set
N	— Training examples
x_i	— Features
y_i	— Label
c	— Number of classes
$X_{\min}$	— Minimum value
$X_{\max}$	— Maximum value
$\bar{X}$	— Mean value
σ	— Standard deviation
k	— Number of folds
Θ_l	— Random vectors
m	— Number of nodes
l	— Number of trees
K	— Number of Neighbors
p	— Positive integer
α_i	— Lagrange multiplier
ϕ	— Kernel function
$\mathbb{R}$	— Real numbers
C	— Regularization parameter
$gamma$	— Kernel parameter

CONTENTS

CHAPTER

1 INTRODUCTION

FAILURE prediction is an essential component of industrial maintenance strategies, which aim to prevent system failures from occurring and minimize unplanned downtime of equipment, machines, and processes Leukel, González and Riekert (2021). Many companies employ MPC (Maintenance Planning and Control) teams to schedule preventive and predictive maintenance, using a variety of technologies, such as vibration sensors, thermography, and oil analysis, to anticipate future failures and plan timely interventions. Preventive maintenance uses data analysis to identify patterns in equipment behavior and performance, leading to interventions before serious failures occur Leukel, González and Riekert (2021).

The availability of data and technological developments are using Machine Learning (ML) algorithms in a variety of complex problems such as preventive maintenance applications. These algorithms have the ability to find patterns in data and learn from them. This makes it possible to later use new data that has not been seen before and make predictions about it based on what has been learned previously Campos, Costa and Vieira (2019).

According to Silva, Sá and Menegatti (2019), "the use of predictive maintenance with technologies is presented as an alternative to guaranteeing and improving the levels of reliability". In this context, the use of machine learning methods is a great ally for monitoring the health of vehicle transmissions. The automotive transmission has the important function of adjusting the engine's torque and speed according to the vehicle's needs in different driving situations. This ensures that the vehicle maintains the right traction and speed, supporting the engine to always work in the best possible conditions Lechner and Naunheimer (1999).

To enhance vehicle dynamics, efficiency, and safety, the use of electronic components such as the TCU (Transmission Control Unit) has been adopted. This is an electronic device that manages and regulates the operation of the automatic transmission system in a vehicle. The TCU communicates with various sensors and systems in the vehicle to obtain information on speed, throttle position, and other parameters that influence the behavior of the transmission.

This information can be stored in EEPROMs (Electrically Erasable Programmable Memory).

EEPROMs, a type of volatile memory in each TCU, can store small amounts of data and allow users to erase and reprogram individual bytes as needed. The Data Analytics team at ZF company developed the TCU Data Tool, which converts the binary data stored on EEPROMs into a human readable format, most specifically JavaScript Object Notation (JSON).

ZF is a global technology company that provides systems for passenger vehicles, commercial vehicles, and industrial technology, contributing to the next generation of mobility. The company operates in four main technological domains: Vehicle Motion Control, Integrated Safety, Automated Driving, and Electric Mobility, offering comprehensive product and software solutions to established vehicle manufacturers. ZF products include transmission technology for cars and commercial vehicles, as well as specialized industrial equipment. The company is involved in various sectors, including railway, maritime, defense, aviation, and general industrial applications ZF (2024).

1.1 Main Objectives

This work aims to use the field data stored in the vehicle's EEPROM to compare machine learning algorithms and identify the most effective one for predicting failures of a powershift transmission used in construction machinery applications, thus indicating the state of health of this component. Another objective is to find an algorithm capable of minimizing the number of cases in which transmission problems are not detected, i.e., when the algorithm indicates that the transmission is healthy despite real problems.

Thereby, this work aims to provide a tool to effectively monitor transmissions, aiding in the early detection of faults and contributing to the maintenance of vehicles equipped with ZF transmissions. Possible impacts include reducing maintenance costs, increasing operational efficiency, improving vehicle safety during operation and reduceing machine downtime.

1.2 Structuring the Document

The document is structured as follows:

- Chapter 2: It presents a literature review on the main topics involved in this document, such as data analytics, machine learning approaches.
- Chapter 3: The methodology employed in the development of the work is presented and detailed, including explanations of the decisions made. The aim is to provide a comprehensive understanding of the context of the work presented.

- Chapter 4: This section presents the results obtained from the methodology used, providing a discussion of the results, limitations encountered during the development of the work and justification for the choice of a final model.
- Chapter 5: This provides a review of the information presented and discussed, with the aim of offering a conclusion to the work and providing perspectives for future research.

CHAPTER 2

LITERATURE REVIEW

THIS chapter describes the main concepts of machine learning, with an emphasis on the pre-processing stages, model selection, and a detailed explanation of how it works. The theory behind these concepts is presented, along with the metrics used to validate their performance. It also provides an overview of the research already conducted by ZF's Data Analytics team.

2.1 Previous Works

The Data analyses Team plays a central role in ZF's business unit, focusing on developing solutions for the industrial division. Its main responsibilities include data visualization, data analyses from tests and field operation and data collection enhancements. Additionally, they are involved in developing integrated software for data acquisition, promoting efficiency, and continuous improvement of systems.

Within the team, several previous projects have explored transmission analyses methods using machine learning. For example, Monteiro (2022) conducted a study using an unsupervised method to analyze the impacts and correlations of transmission's adaption functionality in construction machineries' gearboxes. Berenguer (2021) and Magalhães (2022), on the other hand, employed supervised methods in their projects. Magalhães (2022) investigated relationships between transmission data and available vehicle types, while Berenguer (2021) compared different machine learning classification approaches to compose a predictive maintenance system for a specific construction machinery transmission. These projects provided a solid foundation for this project, especially Berenguer (2021)'s work, which conducts similar analyses on a CVT (Continuously Variable Transmission) transmission contributing to a broader understanding of the subject. Although analyses have already been carried out for other types of transmission, the powershift transmission had not been investigated in detail before, and this work is really important for that.

In this context, the team's work aims not only to analyze existing data, but also to explore techniques and approaches to increase the analyses' efficiency and accuracy, thus contributing to the continuous development of innovative and validated solutions in ZF's industrial area.

2.2 Data Analytics

Data exploration and interpretation, known as Data Analytics, is essential for discovering valuable insights and meaningful patterns in specific data sets. When combined with Machine Learning (ML), this practice becomes even more powerful, enabling complex and scalable analyses that would not be possible manually Tsai *et al.* (2015), Hofmann, Neukart and Bäck (2017).

Data analyses can be divided into several distinct categories: descriptive analyses, which focuses on exploring and describing events through visual representations and statistical data; diagnostic analyses, which seeks to understand the roots of the events observed in the descriptive phase; predictive analyses, which aims to avoid undesirable events by anticipating the possibility of a specific event occurring; and prescriptive analyses, which is dedicated to anticipating the possible results of an action Kumbure and Luukka (2022).

2.3 Machine Learning

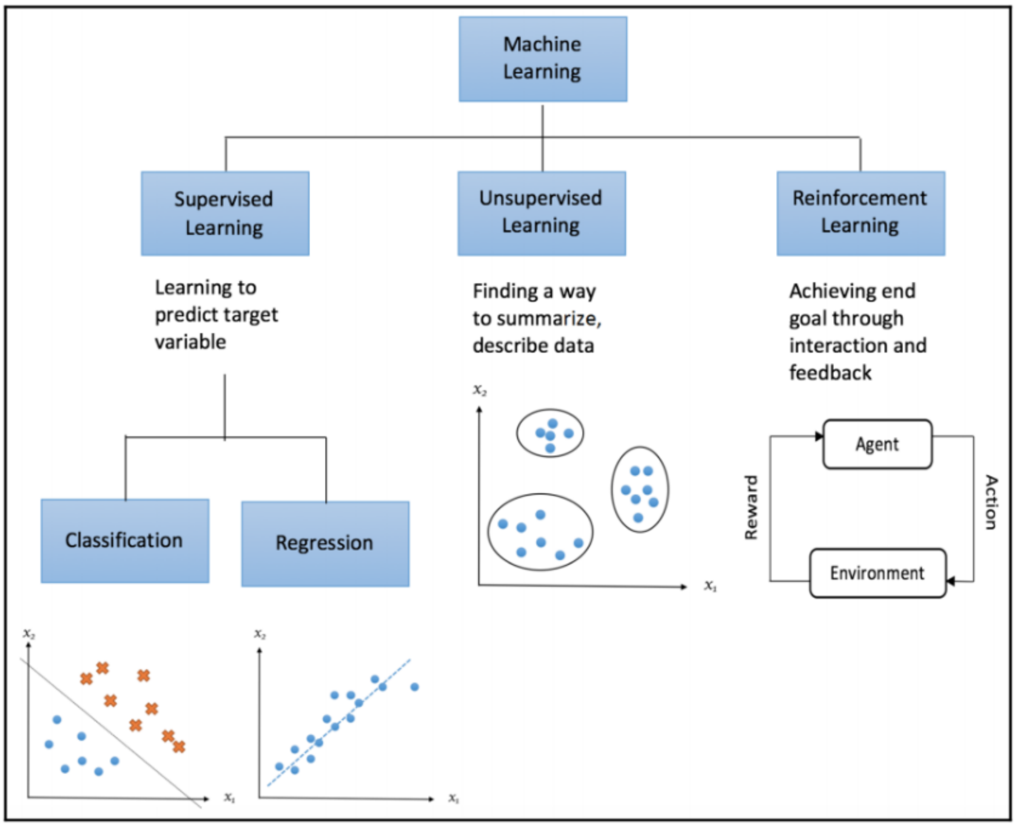
Machine learning represents a dynamic field of computer algorithms that aims to emulate human intelligence through environmental learning processes Naqa and Murphy (2015). The main objective of learning systems involves the exploration and application of algorithms that use previous data to predict outcomes on new inputs, relying heavily on principles of linear algebra, probability theory, statistics, and mathematical optimization Liu (2017).

The substantial volume of data accumulated by industrial systems encompasses valuable insights into processes, events, and alarms on an industrial production line. Equipment maintenance is a fundamental element that directly affects uptime and operational efficiency, thus protecting the company from significant losses. Identifying and correcting equipment faults becomes imperative to avoid possible interruptions in production processes. Hence, ML techniques have emerged as a promising solution in predictive maintenance (PdM) scenarios, to anticipate failures in production lines Carvalho *et al.* (2019).

The formal definition of ML's concept most accepted today was proposed by Mitchell (1997) as "a computer program is said to learn from experience E concerning some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E". Thus, it can be inferred that the machine learns when it improves its success in a given task by interacting with the problem using the provided data.

Depending on the problem's nature, machine learning tasks can be classified into three different categories: supervised learning, unsupervised learning, and reinforcement learning Liu (2017). Figure 1 presents the categories cited above.

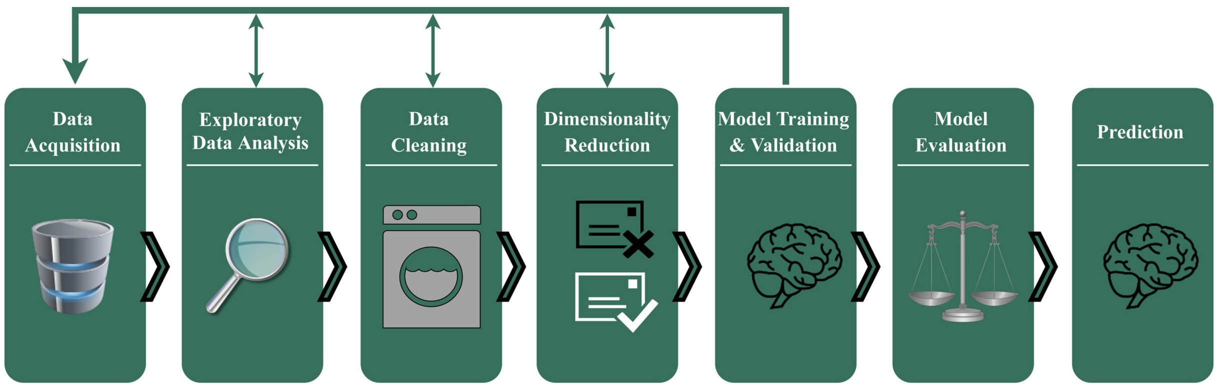
Figure 1 – Types of machine learning tasks.



Source: Liu (2017).

A workflow for developing ML models is shown in Figure 2, providing a brief overview of the main stages of a machine-learning project.

Figure 2 – Workflow for developing ML models.



Source: Maleki *et al.* (2020).

.

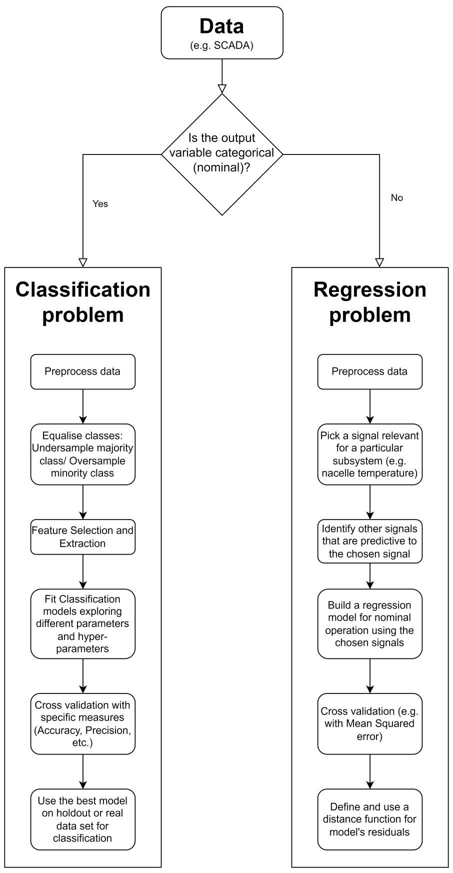
2.3.1 Supervised Learning

In supervised learning, the main objective is to understand the connection between a given set of input variables, denoted as x, and an output variable, represented by y. This understanding enables the model to predict results for new and unseen data instances based on the acquired mapping. The learning process in supervised settings relies heavily on labeled data, where each data point is paired with a known output, usually provided through events or expert annotations. Mathematically, this relationship can be formulated as shown in Equation 2.1, where D represents the training set comprising N training samples.

$$D = \{(x_i, y_i)\}_{i=1}^{N} \tag{2.1}$$

Where x_i is a vector of numbers of dimension D, representing features, attributes, or variables. y_i can be a categorical or nominal variable from some finite set, in which $y_i \in \{1, \ldots, c\}$, with c being the number of classes; or it can be a real-valued scalar. When y_i is categorical, this is a classification problem, and when y_i is a real value, this is a regression problem Murphy (2012). Figure 3 describes the approaches to the classification and regression problems by Stetco *et al.* (2019).

Figure 3 – Typical workflow for two supervised tasks.


Data
(e.g. SCADA)
Is the output variable categorical (nominal)?
Yes
No
Classification problem
Preprocess data
Equalise classes: Undersample majority class/ Oversample minority class
Feature Selection and Extraction
Fit Classification models exploring different parameters and hyper-parameters
Cross validation with specific measures (Accuracy, Precision, etc.)
Use the best model on holdout or real data set for classification
Regression problem
Preprocess data
Pick a signal relevant for a particular subsystem (e.g. nacelle temperature)
Identify other signals that are predictive to the chosen signal
Build a regression model for nominal operation using the chosen signals
Cross validation (e.g. with Mean Squared error)
Define and use a distance function for model's residuals

Source: Stetco *et al.* (2019).

.

2.3.2 Unsupervised Learning

In unsupervised learning, the machine receives inputs $(x_1, x_2, ..)$ but does not obtain target results, i.e., it does not have the output y Ghahramani (2003). The aim is to directly infer the probability density properties of the data without the help of a supervisor or teacher providing the correct answers Hastie *et al.* (2009). This is a much less well-defined problem because you don't tell the model what kind of patterns to look for and there is no obvious error metric to use Murphy (2012). The Equation 2.2 shows the mathematical representation for this method.

$$D = \{(x_i)\}_{i=1}^{N} \tag{2.2}$$

2.3.3 Reinforcement Learning

According to Ghahramani (2003), this is a less commonly used method, in this case, the machine interacts with its environment by producing actions $(a_1, a_2, ..)$ these actions affect the state of the environment, which in turn causes the machine to receive some scalar rewards (or punishments $r_1, r_2, ..$). The machine's goal is to learn to act in such a way as to maximize the future rewards it receives (or minimize the punishments) during its lifetime.

2.4 Gathering Data and Data Pre-processing

Gathering data is a fundamental step in the development of any ML workflow, so ensuring comprehensive data acquisition is essential to its success. Missing crucial features can result in unexpected consequences during subsequent analyses and model development, which can compromise the integrity of the entire study Maleki *et al.* (2020).

The use of ML requires the preparation of the data provided for the algorithms, including, data cleaning, feature scaling, feature selection, data balancing, etc. Muhamedyev (2015).

2.4.1 Data Cleaning

Companies have a huge availability of data that influences decision-making. However, they often contain issues such as missing data, incorrect formatting, and spelling mistakes during data entry. Other common problems include, duplicate records, or other types of invalid data, which affects the accuracy of the results. Therefore, data cleaning offers better data quality, which will ensure that it is ready for the analyses phase Rahm, Do *et al.* (2000), Ridzuan and Zainon (2019).

Although the effort required to clean the data during extraction and integration is time-consuming, it is necessary to achieve query optimization and data quality Fatima, Nazir and Khan (2017).

2.4.2 Feature Scaling

Data scaling is important in data pre-processing because it affects the performance of ML algorithms Ambarwari *et al.* (2020). If the scale is not applied and one feature has a greater magnitude compared to others, it could be dominant during another training Hackeling (2017). The most commonly used techniques are max-min normalization and standardization.

The max-min normalization resizes the original data so that it is all between 0 and 1, if the distribution of the data is not uniform, meaning there is a significant disparity between the maximum and minimum values across different features, normalization can distort the relative importance of the features. This can lead to a significant loss of information in some features, while others may be artificially amplified Sharma (2022),Hsu and Lin (2002). This scaling method is useful when the data set does not contain outliers, as the scale can be defined by extreme values Ali *et al.* (2014). The Equation 2.3 presents the mathematical representation of this normalization.

$$X_{\text{normal}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{2.3}$$

where $X_{\min}$ and $X_{\max}$ are the minimum and maximum values in the dataset respectively and X is each value.

Standardization, on the other hand, first subtracts the mean value $(\overline{X})$ and then divides the result by the standard deviation (σ), Borriello *et al.* (2024), as presented in Equation 2.4.

$$X_{\text{stand}} = \frac{X - \bar{X}}{\sigma} \tag{2.4}$$

The values are not limited to a certain range by standardization; moreover, this approach makes them much less affected by outliers Borriello *et al.* (2024).

2.4.3 Feature Selection

The objectives of feature selection are varied. They include simplifying models to make them more understandable, improving data mining performance by focusing on the most informative features and discarding irrelevant or redundant ones. In addition, feature selection aims to provide clean and understandable data for the model Li *et al.* (2017). When there is a large number of features, learning models tend to overfit, which can cause performance degradation on unseen data. High dimensional data can significantly increase memory storage requirements and the computational costs of data analyses Li *et al.* (2017).

According to Huang *et al.* (2012), characteristics should not be highly correlated with each other as this would add redundancy to the problem. One way to mitigate this problem is to use correlation matrix to check the linear association between features, represented by the correlation coefficient, ranging from -1 to 1. This parameter associated with the covariance provide the foundation for many statistical techniques; magnitudes greater in absolute value indicate a stronger association. Positive values indicate a direct relationship, while negative values indicate a negative or inverse relationship Hadd and Rodgers (2020).

2.4.4 Data Balancing

One of the primary challenges in machine learning is training classifiers with unbalanced data, a common characteristic of most real-world problems Carvalho and Prati (2018). Conventional algorithms tend to underperform on such datasets, as they are typically designed with the assumption of balanced data distribution. For a binary response variable with two classes, when the event is under-represented, it is called a minority class Wah *et al.* (2016).

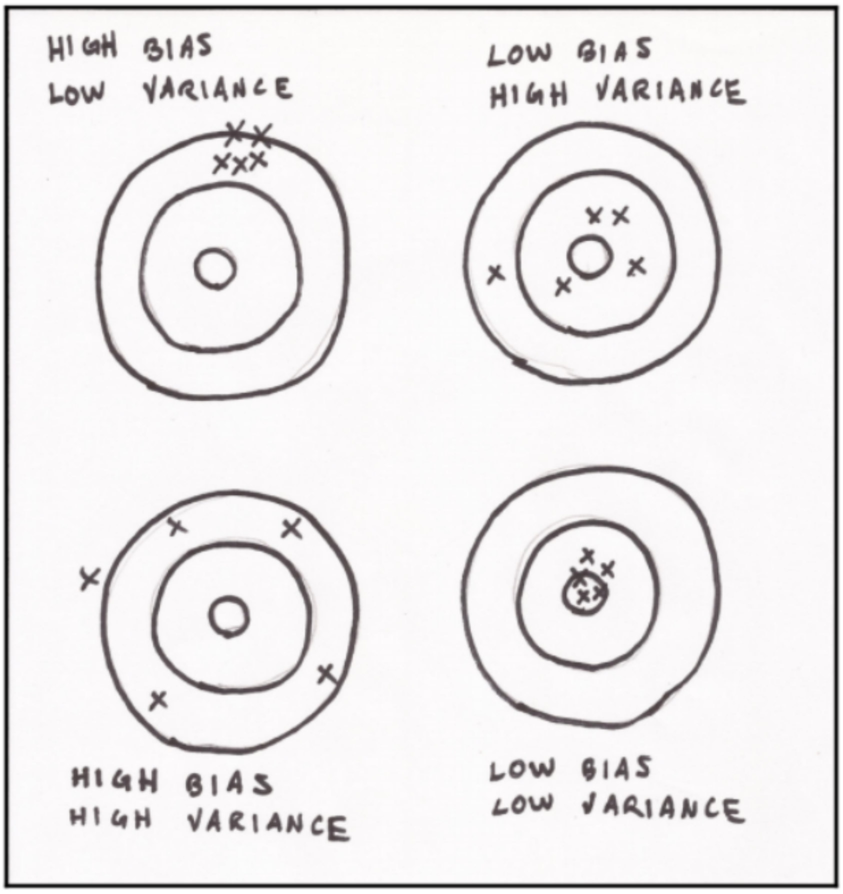
This problem of unbalanced class distribution can lead algorithms to learn excessively complex models that fit the data very well and have little relevance. It can be seen that, despite the better performance of computational intelligence techniques, they are biased towards instances of the majority class, learning better about the majority class than the minority class Farquad and Bose (2012). Usually, models such as Decision Tree (DT), KNN, and Naïve Bayes (NB), are biased towards the majority class. The SVM is less affected by class imbalance when compared to other classification algorithms. However, studies such as Gopi, Suvarna and Padmaja (2016) shown that strong class imbalance makes the SVM threshold skewed towards minority class samples.

2.5 Bias and Variance

The concept of square bias refers to the component of error in an algorithm stemming from its central tendency when trained on diverse datasets Brain and Webb (1999). Mitchell (1980) introduced the term *bias* in machine learning, defining it as "any rationale for favoring one generalization [hypothesis] over another, beyond strict adherence to the observed training instances". Models characterized by high bias tend to yield consistent outcomes for a given input, regardless of the training data utilized. Typically, models with high bias exhibit rigidity, meaning they under-fit the training data Hackeling (2017).

Variance quantifies the extent to which deviations from the central tendency contribute to error Brain and Webb (1999). A model with high variance will produce different errors for input, depending on the training set used. Unlike a model with high bias, a model with high variance may overfit the data, modeling the noise in the training data, in other words, be very flexible Hackeling (2017). The possible combinations of bias and variance are shown in Figure 4.

Figure 4 – Combinations of bias and variance.



Source: Hackeling (2017).

A model considered ideal will have low bias and low variance, but unfortunately, when one decreases the other frequently increases, this relationship is called the bias-variance trade-off, Hackeling (2017). To find the best trade-off Hackeling (2017), suggests the usage of cross-validation.

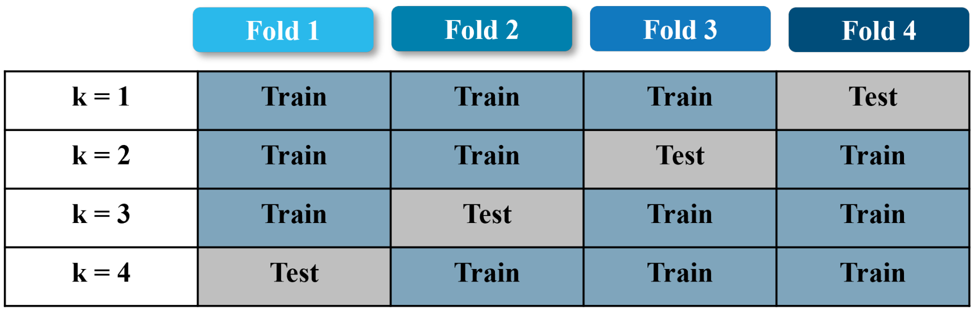
2.6 Cross-validation and Grid Search

A fundamental principle in accuracy assessment emphasizes the importance of separate sets for training and evaluation. A similar concern arises with the selection of user-defined parameters, also kown as hyperparameters, necessary for various machine learning techniques, such as the number of trees in Random Forests, the *gamma* and *c* values in radial basis function kernel for Support Vector Machines (SVM), and the k-distance in k-Nearest Neighbors (k-NN). These parameters significantly impact classification accuracy, often requiring optimization, commonly known as tuning. Careful selection of hyperparameters during the training phase ensures improved learning, resulting in enhanced performance of the chosen machine learning algorithm. Techniques like cross-validation and Grid Search facilitate this process Ramezan, Warner and Maxwell (2019); Ogunsanya, Isichei and Desai (2023).

Cross-validation (CV) is a way to reduce the bias caused by the random selection of samples and to verify the stability and generalization ability of the model Varma and Simon (2006). In cross-validation, multiple partitions are generated, potentially allowing each sample to be used multiple times for multiple purposes, with the overall aim of improving the statistical reliability of the results Ramezan, Warner and Maxwell (2019). Examples of cross-validation methods include k-fold and leave-one-out Duro, Franklin and Dubé (2012).

The k-fold cross-validation method involves randomly dividing the sample set into a series of folds (groups) of equal size, where k indicates the number of partitions, or folds, into which the data set is divided. $k-1$ of these partitions are used for training and the last one is used for testing. This procedure is repeated k times, so that k estimates are generated and tested. The average of the results is then reported Stone (1974). Figure 5 shows the steps described by Stone (1974), considering 4 folds.

Figure 5 – k-fold cross-validation method.



	Fold 1	Fold 2	Fold 3	Fold 4
k = 1	Train	Train	Train	Test
k = 2	Train	Train	Test	Train
k = 3	Train	Test	Train	Train
k = 4	Test	Train	Train	Train

Source: Adapted from Ghorbanzadeh *et al.* (2020).

Leave-one-out cross-validation is similar to k-fold cross-validation, except that the number of folds is equal to the number of samples in the data set. Despite its effectiveness, this method can be slow when faced with excessively large data sets Hastie *et al.* (2009).

The grid search algorithm (GSA) is a fundamental method for hyperparameter optimization. It systematically explores parameter values within predefined ranges to identify the combination that produces the best model performance Gao and Hou (2016). This method evaluates all viable combinations of hyperparameters to ensure optimal results. Grid search is known for its simplicity, ease of understanding, and ability to explore several parameter values simultaneously Ogunsanya, Isichei and Desai (2023) Ramadhan *et al.* (2017). According to Sukamto, Hadiyanto and Kurnianingsih (2023), using cross-validation with GridSearch facilitates testing the model's parameters without having to perform manual validation one by one.

2.7 Classifier Selection

Systematic empirical studies have indicated that the most effective algorithm may differ depending on the application in question Domingos (2012), so it is interesting to experiment with these in relation to the available data, in addition to adjusting hyperparameters in order to improve performance and avoid overfitting Saxena and Aggarwal (2020).

Hastie *et al.* (2009) divides the classification techniques into parametric, non-parametric, and ensemble. A parametric classifier uses the statistical distributions associated with each class to carry out its classifications. In contrast, non-parametric classifiers are used when the density function is unknown and are used to estimate the probability density function. As there is no prior information available, these models estimate the unknown function from the training data set based on trial and error Imam, Musilek and Reformat (2024), Sahoo and Kumar (2012). These techniques can deal with complex data and can be used to make predictions and find patterns and relationships within a data set Hastie *et al.* (2009).

Parametric methods are valued for their efficiency in terms of computational resources compared to non-parametric methods. However, their efficiency depends heavily on precise assumptions about the distribution. When these assumptions diverge from reality, the interpretation of the results can become subjective Jahnke (2015). According to Hastie *et al.* (2009), these models are not accurate representations of the data and are more susceptible to underfitting.

In the case of ensemble methods, the algorithms create a set of classifiers and then classify new data points using a (weighted) vote of their predictions Dietterich (2000), usually being classified into a positive and negative class Hastie *et al.* (2009).

Considering the extensive availability of machine learning algorithms and the limitations of parametric models compared to non-parametric ones, this work will explore two non-parametric models: SVM and KNN, along with an ensemble method, Random Forest. The optimization and testing of these models aim to provide a more flexible, robust, and accurate approach. This choice allows us to explore the diversity of techniques available and to maximize the effectiveness of data analyses. These models will be presented and defined in subsection 2.7.1.

2.7.1 Models Exploring

2.7.1.1 Random Forest

The random forest algorithm, introduced by Breiman (2001) in 2001, has become a very efficient tool for classification and regression tasks in various domains.

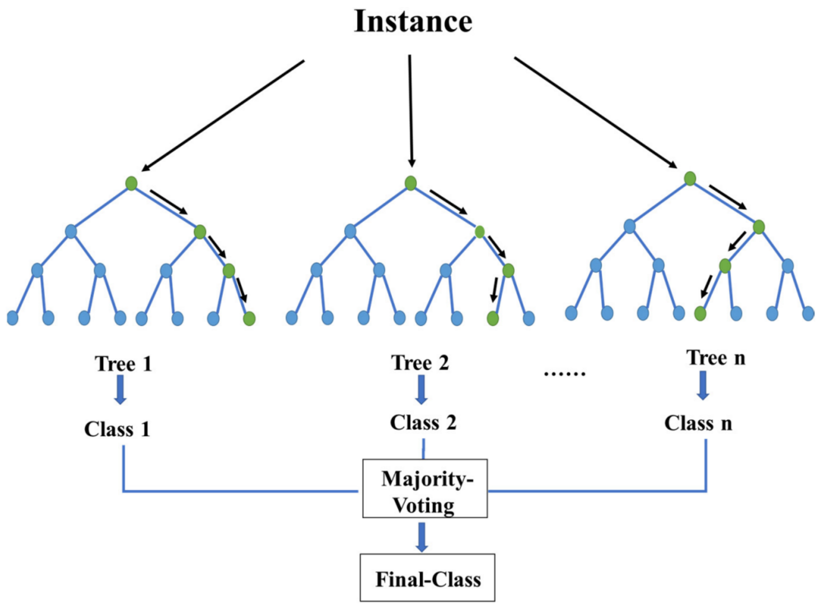
The Random Forest for classification consists of a collection of tree-structured classifiers, where the Θ_z are independent and identically distributed random vectors and each tree casts a unit vote for the most popular class of the input x Breiman (2001). This can be represented by

Equation 2.5.

$$h(x, \Theta_z), \quad z = 1, 2, \ldots \tag{2.5}$$

When employing the bagging technique, every decision tree within the collection is constructed using a randomized subset, drawn with replacement from the training dataset. As per statistics, the subset is anticipated to encompass roughly 64% of the instances present at least once in the subset. These instances within the subset are denoted as "in-bag instances", while the remaining ones (around 36%) are termed "out-of-bag instances". Each tree within the ensemble serves as a primary classifier to ascertain the class label of an unlabeled instance. This mechanism entails a collective voting approach, where each classifier contributes by casting a vote for the anticipated class label, mitigating overfitting Fawagreh, Gaber and Elyan (2014). Figure 6 exemplifies the above explanation.

Figure 6 – Random forest algorithm.



Source: Ma *et al.* (2023).

.

This model is not only robust to noise, enhancing its efficiency and versatility in classification and regression tasks, but it also offers valuable insights through its feature importance measures. Jesmeen *et al.* (2018), AlSagri and Ykhlef (2020) Equation 2.7 formulates the latter.

$$\text{Feature Importance}(i) = \frac{\sum_{j=1}^{m} F(j)}{l} \tag{2.6}$$

where $F(j)$ is the importance of each feature in the tree, m is the total number of nodes, i.e., the decision points in each tree and l is the number of trees.

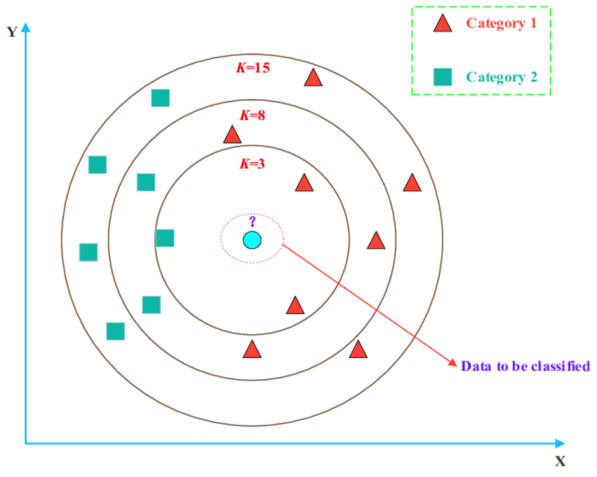
Besides providing the importance of features, this algorithm has important parameters that can be adjusted to obtain the ideal classifier. Two of these parameters are the number of estimators and the depth of the mask AlSagri and Ykhlef (2020).

The number of estimators and the maximum tree depth are two crucial hyperparameters that influence the performance of tree-based models. The number of estimators defines the number of trees that make up the model, while the maximum tree depth determines the maximum number of splits each tree can make. When the mask depth is too low, the model tends to be trained superficially, resulting in high bias. This means that the model may not be able to capture the full complexity of the data, leading to underfitting, where the model is oversimplified to adequately represent the patterns in the data. On the other hand, if the depth of the mask is too high, the model may overfit, even capturing the noise in the data. This leads to high variance, where the model fits the training data very well, but has difficulty generalizing to unseen data, resulting in overfitting. Therefore, finding an appropriate balance in the choice of maximum tree depth is essential to avoid both underfitting and overfitting, ensuring that the model can capture relevant patterns in the training data and generalize well to new data. This optimization of hyperparameters is a crucial part of creating effective, high-performance models, and techniques such as grid search are recommended for this purpose Brain and Webb (1999), Mantovani *et al.* (2018), Bartz *et al.* (2023).

2.7.1.2 K Nearest Neighbor

This algorithm estimates the conditional distribution of y based on x and then classifies a given observation into the class with the highest estimated probability. It classifies an object by conducting a majority vote of its neighbors, assigning the object to the most common class among the k nearest neighbors Hmeidi, Hawashin and El-Qawasmeh (2008) (Figure 7).

Figure 7 – KNN algorithm.



Source: Chen *et al.* (2023).

.

Empirical studies have consistently demonstrated the efficiency of this algorithm in various data sets, especially in classification scenarios. Researchers widely use it in data mining and machine learning applications due to its simple implementation and exceptional performance, being an important technique in data classification tasks Begum, Chakraborty and Sarkar (2015).

However, the performance of KNN classification can be affected by problems such as the selection of the K value and the selection of the distance measures, i.e., their hyperparameters. This is why cross-validation in conjunction with Grid Search has been used to test different values of K Zhang *et al.* (2018),Guo *et al.* (2003). For KNN, a small K parameter can make the result susceptible to noise, as it will only consider very close neighbors, while a large K value can unnecessarily include many points from the other class. It is therefore imperative to fine-tune the optimum value of K to obtain the best performance Sukamto, Hadiyanto and Kurnianingsih (2023).

By default the Euclidean distance is used in scikit learn to calculate the distance measure. The Euclidean distance is a measure of distance between two points in Euclidean space. It is derived from the Pythagoras theorem and is commonly used in geometry and various areas of mathematics, physics, and computer science. Formally, the Euclidean distance between two points $P(s_1, s_2)$ and $Q(t_1, t_2)$ on a two-dimensional plane can be calculated by the Equation 2.7.

$$d(\mathbf{S},\mathbf{T}) = \sqrt{(s_2 - s_1)^2 + (t_2 - t_1)^2} \tag{2.7}$$

According Kumbure and Luukka (2022) the Euclidean distance is often not the ideal choice for practical problems, and better results can be obtained by generalizing it. The Minkowski distance is a generalization of the Euclidean distance that can be used to calculate the distance between two points in an n-dimensional space. Kumbure and Luukka (2022). Using the Minkowski distance allows the method to obtain more reasonable nearest neighbors for the target sample. Another important advantage to this author about this method is that the nearest neighbors are weighted by fuzzy weights based on their similarity to the target sample, leading to a more accurate prediction using a weighted average. The Minkowski distance can be calculated by the Equation 2.15.

$$d(\mathbf{S},\mathbf{T}) = \left(\sum_{i=1}^{n} |s_i - t_i|^p\right)^{\frac{1}{p}} \tag{2.8}$$

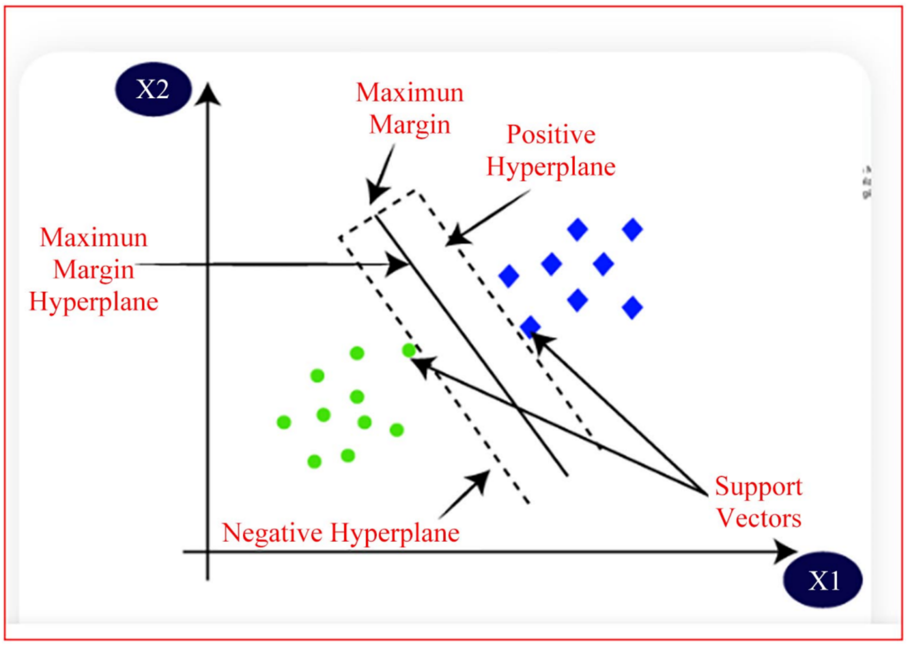
where p is a positive integer that determines the order of the Minkowski distance, and s_i and t_i are two points in a n-dimensional space.

Although the Euclidean distance is the default option in KNN problems, in this work the Minkowski distance is used for the advantages mentioned above.

2.7.1.3 Support Vector Machine

Support Vector Machine (SVM) belong to the field of supervised learning models, which have specialized learning algorithms adapted for classification and regression analyses. In essence, the SVM aims to identify the ideal hyperplane in an n-dimensional classification space, strategically maximizing the margin between distinct classes Cortes and Vapnik (1995) (Figure 8).

Figure 8 – SVM algorithm in 2-dimensions.



Source: Cortes and Vapnik (1995).

.

According to Cao, Naito and Ninomiya (2008), SVM with a non-linear kernel has shown good results in classifying patterns and has therefore been widely used in various application areas. Some common kernel functions include polynomial, RBF, sigmoid, etc. The general formulation for the karnel is shown in Equation 2.9.

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \phi(x, x_i) + b \tag{2.9}$$

Where α_i, the Lagrange multiplier, ϕ, the kernel function, where $y_i \in (-1, +1)$, indicating the class to which the feature vector $x_i \in \mathbb{R}^n$.

The SVM classifier, tailored for binary classification tasks, stands out as a kernel-based supervised learning technique employed for categorizing data into multiple classes. Nevertheless, it might not be the optimal choice when dealing with a considerable volume of training examples Cortes and Vapnik (1995). One of the hurdles associated with non-linear kernel SVMs is the escalating complexity of classification, which tends to correlate with the number of support vectors required Cao, Naito and Ninomiya (2008). However, as highlighted by Xie Yi Zhao and Zhang (2019), the algorithm possesses the capability to strike a balance between model complexity and the extraction of insights from sample information, thereby enhancing generalization performance. Furthermore, SVM enables using the technique of adjusting class weights, which is usually implemented to ensure that the model is trained in a more balanced way and that the minority classes have an adequate influence during training Hsu and Lin (2002), Rawat and Mishra (2022).

This study used the SVM with the RBF (Radial Basis Function) kernel. By adopting this kernel, the SVM's decision function is represented by the kernel function, denoted as ϕ, and mathematically represented by Equation 2.10.

$$\phi(x, x_i) = e^{-gamma||x-x_i||^2}, \quad gamma > 0 \tag{2.10}$$

The effect of SVM RBF depends, on the *gamma* parameter, represented in Equation 2.10 and the selection of the penalty coefficient C for the wrong subsample, another hyperparameter of this model Xie Yi Zhao and Zhang (2019). The *gamma* parameter defines the range of influence of a single training example learn (2024). In general, the selection of the hyperparameters is a non-convex optimization problem and thus many algorithms have been proposed to solve it, among them: grid search, random search, and others Wainer and Fonseca (2021).

2.8 Evaluating Models

In Supervised ML, there are several ways of evaluating the performance of learning algorithms and their classifiers Powers (2020). Which are related to the P performance mentioned at the beginning of the chapter.

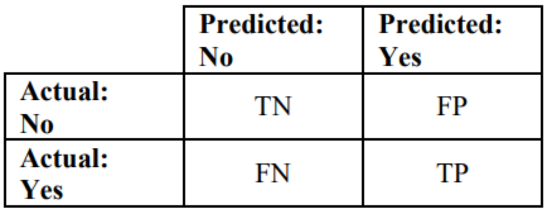
2.8.1 Confusion Matrix

The quality of a binary classification is typically assessed with the aid of a $2x2$ confusion matrix. This is simply a cross-tabulation of the labels assigned to a set of cases by a classifier against the corresponding labels in a reference data set. The labels used for the two classes can vary between studies, but generally take the form of positive versus negative Powers (2020). For this study, the classes will be divided into Category 0 (False) and Category 1 (True).

The four elements of the confusion matrix show the correct and incorrect allocations made by the classifier. The cases allocated correctly and located on the main diagonal of the matrix are true positives (TP, cases that have been classified as positive and have a positive label) and true negatives (TN, cases that have been classified as negative and also have a negative label) Foody (2023).

Instances incorrectly assigned in the classification process represent erroneous classifications, manifesting as false positives (FP) and false negatives (FN) Foody (2023). Figure 9 shows the confusion matrix.

Figure 9 – Confusion matrix to binary classification.



	Predicted: No	**Predicted: Yes**
Actual: No	TN	FP
Actual: Yes	FN	TP

Source: Navin and Pankaja (2016).

In classification models, the decision function generates scores ranging from 0 to 1, indicating the probability of a positive label. By default, the threshold for classifying instances is set at 0.5. Thus, results between 0 and 0.5 are classified as negative, while those above 0.5 are labeled positive. However, project requirements may justify adjusting this threshold as necessary Pedregosa *et al.* (2011).

2.8.2 Precision and Recall

Some common performance measures can be calculated from the confusion matrix, such, as precision and recall and AUC-ROC curve Hackeling (2017).

2.8.3 Precision

Precision measures the specificity of retrieval, defined as the proportion of items collected that are considered relevant by the user; this measure penalizes the retrieval of irrelevant items by the system (FP) but does not penalize the system's failure to collect items that the user considers relevant (FN) and can be calculated by Equation 2.11 Alvarez (2002).

$$Precision = \frac{TP}{TP+FP} \tag{2.11}$$

2.8.4 Recall

The recall also called sensitivity, represents the proportion of outcomes generated by the system in comparison to all actual instances of the malicious class (TP), accurately predicting positive observations, i.e., the model's ability to predict positive cases. It is calculated as the ratio between truly positive predictions and the set of all positive predictions as presented in Equation 2.12 . Salam *et al.* (2021), Luque *et al.* (2019).

$$Recall = \frac{TP}{TP + FN} \tag{2.12}$$

2.8.5 F_1 score

Most machine learning application areas widely use the F_1 measure. The F_1 score combines precision and recall in a single metric, simplifying performance evaluation. It calculates the harmonic mean of these metrics, so when the F_1 score is low, it indicates that either precision or recall is low Alvarez (2002).

The F_1 score is calculated as presented in Equation 2.13:

$$F_1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.13}$$

2.8.6 Specificity

Specificity is a crucial measure of a model's ability to accurately predict negative cases. It quantifies the proportion of correctly identified negative predictions to the total number of actual negative cases. In essence, specificity measures the model's accuracy in discerning instances that do not present the desired condition or outcome Luque *et al.* (2019). The Specificity is calculated as shown in the Equation 2.14

$$Specificity = \frac{TN}{TN + FP} \tag{2.14}$$

2.8.7 AUC-ROC Curve

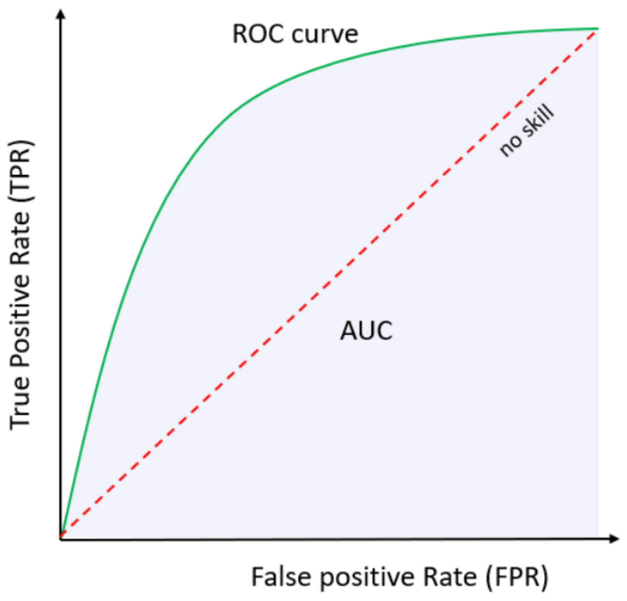
The ROC (Receiver Operating Characteristic) curve graphically represents the performance of a classification model in two dimensions. ROC graphs have long been used to represent the trade-off between classifier hit and miss rates Egan (1975). As a performance graph method, its attributes are particularly advantageous for domains characterized by unbalanced class distributions and variable classification error costs. These qualities have gained importance with ongoing research into cost-sensitive learning and learning amid class imbalances Fawcett (2006).

Furthermore, the most important statistic associated with ROC curves is the Area Under the ROC Curve or AUC. The Area Under Curve (AUC) is a one-dimensional metric that quantifies one perspective of the model's performance Marzban (2004). As the curve is located in the unit square, it has $0 \leq \text{AUC} \leq 1$. The AUC is equal to 1 when the classifier scores each positive higher than each negative, indicating a theoretically perfect test; an AUC value of 0.5 indicates no discrimination ability, and an AUC equal to 0 indicates that each negative is scored higher than each positive Flach (2016). To plot the AUC-ROC curve, it is necessary to calculate the true positive rate (TPR) and false positive rate (FPR), which can be obtained from the confusion matrix. Where TPR is represented by the Equation 2.12, being equal to recall, and FPR by the equation Equation 2.15.

$$FPR = 1 - Specificity \tag{2.15}$$

The Figure 10 exemplifies the AUC-ROC curve.

Figure 10 – Trade-off between sensitivity and FPR rate.



Source: Elbasheer *et al.* (2022).

.

CHAPTER 3

METHODS

This chapter details the methods used to develop the project, including gathering and converting data using the company's in-house tool, the TCU Data Tool. The methodologies applied during data exploration and pre-processing are also discussed. In addition, the techniques employed to optimize the models are presented, along with the metrics used to evaluate them.

3.1 Working and Development Environment

The programming language chosen for this project was Python, primarily because it serves as the team's default working language. Python, as highlighted by Sarkar, Bali and Sharma (2018), stands out as one of the most popular programming languages for data science, with a highly active developer community and a large number of useful libraries for scientific computing and machine learning. The version used is 3.10.10.

According to Sarkar, Bali and Sharma (2018), the ecosystem in Python is a collection of libraries that allow developers to extract and transform data, perform data manipulation operations, apply robust existing machine learning algorithms, and easily develop custom algorithms. These libraries include, NumPy, pandas, scikit-learn Matplotlib, and Seaborn.

The NumPy library offers robust support for multidimensional arrays and a comprehensive set of mathematical functions, facilitating a wide range of scientific calculations Müller and Guido (2016). Pandas, on the other hand, perfectly integrates the computational power of NumPy matrices with versatile data manipulation features reminiscent of relational databases. With its advanced indexing features, Pandas simplifies reshaping and slicing operations, as well as providing optimized methods for dealing with diverse and labeled datasets McKinney (2012).

The scikit-learn library is widely used for machine learning tasks due to its popularity and accessibility in the open-source community Raschka (2015). Taking advantage of a diverse ecosystem, scikit-learn offers cutting-edge implementations of several renowned machine learning algorithms, all in a user-friendly interface tightly coupled with Python. This addresses the growing need for statistical data analyses among individuals outside the traditional domains of the software and web industries Sarkar, Bali and Sharma (2018).

Complementing these tools, Matplotlib and Seaborn stand out as indispensable for creating visual data representations. They offer users a multitude of graphical options and extensive customization features, facilitating data visualization and exploration effectively VanderPlas (2016).

3.2 Gathering Data

Initially, the binary data collected from each vehicle's eeprom were converted into JSON format with the help of the TCU Data Tool. The files were then stored in a folder for further analyses.

After converting the data, all the files stored in the folder were imported into Python in order to extract relevant information for the analyses and ensure that they were properly processed. The criteria for selecting the files were based on the presence of essential information, such as serial number and operating time. This filtering stage enabled the identification and understanding of the vehicle's operation, ensuring the quality of the data analyzed in these respects.

The absence of correct information can be attributed to human error or problems when reading/converting the binary file. Table 1 in section 4.1 shows the number of files available before and after this stage, providing a clear view of the data filtering and selection process.

3.3 Exploration Data

Data exploration is fundamental for understanding its details. In this work, this exploration was carried out with the help of the *dataprep.eda* module, which is part of the *DataPrep* library. This tool is designed to help with exploratory data analyses in Python, providing information on missing values, correlated features, descriptive statistics and visualizations. The aim is to understand the distribution of information, check whether there are missing values in the features used that will need to be dealt with, and the operating time of the vehicles. Exploring the information was essential to understanding the behavior of the variables contained in the dataset.

3.3.1 Distribution of Data

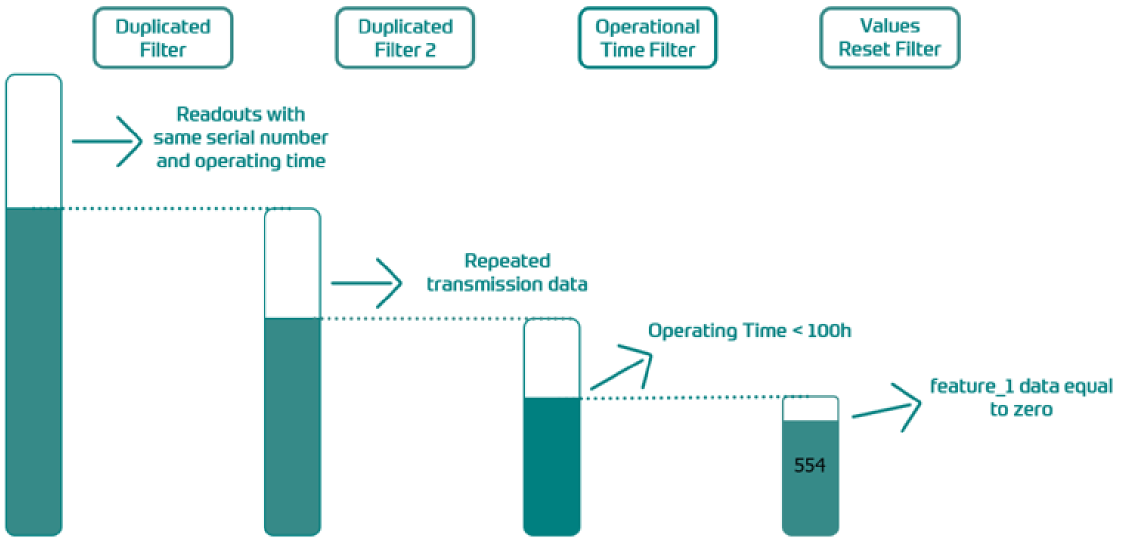
The Kolmogorov-Smirnov (KS) test evaluates the maximum absolute difference between the distribution functions of certain samples. KS is also independent of the distribution, it takes advantage of all the data points in the samples, and it is not sensitive to the direction in which the data is classified Lopes, Reid and Hobson (2007). It was applied to, feature_1, feature_2, and feature_11 to understand the behavior of the data that will compose the dataset. These features were used because they were previously selected as features of interest by the company. Considering that this methodology does not require the data to follow a specific distribution and that the distribution of available data is unknown, it becomes a versatile tool for exploratory analyses. With the KS test, the probability density of the data was checked, in this context, the *kstest* object from the scipy.stats library in SciPy was used, to check if the distribution of the data was uniform. The results can be found in subsection 4.2.1, where the result of the KS test is presented with the data's probability density.

3.4 Data Pre-processing

3.4.1 Data Cleaning

Figure 11 describes the steps that were carried out during data pre-processing. They are discussed below.

Figure 11 – Pre-processing steps.


Duplicated Filter
Duplicated Filter 2
Operational Time Filter
Values Reset Filter
Readouts with same serial number and operating time
Repeated transmission data
Operating Time < 100h
feature_1 data equal to zero
554

Source: Adapted from Monteiro (2022).

3.4.1.1 Duplicated Filter

After the verification described in the previous section, an analysis was initiated to identify if there were duplicate readouts. Consequently, this filter retained only one file from sets of readouts containing the exact same data, such as serial number and operating time.

3.4.1.2 Duplicated Filter 2

In the process of identifying duplicate readouts, some readouts which, although they did not have the same operating time and were not classified as duplicates in the previous filter, had very close operating times, with approximately two hours difference between them. These readouts may be related to a new parameterization version carried out by the testing team to verify if the content of the EEPROM was correct or if the vehicle's behavior was as expected.

Considering that the main objective of the thesis is the health of the transmissions, it is important to consider information where the vehicle is in real operation, that is, it has been conducted in the field, under the operating conditions for which it was designed. Thus, files with the same serial number and small differences in the total vehicle operating time were compared regarding transmission data. Those that had the same information in the collected transmission data were also considered duplicates, and only the one with the higher operating time was retained.

3.4.1.3 Operational Time Filter

In collaboration with the ZF technical team, a threshold of time equal to or greater than 100 hours of the vehicle's operating time was defined. This is because vehicles with little operating time would not provide information about transmission wear, not contributing to the conducted study and potentially resulting in noisy information.

3.4.1.4 Values Reset Filter

Additionally, when the technician performs two readouts, one before and another after resetting the data, this reset results in a second readout not identified in the duplicate filter because their files' content is different since the latest one had its memory erased. Applying this filter involved querying the data for a total transmission operating time equal to zero. Thus, this filter was also added ZF (2024), Monteiro (2022).

3.4.2 Feature Selection

The second part of the data preprocessing involves selecting the features that were used to train and test the models. They need to represent the problem domain well and facilitate the interpretability of the model. Thus, initially, the ZF technical team identified some features that would be important to be evaluated and selected some of them as features for the model. After defining these relevant features, the process of extracting data from JSON files was initiated.

Considering that the collected data were in different formats, such as events, discrete intervals, tables, or parts, not all information could be used directly as features. Therefore, preprocessing these data was necessary. Some data were combined based on key factors, such as identifying the most important intervals for analysis. In addition, Key Performance Indicator

(KPI) techniques were investigated, and, based on the available data and the company's needs, an important KPI feature was implemented. The calculations will not be shown due to the company's data protection policy. This is represented as feature_2 in Figure 18 in subsection 4.4.1, highlighting its importance in the model decision.

Data with little or no variation, such as data with the same classification ("True" or "False") in all files, for example, were removed as they would not provide relevant information for the model. After this step, a total of 32 features remained.

To reduce the dimensionality of the dataset, the correlation matrix was used. As explained in subsection 2.4.3, correlation coefficients range from -1 to +1; magnitudes greater in absolute value indicate a stronger association. Positive values indicate a direct relationship, while negative values indicate a negative or inverse relationship. Thus, the Pearson correlation coefficient was employed to identify the linear correlation among the features. This is the most commonly used coefficient in correlation calculations, according to Stevens (1946).

The correlation matrix was applied to the data using the $corr()$ method from Pandas. Data with a correlation greater than the absolute value of 0.75 were removed, as they would provide redundant information to the model, potentially impairing its interpretation. This approach aims to enhance the model's interpretability and reduce computational power without compromising results as subsection 2.4.3.

After this process, 21 features remained, which were used for training and testing the models. The Table 3 subsection 4.2.3 shows the number of features during the gathering and preprocessing data, and how many remained afterward.

3.4.3 Unbalanced Data

Once the dataset had been formatted and prepared for the model's training and testing stages, an analysis was carried out to check the balance of the data in relation to the labels. This verification was crucial to check that there was an equal distribution between vehicles in Categories 0 (False) and 1 (True) so that the ML algorithms could learn about both.

This is an important approach because, according to subsection 2.4.4, unbalanced data can lead the algorithm to produce biased results towards the majority category. Furthermore, it could influence the choice of algorithms used, as some may not handle this type of dataset very well. The results are presented in section 4.3.

3.4.4 Feature Scaling

Feature scaling is an essential part of pre-processing, as it has a significant impact on scale-sensitive algorithms such as SVM and KNN. This process prevents features with different scales from unduly dominating the learning process.

In this thesis, the chosen method for scaling the data was standardization because it is less sensitive to outliers compared to normalization methods, ensuring that the scale of the data is not defined by extreme values Borriello *et al.* (2024). Additionally, the data used in the study did not have a uniform distribution, as presented in 4.2.1, making the use of this method more appropriate. Considering that machine learning algorithms do not handle missing values well, the approach proposed by Géron (2022) was employed, in which the missing values were replaced with the median of the respective column.

According to Géron (2022), replacing missing values with the median value is less sensitive to outliers than replacing them with the mean. If there are extreme values in a given column, the median value is less affected than the mean, ensuring a more robust replacement. In addition, using the median preserves the distribution of the data as much as possible. This means that replacing missing values does not drastically alter the way the data is distributed, which can be important for maintaining the integrity of the data.

For the application, the Standard Scaler method from Scikit Learn was employed with the *make_pipeline* object from the sklearn.pipeline library. In this case, preprocessing, i.e., the Standard Scaler is applied first, followed by the chosen ML model, thus establishing a chained sequence of transformations and a final estimator. This method proves highly beneficial when the goal is to organize and streamline the workflow. After the preprocessing stage, a total of 554 files were available.

3.5 Model Selection and Training

Given that the company provided information regarding the target values (labels) for the specified problem, the initial step was investigating the most suitable supervised methods.

Model selection is one of the crucial steps in ML, requiring experimentation with the data. Considering the limited amount of data available and the multitude of ML algorithms that could be used, three models were selected and optimized to understand their behavior concerning the data.

The method's choice aimed to determine which one produced the best results according to the company's requirements, i.e., correctly classifying the health of the transmission and minimizing the number of false negatives.

kNN was chosen because it is a non-parametric model, simple to understand and implement, flexible and efficient on various datasets, as discussed in subsubsection 2.7.1.2, being able to perform well on small datasets. Another method chosen was SVM, as it works well with small training samples, can find the best separation between classes and has tools to deal with unbalanced data, being widely used in classification problems.

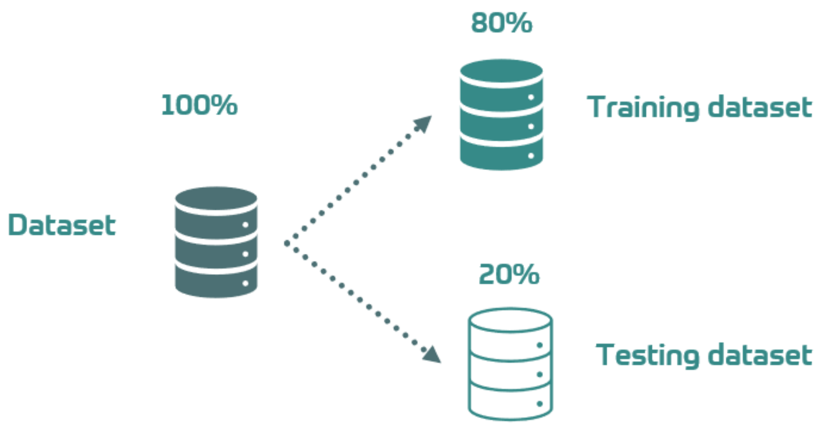
Random Forest was added to the list of models investigated because it is a classification algorithm that calculates the importance of the feature during model training. In addition, it uses several decision trees for classification, each trained independently with a random sample of data and a random subset of resources. Combining the results of these trees through majority voting brings significant benefits, especially in mitigating overfitting, as discussed in subsubsection 2.7.1.1.

3.5.1 Training and Testing

The separation of data into training and testing sets is another crucial step. The training data was used for model optimization, while the testing data was used for validation, ensuring that the model performs well on data it has never seen before.

To split the data, the *train_test_split* object from Scikit Learn was employed. This method takes the specified proportion for data separation and uses a random method for the split. Following the methodology of Géron (2022), the data was split into 80% for training and 20% for testing. Figure 12 indicates the steps described before.

Figure 12 – Training and testing data.



Source: Elaborated by the author.

.

3.5.2 Model Optimization and Evaluation

After defining the models and splitting the data into training and testing data, each model was optimized using the training data. The testing data was then validated using the metrics discussed in section 2.8.

Considering the main hyperparameters of each, as described in Chapter 2, the models were optimized. For this, the Grid Search method in combination with cross-validation was implemented. As discussed earlier, it allows evaluation of the model's average performance on different partitions, ensuring that the choice of hyperparameters is more robust and generic. To split the data during cross-validation, the k-fold cross-validation object from the Scikit-Learn library was used.

For the study conducted in this thesis, the training data was divided into 10 folds using k-fold cross-validation, a value commonly indicated in references, such as Stone (1974). Additionally, Scikit-Learn's shuffle method was used to shuffle the data during separation in order not to use the information in the order available in the dataset, reducing the possibility of introducing bias in the training set. F_1 score results were employed to evaluate if the best hyperparameters returned coherent values, that is, that they were close to each other, in all the cross-validation parts. This metric was chosen because, as discussed in section 2.8, it establishes a direct relationship between Precision, which measures the accuracy of positive predictions, and recall, which measures the model's ability to find all positive cases. F_1 score ranges from 0 to 1, where values close to 1 show better performance, while values close to 0 suggest poor performance. In section 4.5, the results for this analyses can be found, validating the chosen hyperparameters.

In the case of the Random Forest algorithm, in addition to optimizing the hyperparameters, the $feature\_importances$ attribute of Scikit-learn was used, which, according to ScikitLearn, indicates the importance of features during model training. This enables the exclusion of less relevant features for the model. The hyperparameters were adjusted beforehand because the goal was to find the best possible model for the dataset before proceeding with training and evaluating feature importance. This approach is because the hyperparameters have a considerable impact on the results of the model, so the importance of the features could be altered for a model that has not been previously optimized. The results are presented and discussed in subsection 4.4.1.

CHAPTER

4

RESULTS AND DISCUSSIONS

THIS chapter presents the results of the methodologies used above, including the gathering data, exploration, and pre-processing stages, as well as model training. In addition the results of model optimization are presented. Metrics such as confusion matrix, F_1 score, and AUC-ROC curve are applied to identify the best-performing model for the available dataset. In addition, the limitations encountered during development are discussed.

4.1 Gathering Data and Pre-processing

Table 1 presents the results related to the data available after the Gathering data and Pre-processing.

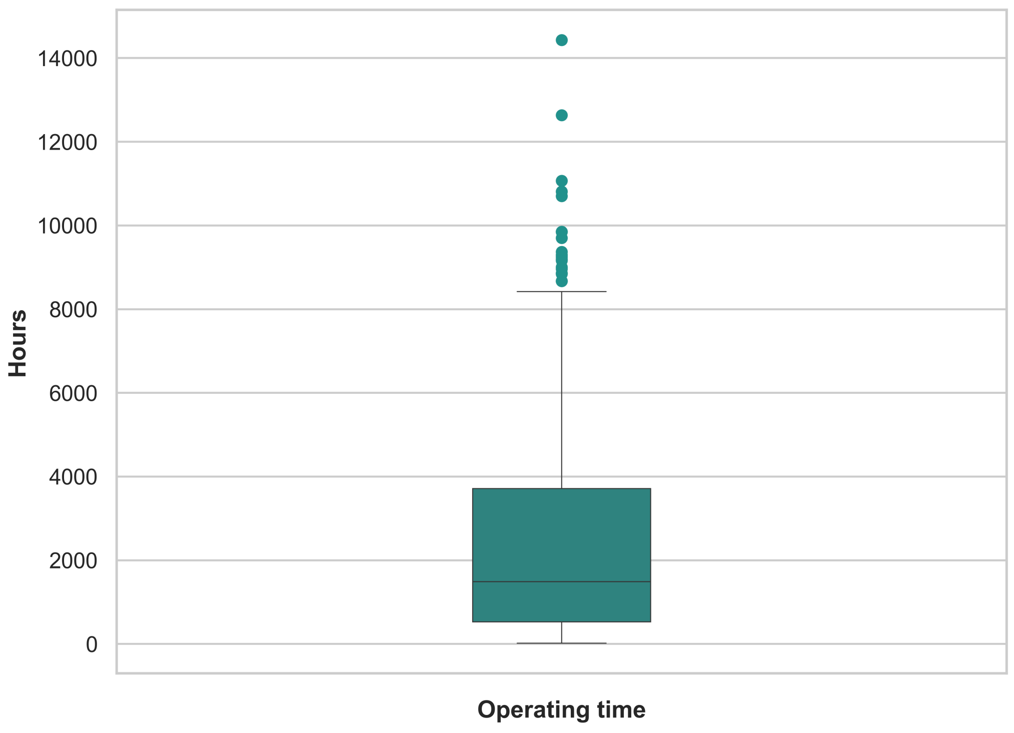
Table 1 – Number of readouts after gathering and preprocessing data.

Stage	Before	After
Gathering data	3027	1811
Pre-processing	1811	554

During the stages described in Table 1, a total of 82% of the data was removed. Of this total, 40% was removed during the gathering data step, indicating a problem in the quality of the provided data, such as incorrect formatting, and duplicate records as discussed in subsection 2.4.1 and 42% during the pre-processing stage.

Thus, 554 files were used in the training and testing stages of the models. Within this set, there are 360 files with a unique serial number (that is, 360 vehicles) and 194 readouts with repeated serial numbers. Although some of these files come from the same vehicle, they exhibit significant variability in operation times, spanning intervals such as 200 hours and 2000 hours of operation, for example. This temporal variation allows transmission to be analyzed in different operating periods. Figure 13 shows the the distribution of the operating time in the whole dataset.

Figure 13 – Operating time using the whole dataset.



Source: Elaborated by the author.

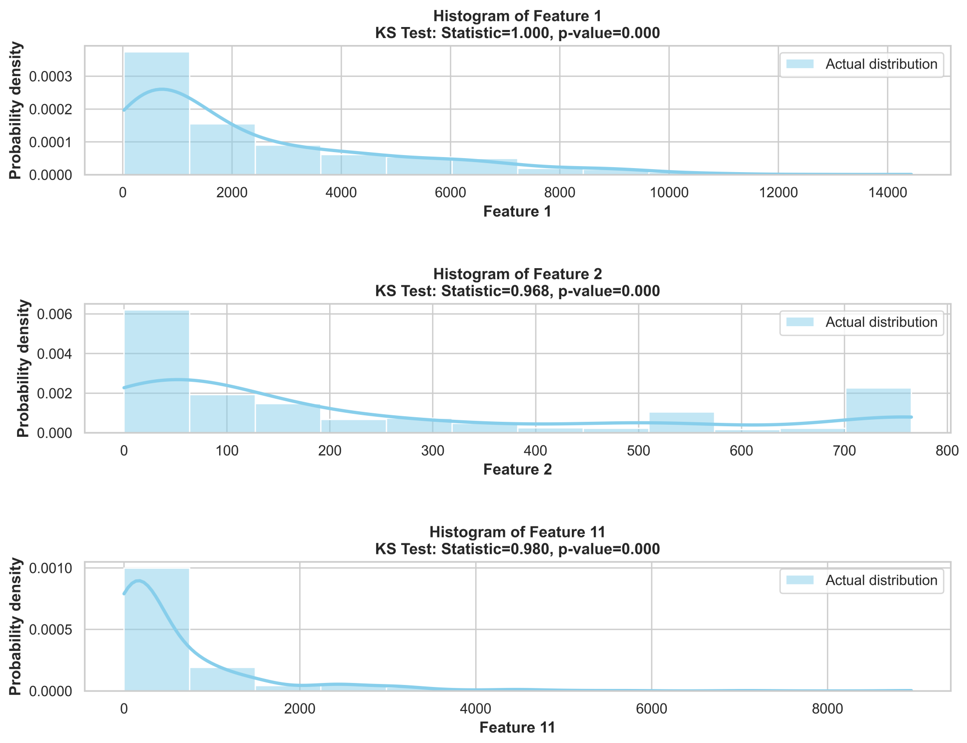
To address the possibility of bias that data at different operation times may cause, techniques such as checking class balance, cross-validation, k-folds, and shuffling were applied, as described in subsection 3.5.1.

4.2 Data Exploration

4.2.1 Distribution of Data

Given a dataset made up of 21 features, features_1, features_2, and features_11 were selected for analysis in terms of their distribution. As discussed in subsection 3.4.4, the choice of scale type is crucial and depends on the nature of the data distribution. This understanding directly influences the definition of the appropriate type of scale to be used. Figure 14 shows the results.

Figure 14 – Distribution of data.



Source: Elaborated by the author.

As the data does not have the same probability density, and the value $p-value$ is zero for all features tested, the hypothesis that the data follows a uniform distribution can be rejected. Probability density refers to the measure of how often values occur in a distribution. If some values occur with greater probability than others, this indicates that the distribution is not uniform, requiring the application of techniques such as adjusting class weights, choosing models capable of dealing with this type of data and applying a standardization method suitable for a non-uniform distribution.

4.2.2 Missing Data Statistics

As shown in Table 2, the dataset had 18 missing data distributed among the features. To obtain these results the *dataprep.eda* module was used, as discussed in section 3.3 . To solve this problem, the median was used in the column in question so as not to alter the distribution of the data, as discussed in subsection 3.4.4.

Table 2 – Missing data statistics.

Missing data	18
Missing data (%)	0.15

4.2.3 Feature Selection

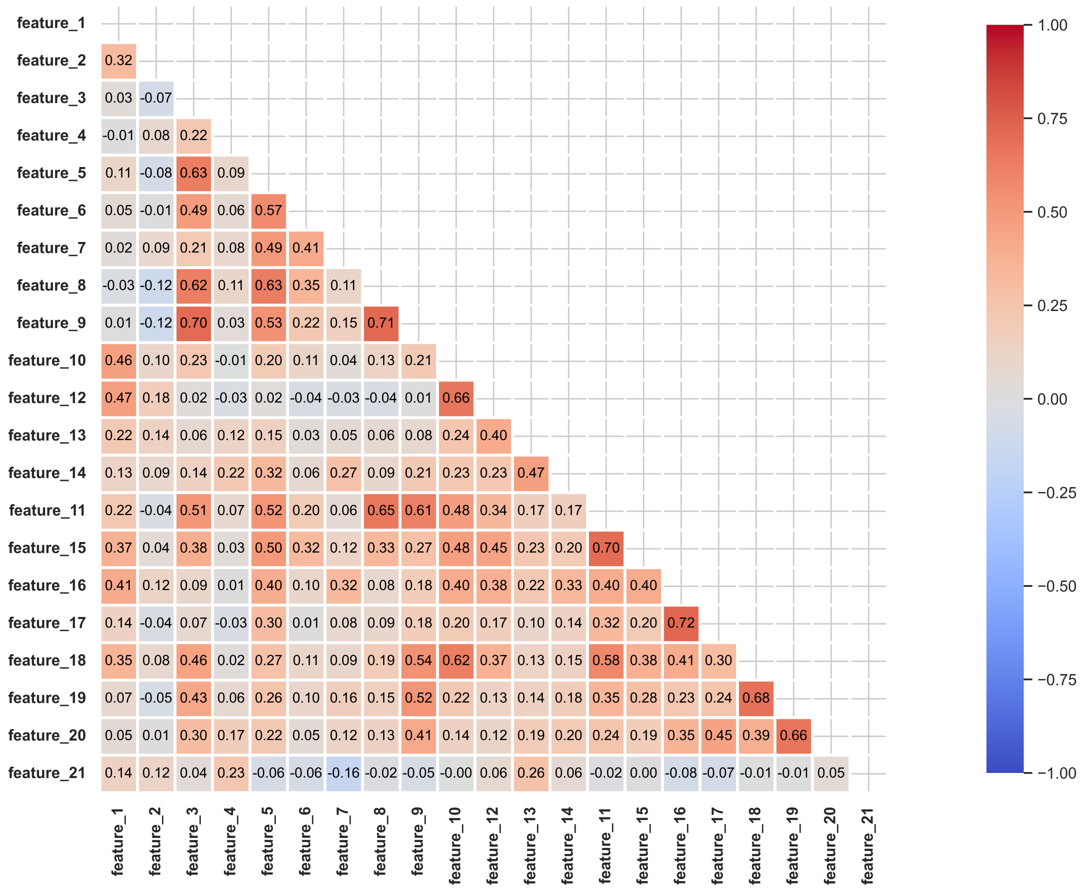
As described in subsection 3.4.2, 10 features were removed as they showed no variation in the data, leaving a total of 32 features. To further reduce the number of features, a correlation matrix was utilized, resulting in a final set of 21 features. Table 3 presents the results.

Table 3 – Number of features.

	Before	**After**	**Reason**
Feature Selection	42	32	Constant features
	32	21	High correlation to other features

Figure 15 displays the outcomes of the final correlation matrix, indicating that the 21 selected features do not exhibit high correlation among themselves, thus avoiding information redundancy for the model. As mentioned in subsection 3.4.2, the threshold used was 0.75.

Figure 15 – Correlation linear between features.

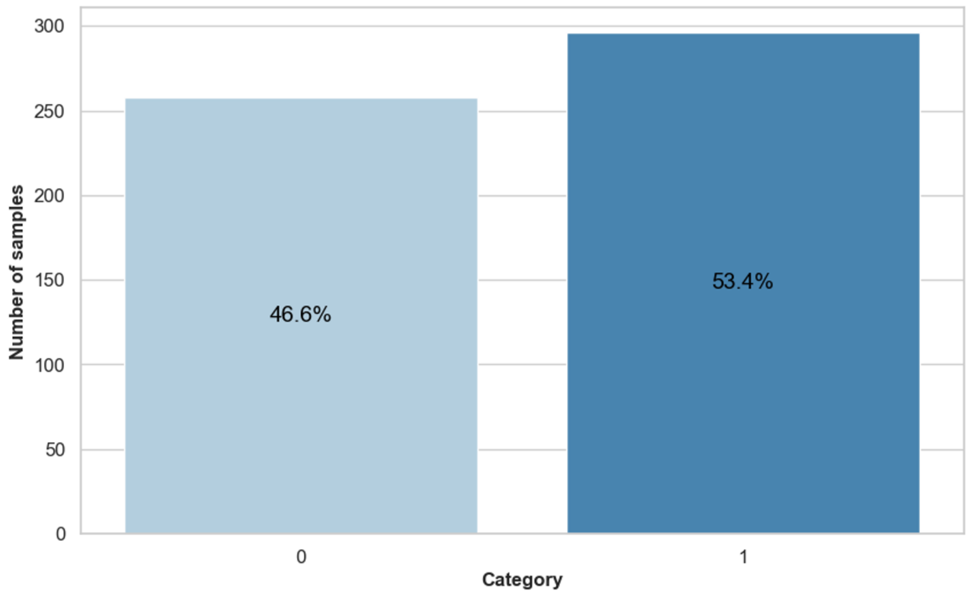


Source: Elaborated by the author.

4.3 Unbalanced Data

According to Figure 16, 46.6% of the data corresponds to Category 0 (False), while 53.4% is associated with Category 1 (True). Therefore, the dataset contains more information about vehicles that experienced issues than about healthy vehicles. Considering that the files typically come from customers who have experienced transmission problems and have sought support from the ZF technical team, it was expected that the dataset would consist of a greater number of instances from Category 1.

Figure 16 – Class distribution.



Source: Elaborated by the author.

As discussed in subsection 3.4.3, the presence of an imbalance in the dataset can lead to results biased towards the majority class, and some algorithms are sensitive to datasets with label imbalances. Although the differences between the classes are not extreme and maybe the models could handle it, due to the limited samples available for training and testing, techniques were employed to deal with the imbalance during the model training. This approach was used to ensure that the models was trained and evaluated accurately, even in the presence of data limitations.

4.4 Model Selection and Training

By the methodologies outlined in section 3.5, the models were tested and optimized using the 21 features for subsequent comparison and evaluation using the metrics discussed in section 2.8.

4.4.1 Random Forest Optimization

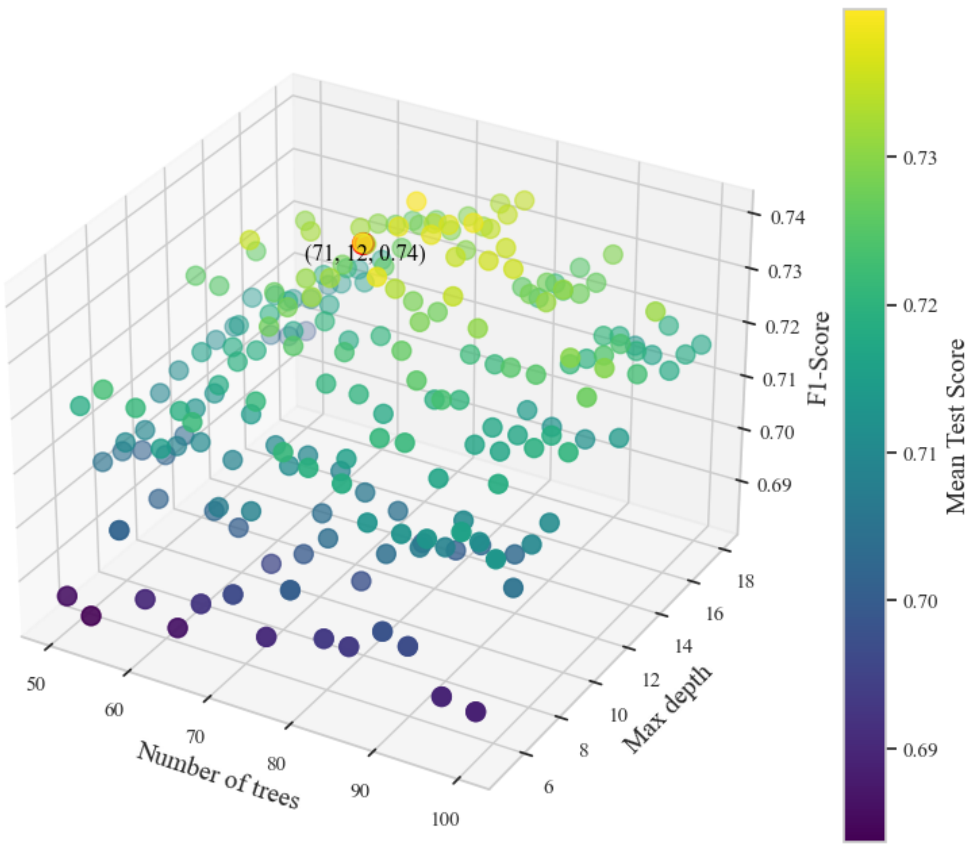
4.4.1.1 Hyperparameters

As previously mentioned in subsubsection subsubsection 2.7.1.1, the most relevant hyperparameters for this model are the number of estimators and the depth of the mask. The number of estimators controls the count of trees used during the process, while the depth of the mask determines the maximum depth for each tree.

As discussed in subsection 3.5.2, GridSearch, combined with cross-validation, was applied to optimize the model's hyperparameters, aiming to find the combination that maximized the F_1 score performance metric. As mentioned in section 2.8, the F_1 score takes into account both precision and recall, providing a balanced evaluation of performance, especially in situations where there is an imbalance between classes, as addressed in this work.

In this context, the F_1 is specially relevant, as it is crucial to accurately identify positive instances (precision) and minimize false negatives. The choice of the F1 score as the evaluation metric aims to achieve a robust balance between positive classification capability and minimal classification errors, thereby making our model more adept at addressing the nuances of the problem. Figure 17 presents the obtained results, making the data easier to understand.

Figure 17 – Random Forest Hyperparameters.



Source: Elaborated by the author.

Figure 17 shows the distribution of the F_1 score when the number of trees and their maximum depth are varied. As highlighted by the orange dot, the best F1 score corresponds to 71 estimators and 12 masks. Thus, the model was tested using these hyperparameters.

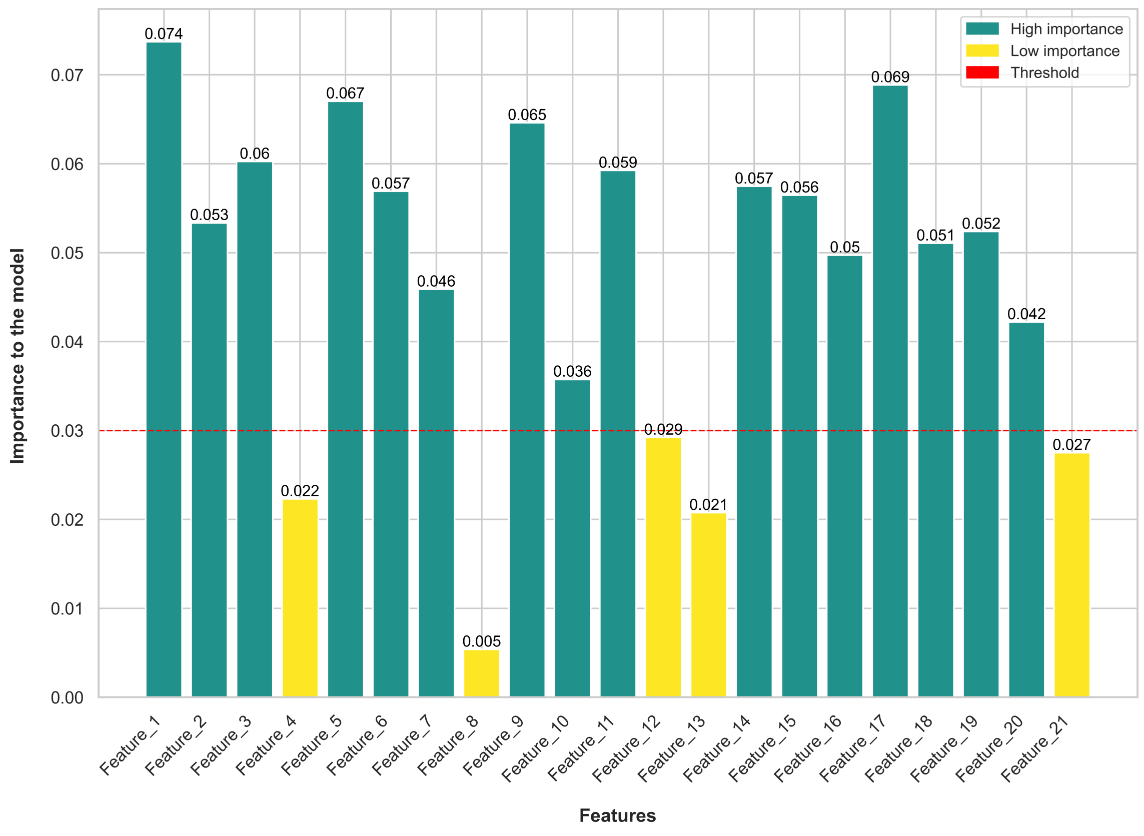
It is observed that the values found during the previously described process are close to those defined in all cross-validation splits, maintaining cohesion among them. As a result, the model demonstrates consistency in its performance, with a range between 0.69 and 0.74.

4.4.1.2 Importance of Features

Given that the random forest provides the ability to calculate feature importance during training, as outlined in subsubsection 2.7.1.1, this will also be a metric for optimizing the model. To achieve this, the $feature\_importances$ object from Scikit-Learn was employed.

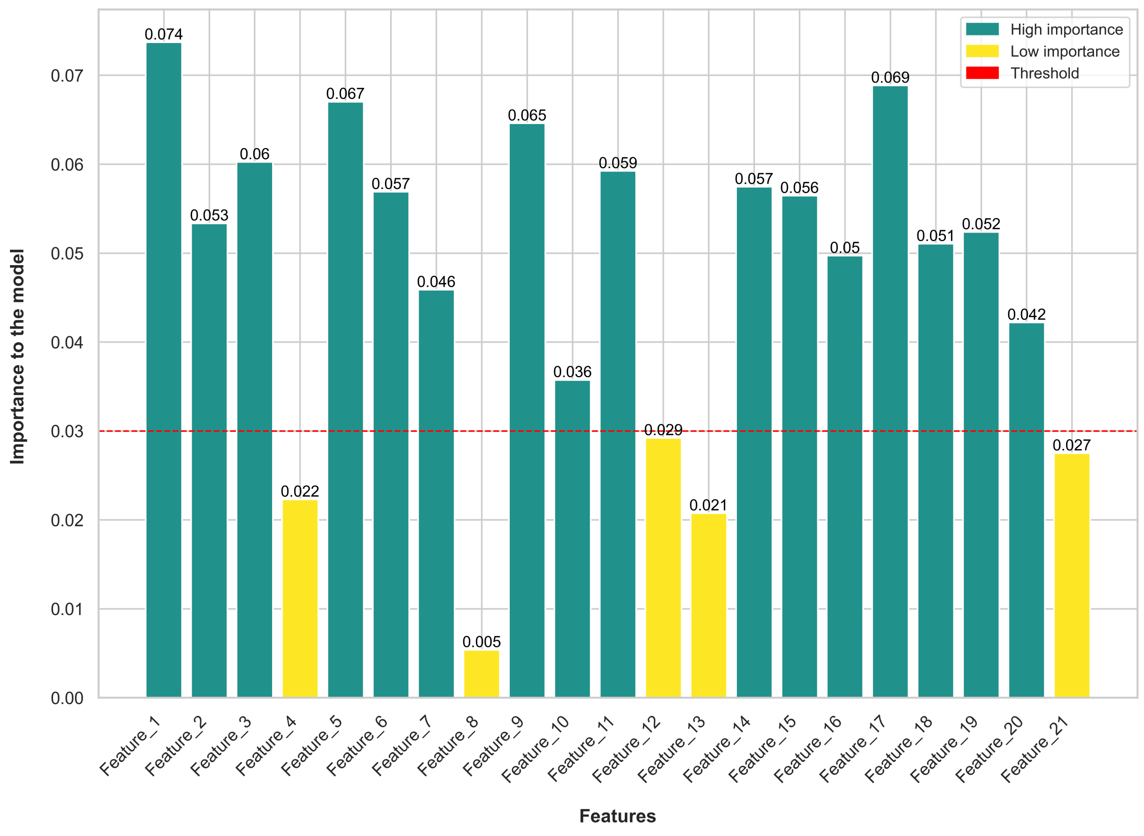
Figure 18 presents the results of feature importance during the model training. In collaboration with the technical team from ZF, a threshold of 0.03 was defined after some analyses to further reduce the dataset's dimensionality. Thus, features with importance less than or equal to 0.03 were discarded, namely, feature_4, feature_8, feature_12, feature_13, and feature_21. Figure 19 illustrates the results after the removal of these features.

Figure 18 – Importance of features to model with all features.



Source: Elaborated by the author.

Figure 19 – Importance of features to model after removal of features.



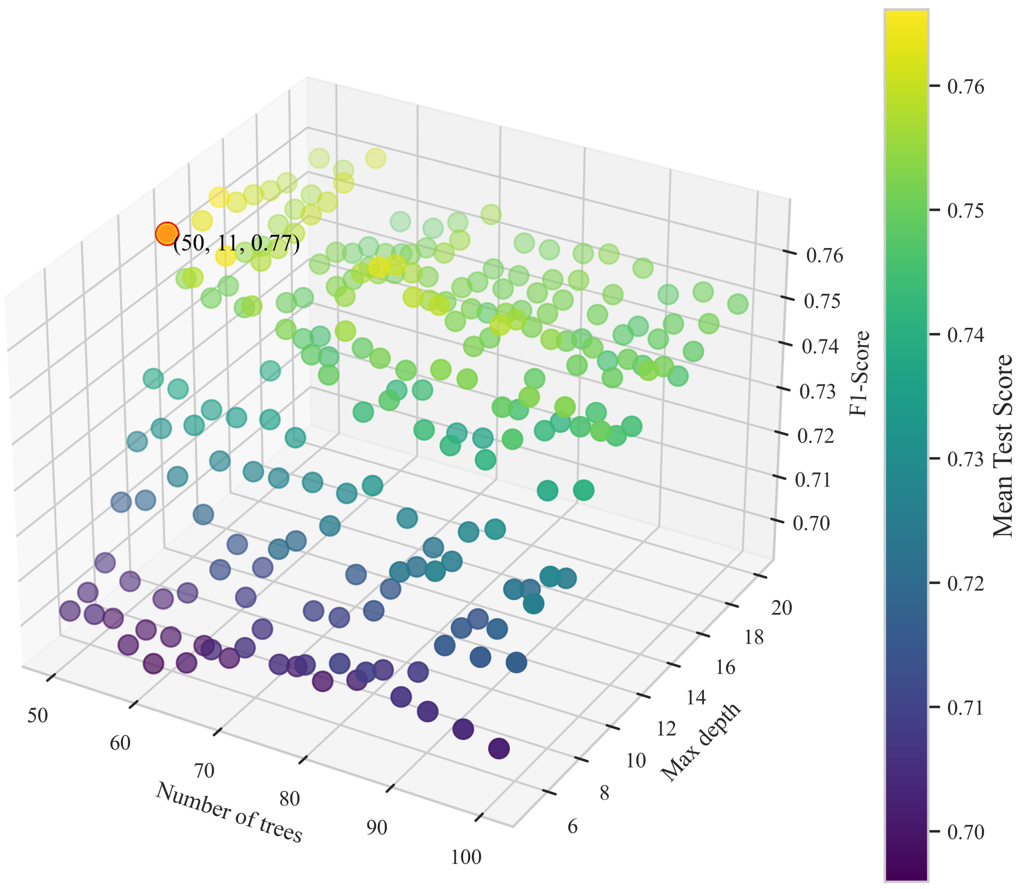
Source: Elaborated by the author.

Analyzing figure 19, it is evident that feature_1 and feature_5 have gained greater importance for the model after the removal less important features. This can be attributed to a more accurate representation of the data, removing noisy features or to a reduction in the complexity of the model, as in subsection 2.4.3. As some features have been removed, 16 were left.

4.4.1.3 Final Model

After the step described in subsubsection 4.4.1.2, the model was reoptimized as detailed in section subsubsection 4.4.1.1, as the removal of features could impact the results of previously assigned hyperparameters. Therefore, for the final model, the number of trees was 50, and the maximum depth was 11, setting an F_1 score of 0.77 as ilustrated in Figure 20.

Figure 20 – Random Forest hyperparameters after removing the features.



Source: Elaborated by the author.

4.4.2 KNN Optimization

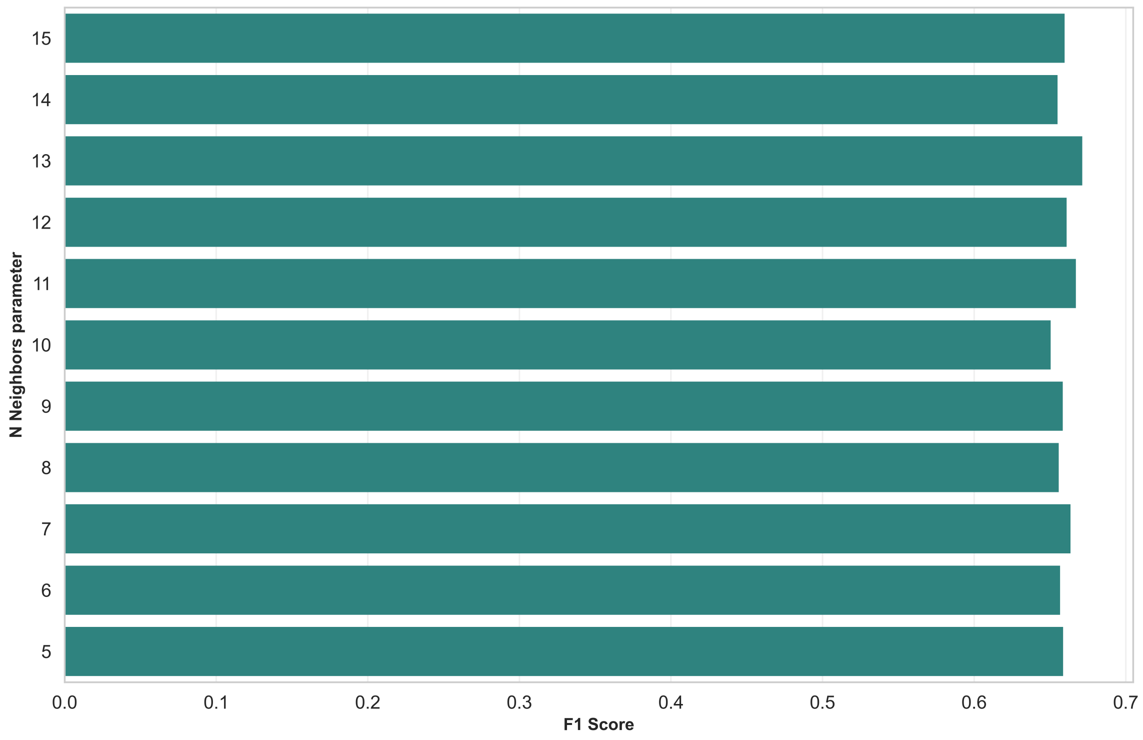
4.4.2.1 Hyperparameters

As described in section subsubsection 2.7.1.2, for the KNN, the number of neighbors is the most important hyperparameter to be optimized, as it affects the model's generalization capability. If K is too small, between 1 and 3, for example, the model may be more sensitive to noise because the assigned class is only related to very close neighbors. However, if the value of K is too large, the model may overfit to training data and not perform well for unseen data.

Therefore, the techniques described in subsection 3.5.2, and discussed in the subsubsection 4.4.1.1, were applied to the training data to find the number of neighbors with the best F_1 score. It is worth mentioning that, as discussed subsection 2.4.1, similar to SVM, this model also has high sensitivity to imbalanced datasets.

Thus, the objects $weights = distance$ and $p = 3$ from Scikit-Learn were used. According to Scikit-Learn, the $weights$ parameter instructs the algorithm to weigh distances between datapoints inversely to the distance, meaning closer instances to the query point will have higher weight in the voting than farther instances. As for the hyperparameter $p = 3$ indicates that the metric used to calculate this distance was Minkowski. As discussed in section subsubsection 2.7.1.2, Minkowski is a generalization of Euclidean distance. Furthermore, this method is versatile and can adapt to any type of distribution, efficiently handling the non-uniform distribution of the data used in this thesis. Figure 21 presents the results obtained, using the Grid Search.

Figure 21 – KNN hyperparameter.



Source: Elaborated by the author.

Analyzing Figure 21 one can notice tha the correlation of the number of neighbors with the optimal F_1 score, demonstrating that 13 neighbors yield the best result. Thus, it was used in the evaluation of the final model.

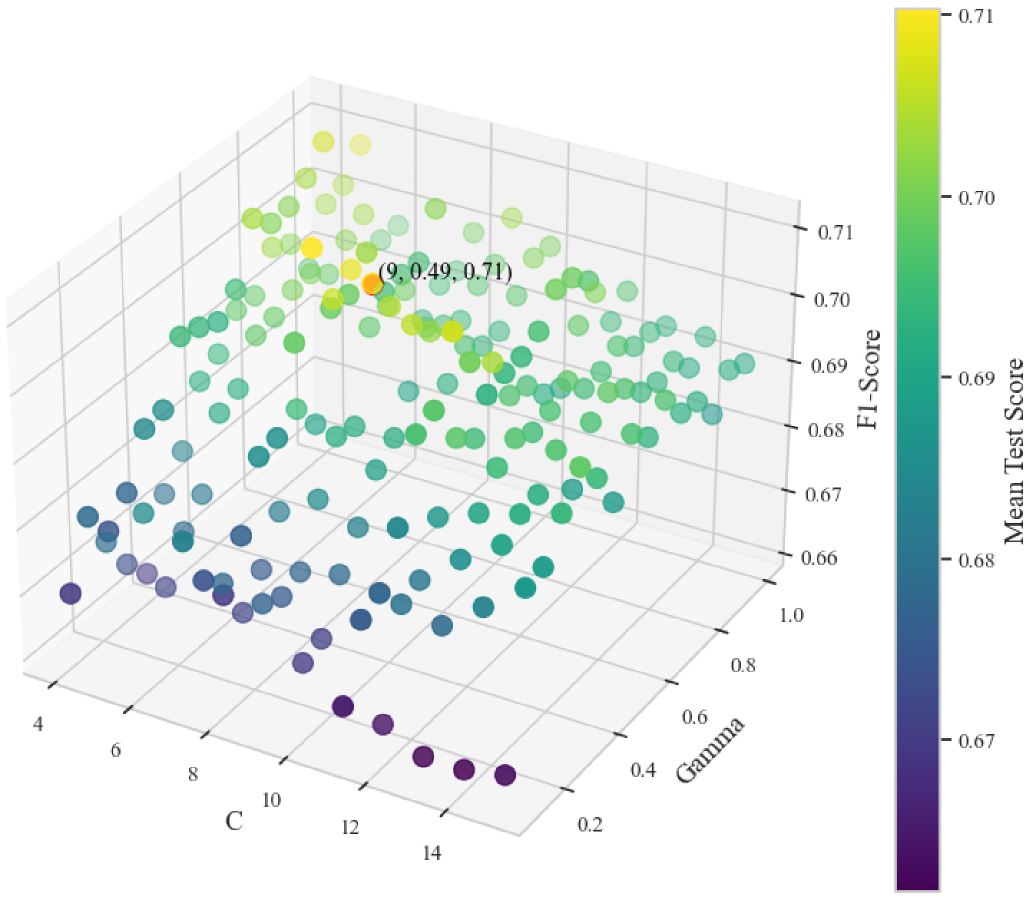
4.4.3 SVM Optimization

4.4.3.1 Hyperparameters

As per subsubsection 2.7.1.3, the most crucial parameters for SVM that was optimized are C, responsible for penalizing classification errors, and *gamma*, which influences how the kernel function affects the decision boundary between classes. To achieve this, the methodology mentioned in section 2.6 and discussed in subsubsection 4.4.1.1, in which grid search in combination with cross-validation is used to maximize the F_1 score, was applied.

Considering that this model is sensitive to dataset imbalance, as discussed in subsection 3.4.3, the $class\_weight = balanced$ parameter was applied during the use of training data. By subsubsection 2.7.1.3, SVM adjusts class weights to compensate for this imbalance and ensure that the model is trained more evenly, taking all classes into account equally, regardless of their frequency. Figure 22 presents the results.

Figure 22 – SVM hyperparameters.



Source: Elaborated by the author.

It is noted that the values of C and *gamma* that provide the best F_1 score are 9 and 0.49, therefore, they were used to train the model and conduct tests to evaluate the model's performance after optimization.

4.5 Evaluating Models

As described in subsection 3.5.1, the data was split, with 80% of 554 readouts presented in Table 1, allocated for training and 20% for testing the models. For model evaluation, test data was utilized, consisting of 112 vehicles.

Table 4 presents the precision, recall, and F_1 score information for the models studied in this work.

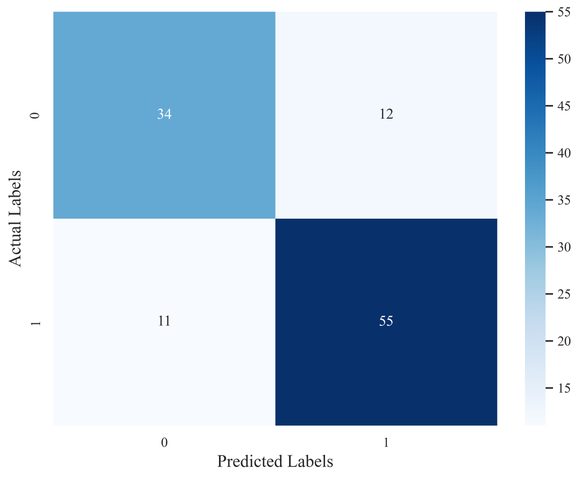
Table 4 – Evaluating models.

Model	**Precision**	**Recall**	**F1-score**
Random Forest	0.794	0.795	0.794
KNN	0.755	0.75	0.751
SVM	0.719	0.714	0.716

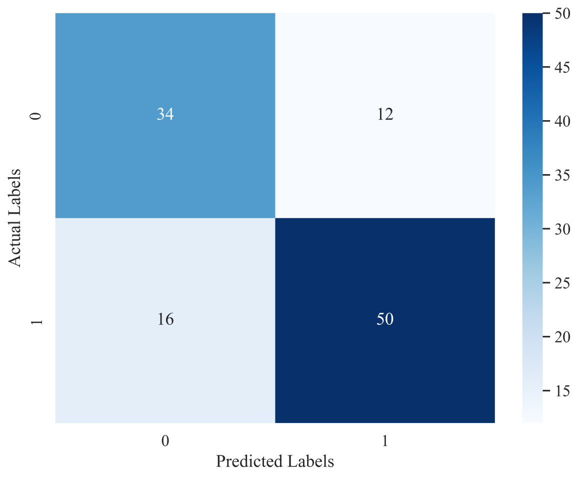
Analyzing Table 4, it is observed that all models achieved a balance between precision and recall. This suggests good performance in identifying both positive and negative instances during data classification.

By section 2.8, when analyzing a model's performance, the confusion matrix is crucial as it provides information about the number of FP, FN, TP, and TN, enabling a comprehensive analysis and facilitating result interpretation. Figure 23 shows the results.

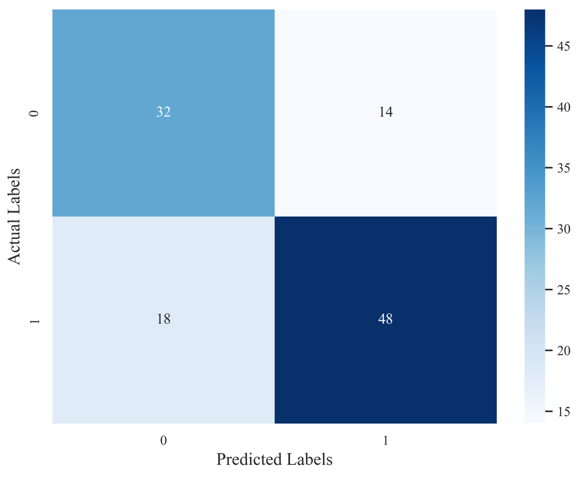
Figure 23 – Confusion matrix to models.



(a) Random Forest.



(b) KNN.



(c) SVM.

Source: Elaborated by the author.

Figure 23a illustrates the Random Forest results, where 34 vehicles were correctly classified as True Negatives (TN) and 55 as True Positives (TP). Moreover, 23 (20% of test data) vehicles were misclassified, resulting in 11 False Negatives (FN) and 12 False Positives (FP).

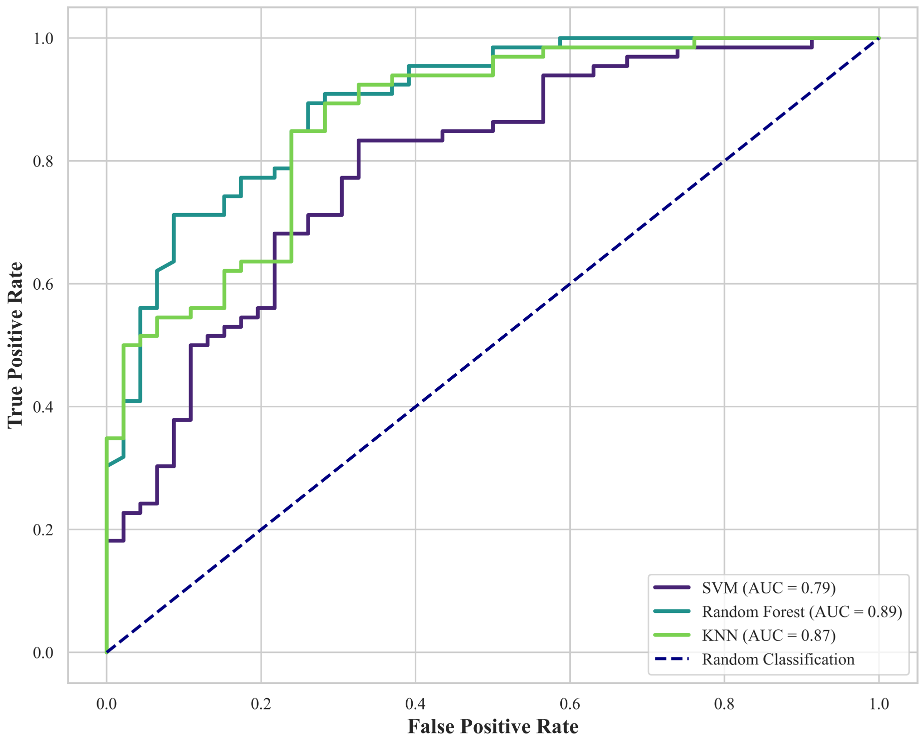
In Figure 23b, the KNN results are presented. 84 vehicles were accurately classified, including 34 as TN and 50 as TP. However, there were misclassifications, leading to 16 FN and 12 FP, indicating a percentage of 25% of test data.

And finally in Figure 23c, the SVM results indicate 80 vehicles are correctly classified, with 32 as TN and 48 as TP. Similar to the other models, 32 (28.6% of test data) vehicles were misclassified, resulting in 18 FN and 14 FP.

Another widely employed metric in model evaluation is the AUC-ROC (Area Under the Receiver Operating Characteristic curve), as detailed in section 2.8. The AUC-ROC quantifies the overall quality of the model in making classifications and distinguishing between Category 0 (False) and Category 1 (True). The results of the AUC-ROC range from 0 to 1, with values closer to 1 considered indicative of superior performance.

Figure 24 displays the AUC-ROC results for the three models, enabling a direct comparison of their performances. It is emphasized that higher values in the AUC-ROC reflect a superior ability of the model to balance sensitivity and specificity. This visual analysis makes it easier to interpret the effectiveness of each model in discriminating between classes, since the larger the area under the curve, the better the model's ability to classify the data.

Figure 24 – AUC-ROC curve.



Source: Elaborated by the author.

Examining the ROC curve, it becomes clear that the Random Forest presents an AUC of 0.89, indicating its superiority compared to SVM and KNN, as assessed by this metric.

4.6 Overall Discussions

When analyzing the results of the evaluated metrics for model assessment, as discussed earlier, a consistent balance is observed in the precision and recall outcomes, as evidenced in Table 4. This balance suggests that the models are making classification decisions in a well-balanced manner. As highlighted in subsubsection 4.4.1.1, it is crucial not only to achieve correct classifications but also to reduce the number of FN. Therefore, seeking a balance between these metrics is ideal, aiming at minimizing FN and correctly classifying instances.

Although all models exhibited similar behavior, the Random Forest stands out by delivering the best results in all metrics analyzed in this study. It achieved an F_1 score of 0.794, while the KNN recorded an F_1 score of 0.76, and the SVM, a value of 0.716. Furthermore, when examining the confusion matrix, Random Forest presented 11 false negatives, compared to 14 for SVM and 14 for KNN, respectively. Finally, the Random Forest exhibited an AUC of 0.89, surpassing the values of 0.87 for KNN and 0.79 for SVM. Therefore, among the models evaluated with the dataset used in this study, the Random Forest consistently demonstrates superior results, supported by the analyzed metrics.

Even though the models performed well in the metrics analyzed, it is essential to acknowledge the limitations in the currently available dataset. One of these limitations lies in the quality of the labels assigned to the data. There is no guarantee that instances classified as "False" really dont have problems, because these may have failed after obtaining the label. Another significant limitation is the scarcity of data available for training and testing the models. The limited quantity of training examples may have constrained the models' ability to learn and generalize various characteristics necessary for accurate classification of transmission health. The quality of the label, coupled with the amount of data provided, may account for the high number of misclassifications.

Thus, the model requires greater reliability regarding the labels assigned to the data, as well as an expansion of the dataset to be analyzed and validated before being employed in industrial applications. Currently, this model can be used in the dashboards developed within the data analysis team as a way to demonstrate to clients that by providing more data, this model can be applied in their daily operations, providing crucial information of the evaluated transmission.

CHAPTER

5

CONCLUSIONS AND PROSPECTS

THIS chapter provides the final conclusions, discussions and general remarks about the presented study. It also gives a perspective of future improvements and research branches which can be derived from the proposed methodology.

5.1 Conclusion

The aim of this thesis was to use field data from vehicle EEPROMs to compare machine learning algorithms and identify the most effective one for predicting powershift transmission failures, thus indicating the state of health of this component. One of the main objectives was to find an algorithm that could minimize the number of undetected transmission problems, i.e., cases in which the transmission is erroneously indicated as healthy. The overall goal was to develop a tool for effective transmission monitoring, aiding fault detection, contributing to vehicle maintenance and improving vehicle safety during operation.

The study revealed that the models evaluated showed a well-balanced performance in terms of precision and recall, suggesting effective classification decisions. The Random Forest model was notably superior, obtaining the highest scores in all metrics, including an F_1 score of 0.794 and an AUC of 0.89, thus establishing itself as the most effective model in this study.

Despite these positive results, it is necessary to recognize the limitations of the available data set. The quality of the labels assigned to the data is uncertain, which represents a risk of incorrect classification. Instances marked as "False" may not be labeled accurately, reflecting possible problems in the quality of the data. In addition, the limited amount of data available to train and test the models may have restricted their ability to learn and generalize the features needed for accurate transmission health classification. This data scarcity could be a significant factor behind the high number of misclassifications observed.

At the current stage, the model can be used in dashboards within the data analysis team to demonstrate to clients the benefits of collaborative data provision. However, it is essential to use a more extensive data set to validate this tool. By improving the quality and increasing the volume of data in the set, the model can be validated more thoroughly, which will result in greater reliability. Subsequently, there is potential for its deployment in industrial applications, providing crucial information on the integrity of the vehicle's transmission, this can reduce maintenance costs and machine downtime.

5.2 Future Works

For future work, it is recommended to expand the dataset to validate the effectiveness of the proposed tool. This implies obtaining more data, preferably with a higher accuracy label, to strengthen the reliability of the model. After this step, it is feasible to employ the same pipeline developed in this project to implement tools that assess different types of transmission. Furthermore, it is suggested to explore the probability function *predict_proba*() available in scikit-learn to visualize the classification decision boundary of the Random Forest model. Adjusting precision and recall parameters will provide valuable insights into the model's behavior, contributing to possible optimization. Finally, it is recommended to explore other ensemble methods, such as Gradient Boosting Machines (GBM), which have stood out in the field of data science. Evaluating the data from this perspective may offer a more comprehensive understanding, as ensemble methods have shown promising results compared to the non-parametric methods studied in this work.

BIBLIOGRAPHY

ALI, P. J. M.; FARAJ, R. H.; KOYA, E.; ALI, P. J. M.; FARAJ, R. H. Data normalization and standardization: a technical report. **Mach Learn Tech Rep**, v. 1, n. 1, p. 1–6, 2014. Citation on page 22.

ALSAGRI, H.; YKHLEF, M. Quantifying feature importance for detecting depression using random forest. **International Journal of Advanced Computer Science and Applications**, Science and Information (SAI) Organization Limited, v. 11, n. 5, 2020. Citations on pages 27 and 28.

ALVAREZ, S. A. An exact analytical relation among recall, precision, and classification accuracy in information retrieval. **Boston College, Boston, Technical Report BCCS-02-01**, p. 1–22, 2002. Citations on pages 32 and 33.

AMBARWARI, A.; ADRIAN, Q. J.; HERDIYENI, Y. *et al.* Analysis of the effect of data scaling on the performance of the machine learning algorithm for plant identification. **Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)**, v. 4, n. 1, p. 117–122, 2020. Citation on page 22.

BARTZ, E.; BARTZ-BEIELSTEIN, T.; ZAEFFERER, M.; MERSMANN, O. **Hyperparameter Tuning for Machine and Deep Learning with R: A Practical Guide**. [S.l.]: Springer Nature, 2023. Citation on page 28.

BEGUM, S.; CHAKRABORTY, D.; SARKAR, R. Data classification using feature selection and knn machine learning approach. In: **2015 International Conference on Computational Intelligence and Communication Networks (CICN)**. [S.l.: s.n.], 2015. p. 811–814. Citation on page 29.

BERENGUER, L. **Supervised Machine Learning Model to Predict Health Condition of Construction Machinery Transmissions**. 2021. Citation on page 17.

BORRIELLO, P.; TESSICINI, F.; RICUCCI, G.; FROSINA, E.; SENATORE, A. A fault detection strategy for an epump during eol tests based on a knowledge-based vibroacoustic tool and supervised machine learning classifiers. **Meccanica**, Springer, p. 1–26, 2024. Citations on pages 22 and 40.

BRAIN, D.; WEBB, G. I. On the effect of data set size on bias and variance in classification learning. In: **Proceedings of the Fourth Australian Knowledge Acquisition Workshop, University of New South Wales**. [S.l.: s.n.], 1999. p. 117–128. Citations on pages 23 and 28.

BREIMAN, L. Random forests. **Machine learning**, Springer, v. 45, p. 5–32, 2001. Citation on page 26.

CAMPOS, J. R.; COSTA, E.; VIEIRA, M. Improving failure prediction by ensembling the decisions of machine learning models: A case study. **IEEE Access**, IEEE, v. 7, p. 177661–177674, 2019. Citation on page 14.

CAO, H.; NAITO, T.; NINOMIYA, Y. Approximate rbf kernel svm and its applications in pedestrian classification. In: **The 1st International Workshop on Machine Learning for Vision-based Motion Analysis-MLVMA'08**. [S.l.: s.n.], 2008. Citations on pages 30 and 31.

CARVALHO, A. M. d.; PRATI, R. C. Improving knn classification under unbalanced data. a new geometric oversampling approach. In: **2018 International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2018. p. 1–6. Citation on page 23.

CARVALHO, T. P.; SOARES, F. A.; VITA, R.; FRANCISCO, R. d. P.; BASTO, J. P.; ALCALÁ, S. G. A systematic literature review of machine learning methods applied to predictive maintenance. **Computers & Industrial Engineering**, Elsevier, v. 137, p. 106024, 2019. Citation on page 18.

CHEN, Q.; LI, Q.; WU, J.; HE, J.; MAO, C.; LI, Z.; YANG, B. State monitoring and fault diagnosis of hvdc system via knn algorithm with knowledge graph: A practical china power grid case. **Sustainability**, v. 15, p. 3717, 02 2023. Citation on page 28.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, p. 273–297, 1995. Citation on page 30.

DIETTERICH, T. G. Ensemble methods in machine learning. In: SPRINGER. **International workshop on multiple classifier systems**. [S.l.], 2000. p. 1–15. Citation on page 26.

DOMINGOS, P. A few useful things to know about machine learning. **Communications of the ACM**, ACM New York, NY, USA, v. 55, n. 10, p. 78–87, 2012. Citation on page 26.

DURO, D. C.; FRANKLIN, S. E.; DUBÉ, M. G. A comparison of pixel-based and object-based image analysis with selected machine learning algorithms for the classification of agricultural landscapes using spot-5 hrg imagery. **Remote sensing of environment**, Elsevier, v. 118, p. 259–272, 2012. Citation on page 25.

EGAN, J. P. Signal detection theory and roc analysis. **(No Title)**, 1975. Citation on page 33.

ELBASHEER, M. E. E. E.; CORZO, G. A.; SOLOMATINE, D.; VAROUCHAKIS, E. Machine learning and committee models for improving ecmwf subseasonal to seasonal (s2s) precipitation forecast. **Hydrology and Earth System Sciences Discussions**, Göttingen, Germany, v. 2022, p. 1–37, 2022. Citation on page 34.

FARQUAD, M. A. H.; BOSE, I. Preprocessing unbalanced data using support vector machine. **Decision Support Systems**, Elsevier, v. 53, n. 1, p. 226–233, 2012. Citation on page 23.

FATIMA, A.; NAZIR, N.; KHAN, M. G. Data cleaning in data warehouse: A survey of data pre-processing techniques and tools. **Int. J. Inf. Technol. Comput. Sci**, v. 9, n. 3, p. 50–61, 2017. Citation on page 21.

FAWAGREH, K.; GABER, M. M.; ELYAN, E. Random forests: from early developments to recent advancements. **Systems Science & Control Engineering: An Open Access Journal**, Taylor & Francis, v. 2, n. 1, p. 602–609, 2014. Citation on page 27.

FAWCETT, T. An introduction to roc analysis. **Pattern recognition letters**, Elsevier, v. 27, n. 8, p. 861–874, 2006. Citation on page 33.

FLACH, P. A. Roc analysis. In: **Encyclopedia of machine learning and data mining**. [S.l.]: Springer, 2016. p. 1–8. Citation on page 33.

FOODY, G. M. Challenges in the real world use of classification accuracy metrics: From recall and precision to the matthews correlation coefficient. **Plos one**, Public Library of Science San Francisco, CA USA, v. 18, n. 10, p. e0291908, 2023. Citations on pages 31 and 32.

GAO, X.; HOU, J. An improved svm integrated gs-pca fault diagnosis approach of tennessee eastman process. **Neurocomputing**, Elsevier, v. 174, p. 906–911, 2016. Citation on page 25.

GÉRON, A. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow**. [S.l.]: " O'Reilly Media, Inc.", 2022. Citations on pages 40 and 41.

GHAHRAMANI, Z. Unsupervised learning. In: **Summer school on machine learning**. [S.l.]: Springer, 2003. p. 72–112. Citation on page 21.

GHORBANZADEH, O.; SHAHABI, H.; MIRCHOOLI, F.; KAMRAN, K. V.; LIM, S.; ARYAL, J.; JARIHANI, B.; BLASCHKE, T. Gully erosion susceptibility mapping (gesm) using machine learning methods optimized by the multi-collinearity analysis and k-fold cross-validation. **Geomatics, Natural Hazards and Risk**, Taylor & Francis, v. 11, n. 1, p. 1653–1678, 2020. Citation on page 25.

GOPI, S. C.; SUVARNA, B.; PADMAJA, T. M. High dimensional unbalanced data classification vs svm feature selection. **Indian Journal of Science and Technology**, v. 9, p. 30, 2016. Citation on page 23.

GUO, G.; WANG, H.; BELL, D.; BI, Y.; GREER, K. Knn model-based approach in classification. In: SPRINGER. **On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings**. [S.l.], 2003. p. 986–996. Citation on page 29.

HACKELING, G. **Mastering Machine Learning with scikit-learn**. [S.l.]: Packt Publishing Ltd, 2017. Citations on pages 22, 23, 24, and 32.

HADD, A.; RODGERS, J. L. **Understanding correlation matrices**. [S.l.]: Sage Publications, 2020. Citation on page 23.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. H.; FRIEDMAN, J. H. **The elements of statistical learning: data mining, inference, and prediction**. [S.l.]: Springer, 2009. Citations on pages 21, 25, and 26.

HMEIDI, I.; HAWASHIN, B.; EL-QAWASMEH, E. Performance of knn and svm classifiers on full word arabic articles. **Advanced Engineering Informatics**, Elsevier, v. 22, n. 1, p. 106–111, 2008. Citation on page 28.

HOFMANN, M.; NEUKART, F.; BÄCK, T. Artificial intelligence and data science in the automotive industry. **arXiv preprint arXiv:1709.01989**, 2017. Citation on page 18.

HSU, C.-W.; LIN, C.-J. A comparison of methods for multiclass support vector machines. **IEEE transactions on Neural Networks**, IEEE, v. 13, n. 2, p. 415–425, 2002. Citations on pages 22 and 31.

HUANG, J.; HUANG, N.; ZHANG, L.; XU, H. A method for feature selection based on the correlation analysis. In: **Proceedings of 2012 International Conference on Measurement, Information and Control**. [S.l.: s.n.], 2012. v. 1, p. 529–532. Citation on page 22.

IMAM, F.; MUSILEK, P.; REFORMAT, M. Z. Parametric and nonparametric machine learning techniques for increasing power system reliability: A review. **Information**, MDPI, v. 15, n. 1, p. 37, 2024. Citation on page 26.

JAHNKE, P. Machine learning approaches for failure type detection and predictive maintenance. **Technische Universität Darmstadt**, Citeseer, v. 19, 2015. Citation on page 26.

JESMEEN, M.; HOSSEN, J.; SAYEED, S.; HO, C.; TAWSIF, K.; RAHMAN, A.; ARIF, E. A survey on cleaning dirty data using machine learning paradigm for big data analytics. **Indonesian Journal of Electrical Engineering and Computer Science**, v. 10, n. 3, p. 1234–1243, 2018. Citation on page 27.

KUMBURE, M. M.; LUUKKA, P. A generalized fuzzy k-nearest neighbor regression model based on minkowski distance. **Granular Computing**, Springer, v. 7, n. 3, p. 657–671, 2022. Citations on pages 18 and 29.

LEARN scikit. **RBF SVM parameters**. 2024. 2024. Available: <https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html>. Citation on page 31.

LECHNER, G.; NAUNHEIMER, H. **Automotive transmissions: fundamentals, selection, design and application**. [S.l.]: Springer Science & Business Media, 1999. Citation on page 14.

LEUKEL, J.; GONZÁLEZ, J.; RIEKERT, M. Adoption of machine learning technology for failure prediction in industrial maintenance: A systematic review. **Journal of Manufacturing Systems**, Elsevier, v. 61, p. 87–96, 2021. Citation on page 14.

LI, J.; CHENG, K.; WANG, S.; MORSTATTER, F.; TREVINO, R. P.; TANG, J.; LIU, H. Feature selection: A data perspective. **ACM computing surveys (CSUR)**, ACM New York, NY, USA, v. 50, n. 6, p. 1–45, 2017. Citation on page 22.

LIU, Y. H. **Python machine learning by example**. [S.l.]: Packt Publishing Ltd, 2017. Citations on pages 18 and 19.

LOPES, R. H.; REID, I.; HOBSON, P. R. The two-dimensional kolmogorov-smirnov test. Proceedings of Science, 2007. Citation on page 37.

LUQUE, A.; CARRASCO, A.; MARTÍN, A.; HERAS, A. de L. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. **Pattern Recognition**, Elsevier, v. 91, p. 216–231, 2019. Citation on page 33.

MA, X.; MAN, Q.; YANG, X.; DONG, P.; YANG, Z.; WU, J.; LIU, C. Urban feature extraction within a complex urban area with an improved 3d-cnn using airborne hyperspectral data. **Remote Sensing**, v. 15, n. 4, 2023. ISSN 2072-4292. Available: <https://www.mdpi.com/2072-4292/15/4/992>. Citation on page 27.

MAGALHãES, C. **Analysis of Mechanical Transmission Data**. Master's Thesis (Master's Thesis) — École Nationale Supérieure de Mécanique et d'Aérotechnique, 2022. Citation on page 17.

MALEKI, F.; OVENS, K.; NAJAFIAN, K.; FORGHANI, B.; REINHOLD, C.; FORGHANI, R. Overview of machine learning part 1: fundamentals and classic approaches. **Neuroimaging Clinics**, Elsevier, v. 30, n. 4, p. e17–e32, 2020. Citations on pages 19 and 21.

MANTOVANI, R. G.; HORVÁTH, T.; CERRI, R.; JUNIOR, S. B.; VANSCHOREN, J.; CARVALHO, A. C. P. d. L. F. de. An empirical study on hyperparameter tuning of decision trees. **arXiv preprint arXiv:1812.02207**, 2018. Citation on page 28.

MARZBAN, C. The roc curve and the area under it as performance measures. **Weather and Forecasting**, American Meteorological Society, v. 19, n. 6, p. 1106–1114, 2004. Citation on page 33.

MCKINNEY, W. **Python for data analysis: Data wrangling with Pandas, NumPy, and IPython**. [S.l.]: " O'Reilly Media, Inc.", 2012. Citation on page 35.

MITCHELL, T. M. **AI magazine**, v. 18, n. 3, p. 11–11, 1997. Citation on page 18.

MONTEIRO, N. **Data Analytics Applied to Adaptive Automatic Gearboxes**. Master's Thesis (Master's Thesis) — Ecole Nationale Sup´erieure de Mecanique et d'Aerotechnique, 2022. Citations on pages 17, 37, and 38.

MUHAMEDYEV, R. Machine learning methods: An overview. **Computer modelling & new technologies**, v. 19, n. 6, p. 14–29, 2015. Citation on page 21.

MÜLLER, A. C.; GUIDO, S. **Introduction to machine learning with Python: a guide for data scientists**. [S.l.]: " O'Reilly Media, Inc.", 2016. Citation on page 35.

MURPHY, K. P. **Machine learning: a probabilistic perspective**. [S.l.]: MIT press, 2012. Citations on pages 20 and 21.

NAQA, I. E.; MURPHY, M. J. **What is machine learning?** [S.l.]: Springer, 2015. Citation on page 18.

NAVIN, J. M.; PANKAJA, R. Performance analysis of text classification algorithms using confusion matrix. **International Journal of Engineering and Technical Research (IJETR)**, v. 6, n. 4, p. 75–8, 2016. Citation on page 32.

OGUNSANYA, M.; ISICHEI, J.; DESAI, S. Grid search hyperparameter tuning in additive manufacturing processes. **Manufacturing Letters**, v. 35, p. 1031–1042, 2023. ISSN 2213-8463. 51st SME North American Manufacturing Research Conference (NAMRC 51). Available: <https://www.sciencedirect.com/science/article/pii/S221384632300113X>. Citation on page 25.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V. *et al.* Scikit-learn: Machine learning in python. **the Journal of machine Learning research**, JMLR. org, v. 12, p. 2825–2830, 2011. Citation on page 32.

POWERS, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. **arXiv preprint arXiv:2010.16061**, 2020. Citation on page 31.

RAHM, E.; DO, H. H. *et al.* Data cleaning: Problems and current approaches. **IEEE Data Eng. Bull.**, v. 23, n. 4, p. 3–13, 2000. Citation on page 21.

RAMADHAN, M. M.; SITANGGANG, I. S.; NASUTION, F. R.; GHIFARI, A. Parameter tuning in random forest based on grid search method for gender classification based on voice frequency. **DEStech transactions on computer science and engineering**, v. 10, n. 2017, 2017. Citation on page 25.

RAMEZAN, C. A.; WARNER, T. A.; MAXWELL, A. E. Evaluation of sampling and cross-validation tuning strategies for regional-scale machine learning classification. **Remote Sensing**, MDPI, v. 11, n. 2, p. 185, 2019. Citation on page 25.

RASCHKA, S. **Python machine learning**. [S.l.]: Packt publishing ltd, 2015. Citation on page 35.

RAWAT, S. S.; MISHRA, A. K. Review of methods for handling class-imbalanced in classification problems. **arXiv preprint arXiv:2211.05456**, 2022. Citation on page 31.

RIDZUAN, F.; ZAINON, W. M. N. W. A review on data cleansing methods for big data. **Procedia Computer Science**, Elsevier, v. 161, p. 731–738, 2019. Citation on page 21.

SAHOO, G.; KUMAR, Y. Analysis of parametric & non parametric classifiers for classification technique using weka. **International Journal of Information Technology and Computer Science (IJITCS)**, v. 4, n. 7, p. 43, 2012. Citation on page 26.

SALAM, M. A.; AZAR, A. T.; ELGENDY, M. S.; FOUAD, K. M. The effect of different dimensionality reduction techniques on machine learning overfitting problem. **Int. J. Adv. Comput. Sci. Appl**, v. 12, n. 4, p. 641–655, 2021. Citation on page 33.

SARKAR, D.; BALI, R.; SHARMA, T. The python machine learning ecosystem. In: ____. **Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems**. Berkeley, CA: Apress, 2018. p. 67–118. ISBN 978-1-4842-3207-1. Available: <https://doi.org/10.1007/978-1-4842-3207-1_2>. Citations on pages 35 and 36.

SAXENA, V.; AGGARWAL, A. Comparative study of select non parametric and ensemble machine learning classification techniques. In: IEEE. **2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)**. [S.l.], 2020. p. 110–115. Citation on page 26.

SHARMA, V. A study on data scaling methods for machine learning. **International Journal for Global Academic & Scientific Research**, v. 1, n. 1, p. 23–33, 2022. Citation on page 22.

SILVA, C. A. G. da; SÁ, J. L. R. de; MENEGATTI, R. Diagnostic of failure in transmission system of agriculture tractors using predictive maintenance based software. **AgriEngineering**, MDPI, v. 1, n. 1, p. 132–144, 2019. Citation on page 14.

STETCO, A.; DINMOHAMMADI, F.; ZHAO, X.; ROBU, V.; FLYNN, D.; BARNES, M.; KEANE, J.; NENADIC, G. Machine learning methods for wind turbine condition monitoring: A review. **Renewable Energy**, v. 133, p. 620–635, 2019. ISSN 0960-1481. Available: <https://www.sciencedirect.com/science/article/pii/S096014811831231X>. Citation on page 20.

STEVENS, S. S. On the theory of scales of measurement. **Science**, American Association for the Advancement of Science, v. 103, n. 2684, p. 677–680, 1946. Citation on page 39.

STONE, M. Cross-validatory choice and assessment of statistical predictions. **Journal of the royal statistical society: Series B (Methodological)**, Wiley Online Library, v. 36, n. 2, p. 111–133, 1974. Citations on pages 25 and 42.

SUKAMTO, S.; HADIYANTO, H.; KURNIANINGSIH, K. Knn optimization using grid search algorithm for preeclampsia imbalance class. In: EDP SCIENCES. **E3S Web of Conferences**. [S.l.], 2023. v. 448, p. 02057. Citations on pages 25 and 29.

TSAI, C.-W.; LAI, C.-F.; CHAO, H.-C.; VASILAKOS, A. V. Big data analytics: a survey. **Journal of Big data**, SpringerOpen, v. 2, n. 1, p. 1–32, 2015. Citation on page 18.

VANDERPLAS, J. **Python data science handbook: Essential tools for working with data**. [S.l.]: " O'Reilly Media, Inc.", 2016. Citation on page 36.

VARMA, S.; SIMON, R. Bias in error estimation when using cross-validation for model selection. **BMC bioinformatics**, BioMed Central, v. 7, n. 1, p. 1–8, 2006. Citation on page 25.

WAH, Y. B.; RAHMAN, H. A. A.; HE, H.; BULGIBA, A. Handling imbalanced dataset using svm and k-nn approach. In: AIP PUBLISHING. **AIP Conference Proceedings**. [S.l.], 2016. v. 1750, n. 1. Citation on page 23.

WAINER, J.; FONSECA, P. How to tune the rbf svm hyperparameters? an empirical evaluation of 18 search algorithms. **Artificial Intelligence Review**, Springer, v. 54, n. 6, p. 4771–4797, 2021. Citation on page 31.

XIE YI ZHAO, S. X. M. H. G.; ZHANG, Y. Multi-classification method for determining coastal water quality based on svm with grid search and knn. **International Journal of Performability Engineering**, Int J Performability Eng, v. 15, n. 10, p. 2618, 2019. Available: <https://www.ijpe-online.com/EN/abstract/article_4251.shtml>. Citation on page 31.

ZF. **ZF Company Profile**. 2024. Citations on pages 15 and 38.

ZHANG, S.; LI, X.; ZONG, M.; ZHU, X.; WANG, R. Efficient knn classification with different numbers of nearest neighbors. **IEEE Transactions on Neural Networks and Learning Systems**, v. 29, n. 5, p. 1774–1785, 2018. Citation on page 29.