Aalto University
School of Science

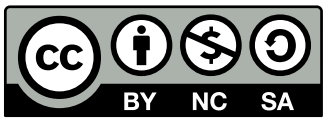
Master's programme in ICT Innovation - Data Science

Applying Machine Learning to Forecast Formula 1 Race Outcomes

Loreto García Tejada

Master's Thesis
2023

© 2023
This work is licensed under a Creative Commons "Attribution-NonCommercial-ShareAlike 4.0 International" license.



Aalto University
School of Science

Author Loreto García Tejada

Title Applying Machine Learning to Forecast Formula 1 Race Outcomes

Degree programme ICT Innovation - Data Science

Major Data Science

Supervisor Prof. Alex Jung

Advisor Prof. Alex Jung

Date 31 July 2023 **Number of pages** 69+1 **Language** English

Abstract

Pit stops are integral to the success of drivers in Formula 1 racing. This thesis aims to develop a predictive model that effectively determines the optimal timing for pit stops during specific laps of Formula 1 races. By employing machine learning algorithms and analyzing historical race data from the 2019 to 2022 seasons, this study creates a reliable system that considers various race factors, including tire degradation, car positions, and overall race dynamics. Three machine learning algorithms, namely Support Vector Machines (SVM), Random Forest, and Artificial Neural Networks are utilized and compared based on performance metrics, primarily the F1 score. The objective is to identify the most suitable algorithm capable of accurately predicting pit stop requirements. The findings of this thesis highlight the challenging nature of pit stop prediction in Formula 1. While the models demonstrate reasonable accuracy in predicting pit stops, achieving precise predictions remains complex due to the multitude of variables and inherent uncertainties involved. The results emphasize the models' potential as valuable decision-support tools rather than standalone predictors, emphasizing the importance of incorporating additional information and expert knowledge into the decision-making process.

Keywords Machine learning, Data science, Artificial neural networks, Random forest, Support vector machine, Pit stops, Formula 1, Predictive model

Preface

I would like to begin by expressing my deepest gratitude to my parents, who have been unwavering pillars of support throughout my life. Their unconditional love, guidance, and motivation have played a crucial role in shaping my educational and personal journey.

I am deeply grateful to my brother Diego for introducing me to the world of Formula 1. Through our shared enthusiasm for this sport, our bond has grown stronger, and it has increased the range of things that we enjoy together.

I extend my sincere appreciation to the remarkable people I have had the privilege of knowing during my two-year master's program. To those who have shared my experiences in Madrid and Helsinki, thank you for the invaluable friendships, bonds and memorable moments we have shared.

Special gratitude goes to Professor Alex Jung for his guidance and mentorship as my advisor and supervisor. I am immensely grateful for the opportunity he has given me to undertake this thesis and for placing his trust in my idea.

Loreto García Tejada

Contents

Abbreviations

AI	Artificial Intelligence
ANNs	Artificial neural network
AUC	Area under the curve
DL	Deep Learning
E_t	Training error
E_v	Validation error
F1	Formula 1
FIA	Fédération Internationale de l'Automobile
FN	False negative
FP	False positive
FPR	False positive rate
MSE	Mean squared error
ML	Machine learning
NASCAR	National Association for Stock Car Auto Racing
PCA	Principal Component Analysis
Poly	Polynomial
RF	Random Forest
ROC	Receiver Operating Characteristic
RBF	Radial basis function
SC	Safety car
SVM	Support Vector Machine
TN	True negative
TNR	True negative rate
TP	True positive
TPR	True positive rate
VSC	Virtual safety car

1 Introduction

In recent years, the field of machine learning has made significant strides in various domains, including sports analytics. With the advent of powerful computational techniques and the availability of vast amounts of data, machine learning algorithms have proven to be effective tools in predicting various outcomes and making informed decisions. In the world of motorsports, Formula 1 stands at the forefront of technological advancements, pushing the boundaries of engineering and performance.

Pit stops undoubtedly represent a critical and fundamental aspect of a Formula 1 race, as they play an irrefutable role in determining the outcome of the competition. In this intensely competitive motorsport, where fractions of a second separate victory from defeat, pit stops become crucial moments that can decide a driver's chances of success. With teams constantly seeking an advantage over their rivals, pit stops' efficiency and strategic timing become paramount. A well-executed pit stop can propel a driver to the front of the pack, giving him an advantageous position, fresh tires and the opportunity to unleash his full potential on the track. On the other hand, a poorly timed or suboptimal pit stop can result in the loss of valuable positions, forcing the driver to make up ground and potentially compromising his chances of victory. Consequently, accurately predicting the optimal time for a pit stop is a vital task in Formula 1, allowing teams to fine-tune their race strategies, gain a competitive advantage and potentially determine the outcome of a race.

1.1 Context

Formula 1, officially known as the FIA Formula 1 World Championship, is a globally renowned motorsport representing the pinnacle of automotive engineering, speed, and competition. Established in 1950, Formula 1 has evolved into the premier racing championship on wheels, captivating millions of motorsport enthusiasts worldwide. The championship features a series of high-speed races held at various circuits around the world, where drivers and teams compete for victory and the coveted World Championship titles, both for drivers and constructors.

The FIA (Fédération Internationale de l'Automobile) serves as the governing body of Formula 1, responsible for setting the regulations, safety standards, and sporting rules that govern each race. Teams, comprised of drivers, engineers, mechanics, and support staff, meticulously design and construct their race cars with the utmost precision to achieve optimal performance and reliability. The championship unfolds through a series of races known as Grand Prix, hosted on circuits of varying lengths, layouts,

and environmental conditions.

The timing of a pit stop is crucial due to several factors. First and foremost is tyre degradation. Formula 1 tyres suffer significant wear and lose grip over time, affecting the car's performance. As tyres degrade, lap times increase, so it is essential to change tyres at the optimum time to minimise time loss. If a driver stays on worn tyres for too long, his lap times will suffer and he risks being overtaken by competitors who have made timely pit stops.

In addition, the timing of pit stops is influenced by the positions of other cars on the track. Pit stops are not performed in isolation but in the context of a race. Teams must carefully assess the positions of their rivals and the potential traffic they may encounter during their laps out of the pit lane. If a driver enters the pits at the wrong time, he may rejoin the race in heavy traffic, losing valuable seconds and compromising his chances of gaining positions.

In addition, the state of the race itself is an important factor in the timing of the pit stop. Teams analyse various race factors, such as safety car periods, weather conditions and overall race strategies, to make informed decisions on when to pit. A well-timed pit stop during a safety car period, for example, can provide a substantial advantage, as the driver can make his pit stop without losing too much time relative to his competitors.

1.2 Problem Statement

The goal of this thesis is to create a predictive algorithm that can forecast if a driver should pit on a certain lap in a Formula 1 race. By analysing historical race data and exploiting machine learning algorithms, the aim is to create a reliable system that can effectively predict the need for a pit stop, taking into account various race factors such as tyre degradation, the positions of other cars on the track or the state of the race.

1.3 Methodology

To achieve the objective of predicting pit stop requirements, this study utilizes diverse information from all drivers in the races spanning from the 2019 to 2022 seasons. The analysis starts in 2019 to account for the regulatory changes implemented during that year [1], specifically related to the tire types used in Formula 1. By focusing on this period, the study ensures the inclusion of homogeneous data, allowing for a comprehensive analysis of pit stop dynamics.

Three machine learning algorithms are used: Support Vector Machines (SVM),

Random Forest and Neural Networks. These algorithms have been specifically selected for their effectiveness in classification tasks and their ability to capture complex patterns and relationships in data. By comparing and evaluating the performance of these algorithms, the thesis aims to identify the most suitable approach to accurately predict whether or not a pit stop should be performed.

Machine learning models are evaluated using multiple metrics to assess their performance. The primary measure is the F1 score, however, other metrics such as the confusion matrix and its associated metrics, the comparison between the training and validation error, the ROC curve and AUC, and the analysis of loss evolution are also utilized for evaluation. By comparing the performance of the three selected algorithms in the independent test set, the aim is to identify the most effective approach to pit stop prediction in Formula 1. The chosen algorithm will provide valuable information for optimising race strategies and making pit stop decisions, ultimately offering a potential competitive advantage in the world of Formula 1 and contributing to advances in sports analytics.

1.4 Limitations

Pit stop prediction in Formula 1 poses several inherent challenges that need to be taken into account. Firstly, the complexity of pit stop prediction is due to the multitude of variables involved, such as tyre degradation, fuel consumption, weather conditions, track position and the actions of other drivers. These variables interact in complex ways, making it difficult to capture all the nuances and accurately predict the optimal timing of a pit stop. Despite the use of machine learning algorithms, the intricate nature of pit stop dynamics presents a limitation to achieving perfect predictions.

In addition, Formula 1 teams have access to private data that is not publicly available during the race. These include telemetry data and race strategies, which provide teams with valuable information that can significantly influence pit stop decisions. The fact that this data is not publicly available limits the depth and breadth of information that can be incorporated into predictive models, which may affect the results obtained in this study.

In addition, changes in race regulations over time can influence pit stop dynamics in Formula 1. Regulatory changes may introduce new variables or alter existing ones. Consequently, predictive models developed from historical data may have limited applicability to future races with different regulations.

1.5 Industrial relevance

Formula 1 is a multi-billion dollar industry that moves staggering amounts of money. In 2022 there was a profit of US$2.6 billion, which also increased by 20% compared to the previous year [2]. This huge financial impact is reflected in the large budgets allocated by the various teams throughout the year. In an effort to promote financial sustainability and fairness, a budget cap was introduced in 2021. Teams are now limited in their annual spending, with the budget cap set at $135 million per season in 2023 [3]. This budget restriction aims to level the playing field and create a more competitive environment for all teams.

Within these very high budgets, a considerable part is dedicated to strategy development, including pit stop optimisation. Although the exact amounts may vary from team to team, it is estimated that Formula 1 teams spend a considerable part of their budget on race strategy [4]. This allocation covers several aspects, such as the development of predictive models, data analysis and the use of strategists to make informed decisions during races. By using accurate pit stop predictions, teams can optimise their strategy, to gain a competitive advantage and secure better race results.

The applications of the thesis can go beyond the sport itself. Media involvement plays a vital role in the global appeal and financial success of Formula 1. Broadcasting rights, advertising revenues and sponsorship deals contribute significantly to the sport's revenue streams. Accurate pit stop predictions can enhance the viewer experience by providing real-time information and analysis, captivate audiences and attract greater advertising investment.

In addition, the global popularity of Formula 1 has led to a flourishing betting industry, which offers a wide variety of markets and opportunities for fans around the world. Accurate pit stop predictions can enable bettors to make well-informed decisions, enhancing their betting experience with valuable insights. By leveraging predictive pit stop models, betting fans can better understand pit stop dynamics and factor them into their bets.

Overall, the industrial relevance of this thesis lies in its potential to contribute to the financial success of Formula 1, the engagement of media audiences and the enhancement of the betting experience. Through the development of accurate and reliable pit stop prediction models, this research may have significant implications for teams, broadcasters, advertisers and betting companies, ultimately contributing to the continued growth and success of the Formula 1 industry.

1.6 Outline

The thesis is structured as follows: Chapter 2 serves as a background, providing an overview of the technical foundations and contextual background of the thesis, with a specific emphasis on machine learning and its various methodologies. This chapter also reviews previous work within this field, providing a comprehensive understanding of the existing literature and highlighting the novelty and contribution of the present study. Chapter 3 delves into the detailed methodology, explaining the various steps involved in the development of the predictive model. It covers components such as data collection and pre-processing procedures, as well as the creation and configuration of multiple machine learning algorithms. Subsequently, Chapter 4 presents the results obtained from the application of the algorithms. This chapter also examines the influence of different factors and variables on the prediction results. Finally, in Chapter 5, the thesis concludes by summarising the main results and drawing significant conclusions, highlighting the contributions and limitations of the research. Furthermore, it offers recommendations for future studies and identifies possible areas for improvement, providing an overall understanding of the research process and its results.

2 Background

This section is an overview of the background and technical foundations of machine learning, focusing on its various methods. It explores different approaches in machine learning, including supervised, unsupervised, semi-supervised, and reinforcement learning, while also discussing evaluation methods for assessing model performance. Additionally, this section emphasizes three prominent machine learning methods: Random Forest, Support Vector Machines (SVM), and Artificial Neural Networks (ANN).

Furthermore, the thesis conducts a review of previous works in the field, aiming to provide context for the research and highlight the novelty of its contribution. Overall, this background section serves to provide a comprehensive understanding of machine learning, highlight specific methods, and contextualize the thesis within the existing body of knowledge, setting the stage for the subsequent chapters.

2.1 Machine learning

Machine learning is the development of algorithms and statistical models that can comprehend and learn from data without being programmed in advance. Machine learning algorithms are able to improve their performance on specific tasks over time, as they are exposed to more data and as they gain experience. Patterns and connections within the data can be identified by the models, which can then be used to make predictions or decisions on new data [5].

The field of machine learning comprises three essential elements: model, data, and loss. At its core, machine learning is based on the scientific concept of experimentation and testing. Models are continuously evaluated and refined based on the loss they incur in predicting outcomes of real-world phenomena. This iterative process of trial and error allows machine learning methods to improve the accuracy and effectiveness of their predictions over time [6].

2.1.1 Approaches

Generally, machine learning methods can be classified into various groups that correspond to different learning paradigms based on the feedback the learning system can employ. These categories depend on the type of feedback accessible to the learning system.

2.1.1.1 Supervised learning

Supervised learning involves an algorithm that learns to make predictions by analyzing input-output pairs that have already been labelled [7]. This means that the algorithm is presented with a set of training examples that comprise both input data and corresponding output labels, and it uses these examples to make predictions on unseen data by drawing from the patterns and relationships between the input data and the output labels that it has learned.

Two common types of supervised learning are classification and regression. In classification problems, the algorithm is trained to predict which category or class a new observation belongs to based on previously labelled samples. Categories can be binary, with only two classes, such as true/false or yes/no, or multi-class, with more than two classes [8]. In regression problems, the algorithm is trained to predict a continuous output variable, a numerical value, based on a set of input features [9]. The success of a supervised learning model is heavily dependent on the choice of the appropriate type of problem. It is crucial to thoroughly assess both the problem at hand and the type of data available, as making the wrong decision might lead to ineffective or incorrect models.

2.1.1.2 Unsupervised learning

In unsupervised learning, the algorithm discovers patterns and relationships in data without relying on labelled information. Unlike supervised learning, which involves providing labelled data, unsupervised learning focuses on uncovering inherent structures or groups within the data itself [10]. Unsupervised learning techniques are divided into two categories: clustering and dimensionality reduction.

The clustering method divides similar data points into clusters or subgroups depending on their characteristics or traits, so that observations within the same cluster are similar based on one or more predefined criteria, whereas observations drawn from different clusters are dissimilar [11]. Dimensionality reduction, on the other hand, is a technique that aims to decrease the number of characteristics or attributes in a dataset while preserving the maximum amount of relevant information. The idea is to transform the data from a high-dimensional space to a lower-dimensional space, making it easier to visualize and analyze [12].

2.1.1.3 Semi-supervised learning

Semi-supervised learning combines supervised and unsupervised learning features. The algorithm in this type of learning receives both labelled and unlabeled data and learns to generate predictions by utilizing the information contained in both types of input. The goal is to use the available labelled data to construct a model that can make accurate predictions about new unlabeled data, utilizing the unlabeled data to enrich the labelled data and improve model performance. It is founded on the premise that obtaining labelled data can be costly and time-consuming, and there is frequently only a limited amount available. Unlabeled data, on the other hand, is frequently abundant and easy to gather but lacks precise labelling [13].

There are various approaches to semi-supervised learning, including co-training and multi-view learning. In co-training, two or more models are trained on different data views and are used to label unlabeled data, which is then added to the training set [14]. On the other hand, in multi-view learning, the data is represented in several views and models are trained on them to take advantage of the information contained in each view [15].

2.1.1.4 Reinforcement learning

In reinforcement learning, an agent interacts with the environment in order to learn how to make the best decisions. This agent gathers information from the environment through rewards or punishments according to the behaviours it takes. The goal is to determine a strategy that maximizes the cumulative reward over time. This is accomplished by allowing the agent to learn through trial and error, experimenting with various actions and watching the ensuing reaction of the environment. The agent adjusts its policy in response to input, learning to do actions that maximize the expected future reward [16].

Reinforcement learning algorithms can be categorized into two main types: model-based and model-free. Model-based algorithms develop an environment model and use it to predict the results of certain actions. In contrast, model-free algorithms learn a mapping from states to actions without explicitly modelling the environment.

2.1.2 Evaluation and validation

Evaluating the performance of machine learning methods is an important step in developing and deploying effective machine learning models. The goal of this evaluation is to assess the performance and generalization capability of a machine

learning model when applied to novel and unseen data, which is critical for ensuring that the model performs well not only on training data but also in real-world circumstances. This prevents the model from overfitting the training data.

2.1.2.1 Confusion matrix

There are various metrics that can be utilized to evaluate the performance of machine learning models, depending on the problem being addressed. In classification problems, a very useful tool is the confusion matrix, which can be seen in Figure 1, from which different metrics can be extracted.

		Predicted condition	
		Positive	**Negative**
Actual condition	**Positive**	True positive (TP)	False negative (FN)
	Negative	False positive (FP)	True negative (TN)

Figure 1: Confusion matrix. Inspired by [17].

In binary classification, a confusion matrix is commonly used to assess the performance of a model. This matrix consists of four entries: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). True positives represent the number of cases where the model correctly predicted a positive class when the actual class was indeed positive. True negatives correspond to the number of cases where the model accurately predicted a negative class when the actual class was negative. On the other hand, false positives indicate the number of cases where the model incorrectly predicted a positive class, despite the actual class being negative. Similarly, false negatives represent the instances where the model predicted a negative class, but the actual class was positive. These four entries provide valuable information about the model's performance in differentiating between the positive and negative classes [18].

Different metrics can be extracted from the confusion matrix [19], where the best value is a 1 and the worst value is a 0. One of the most widely used is accuracy, which provides a simple measure of how well a model has performed. It calculates the proportion of correctly identified cases in the dataset, calculated by dividing the number of correctly classified classes by the total number of items. In other words, it indicates how often the model correctly predicts the class label. However, accuracy may not be a reliable metric for assessing the performance of a classification model when the dataset is unbalanced, with more elements belonging to one class than the

rest, in these cases, we may have a high accuracy value, but this is because the model is limited to predicting the majority class.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

To overcome these limitations, one can use alternative metrics like precision and recall. Precision measures the proportion of correctly predicted positive instances out of all the instances the model labeled as positive. In simpler terms, it indicates how accurate the model is in its positive predictions and its ability to minimize false positive errors.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

Recall, in contrast, measures the percentage of accurately predicted positive instances among all the true positive instances. To clarify, recall indicates how frequently the model accurately identifies positive instances out of the total actual positive instances or, in simpler terms, how effectively the model avoids making false negative errors.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

It is important to note, however, that precision and recall are frequently inversely connected. For example, a model that classifies all instances as positive will have a high recall but poor precision, whereas a model that only predicts occurrences with a high degree of confidence will have high accuracy but a low recall. The F1 score offers a balanced assessment by considering both precision and recall. It proves particularly valuable in scenarios where it is crucial to account for both precision and recall as essential metrics, especially in cases where the data is imbalanced.

$$\text{F1 Score} = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

2.1.2.2 Comparison of training error with validation error

The comparison of training error with validation error is a diagnostic approach in evaluating the generalization capabilities of machine learning (ML) methods. During the training process, the model aims to minimize the training error (E_t) by adjusting its parameters to fit the training data. The training error represents how well the model

performs on the data it was trained on, providing insights into its ability to learn from the training set.

On the other hand, the validation error (E_v) is computed by evaluating the model's performance on a separate dataset that it has not seen during training. This independent evaluation set allows us to assess how well the model generalizes to new and unseen data. A low validation error indicates that the model can make accurate predictions beyond the training data, demonstrating its ability to generalize.

A key aspect of this comparison lies in the establishment of a baseline performance denoted as $E^{(ref)}$. This baseline serves as a reference point for evaluating the model's performance. It can be derived from probabilistic models for the data points, where the minimum achievable risk corresponds to the expected loss of the Bayes estimator—a model that provides optimal predictions given the underlying data distribution. Alternatively, the baseline can be set based on existing ML methods that may not be directly suitable for the specific task at hand but offer valuable statistical properties. Furthermore, human performance can serve as a benchmark in tasks where human experts demonstrate expertise.

The evaluation of an ML method involves analyzing its performance by comparing the training error (E_t) with the validation error (E_v) and, when accessible, the benchmark ($E^{(ref)}$) [6].

- $E_t \approx E_v \approx E^{(ref)}$: When both the training error and validation error are similar and approximately equal to the baseline error, it indicates that the model performs well on the validation set and achieves results comparable to a benchmark. The small difference between training and validation errors suggests that overfitting is not a significant concern.
- $E_v > E^{(ref)}$: A significantly higher validation error compared to the training error may imply potential overfitting. In such cases, the model's performance on data points outside the training set is notably worse.
- $E^{(ref)} \approx E_v > E^{(ref)}$: In this scenario, both the training error and validation error are significantly higher than the baseline error, but they are approximately equal to each other. This suggests that the learned hypothesis does not overfit the training set. However, the training error achieved by the hypothesis is still notably larger than the benchmark error level.
- $E^{(ref)} > E_v$: A considerably greater training error compared to the validation error indicates potential issues with the i.i.d. (independent and identically

distributed) assumption for the data points. The i.i.d. assumption assumes that the data points are drawn from the same probability distribution.

2.1.2.3 ROC Curve

The ROC curve, which stands for receiver operating characteristic curve, is a visual representation that assesses the performance of a binary classifier. It accomplishes this by plotting the true positive rate (TPR) against the false positive rate (FPR) at various classification thresholds. This curve offers a graphical overview of the classifier's performance characteristics [18].

The TPR, also known as sensitivity or recall, is the percentage of positive cases that are accurately categorized as positive and is given by the formula.

$$\text{TPR (Sensitivity)} = \frac{TP}{TP + FN} \tag{5}$$

FPR, on the other hand, is the ratio of negative instances that are incorrectly classified as positive. It can be calculated using the following formula.

$$\text{FPR} = \frac{FP}{FP + TN} \tag{6}$$

By plotting TPR against FPR, the ROC curve shows the balance between the sensitivity of the classifier (TPR) and its specificity, also known as the true negative rate (TNR), which is equal to 1 minus the false positive rate.

$$\text{TNR (Specificity)} = 1 - FPR \tag{7}$$

The ROC curve offers a graphical depiction of the performance of a classifier at different classification thresholds. It provides a visual representation of how well the classifier performs across a range of threshold values. A steeper curve toward the upper left corner denotes better classifier performance with higher sensitivity and specificity, whereas a curve that nearly matches with the diagonal line denotes a classifier with random or bad performance. In Figure 2, the visual representation provides insight into determining the quality of a curve, allowing for the interpretation of its effectiveness in classification.

The area under the curve (AUC) is a widely employed metric obtained from the ROC curve. It quantifies the overall performance of a model by summarizing its capacity to

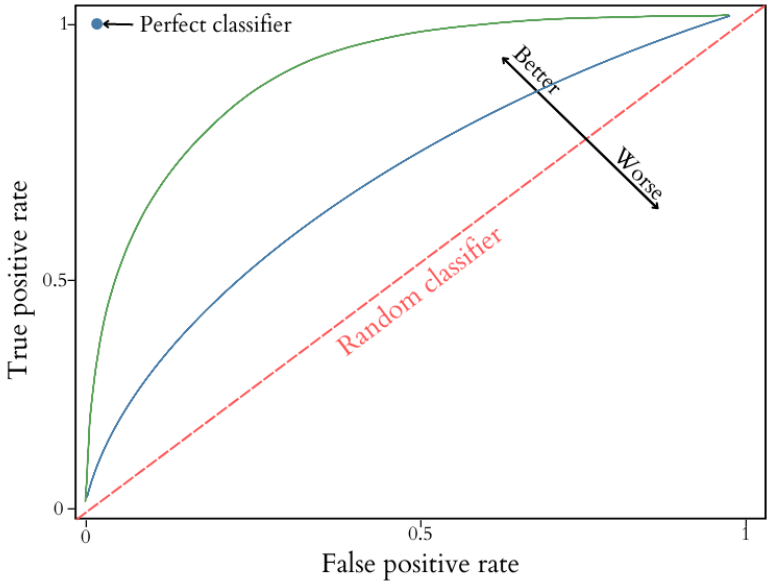


Figure 2: The ROC curve visualization for a "better" and "worse" curve. Inspired by [20].

distinguish between positive and negative instances across all available thresholds. A higher AUC value signifies superior discrimination and performance, with the range of values falling between 0 and 1.

Overall, the ROC curve is a powerful tool that complements the confusion matrix and traditional performance metrics and allows for a more detailed evaluation of binary classification algorithms.

2.1.2.4 Cross Validation

Cross-validation is a method utilized to evaluate a model's performance on new and unseen data. It involves dividing the available data into multiple folds or parts. Each fold is used as a test set, while the remaining data is utilized to train the model. This approach enables the assessment of the model's generalization ability to unfamiliar data and provides a reliable estimate of its performance [21].

Cross-validation encompasses several techniques, including k-fold cross-validation, leave-one-out cross-validation, and stratified cross-validation. In k-fold cross-validation, the dataset is divided into k equally-sized folds. The model is trained and evaluated k times, using each fold as the test set once, while the remaining k-1 folds are employed for training in each iteration. This systematic process ensures comprehensive testing of the model's performance on various subsets of the data,

leading to a more reliable and robust evaluation.

Cross-validation has the advantage of providing a more accurate assessment of a model's performance on new, unknown data because it assesses the model on different subsets of the data. Furthermore, because the model is evaluated on data that was not used for training, it helps to prevent overfitting [22].

2.2 Machine learning methods

This section will provide a detailed analysis of three popular machine learning algorithms: Random Forest, Support Vector Machine (SVM), and Artificial Neural Network (ANN). These methods have found applications in various fields and were selected based on their potential suitability for the research problem, as well as a review of related studies.

2.2.1 Random Forest

Random Forest is a popular machine learning technique that is widely utilized due to its ability to effectively handle high-dimensional data and capture complex nonlinear relationships. As a result, it has demonstrated its utility across various domains and applications. This method makes predictions using an ensemble of decision trees, which enhances accuracy and reduces overfitting [23].

The random forest algorithm initiates by randomly selecting a subset of the training data, with replacement, along with a subset of features. This selection process is iterated multiple times, resulting in the creation of a collection or "forest" of decision trees. Each decision tree within the forest is trained on a different subset of data and features. As a result, the ensemble of trees formed is diverse, allowing it to capture various aspects and patterns present in the data.

In each decision tree, the data is partitioned by randomly selecting a subset of features and finding the optimal split based on a criterion like information gain or Gini impurity. This process is repeated recursively until a stopping condition is satisfied, which may involve reaching a maximum tree depth or ensuring a minimum number of samples in each leaf node [24].

$$\text{Information gain} = IG(D, F) = H(D) - H(D|F) \tag{8}$$

$$\text{Gini impurity} = Gini(D) = 1 - \sum_{i=1}^{c} (p(i|D))^2 \tag{9}$$

where D is a set of training samples, F is a feature, $H(D)$ is the entropy of D, $H(D|F)$ is the conditional entropy of D given F, $p(i|D)$ is the proportion of samples in class i in set D, and c is the number of classes.

The random forest algorithm combines the predictions made by each individual tree in the forest to produce a final prediction for a new instance. In classification problems, the method takes the majority vote of all the trees' predictions, whereas, in regression-related problems, it takes the average of all the trees' forecasts.

$$\text{majority vote} = \arg\max_{k} \sum_{t=1}^{T} [t(y) = k] \tag{10}$$

$$\text{average} = \frac{1}{T} \sum_{t=1}^{T} t(y) \tag{11}$$

where y is the predicted class or value, k is a class label, T is the number of trees, and the sum is over all trees t. Furthermore, a visual diagram of how the random forest algorithm works can be seen in Figure 3.

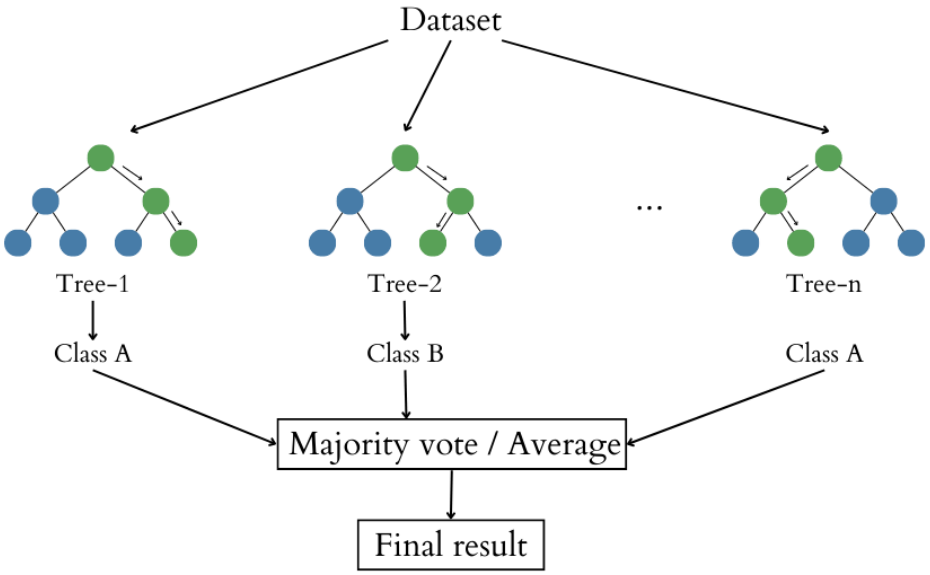


Figure 3: Visual representation of the random forest algorithm. Inspired by [25].

The effectiveness of Random Forest is influenced by various hyperparameters, including the number of trees, the number of features chosen for each split, and the depth of individual trees [26]. These hyperparameters can be tuned using cross-validation to optimize the performance of the algorithm on a given dataset.

In summary, Random Forest is an effective machine-learning method for problems with regression and classification. It integrates numerous decision trees to provide a more accurate and resilient model.

2.2.2 Support Vector Machine

SVM is a supervised machine learning technique that is used for classification and regression analysis. Vladimir Vapnik and his team pioneered it in 1995 [27]. SVMs are frequently employed in many domains, including biology, economics, engineering, and computer science.

The fundamental purpose of SVM is to determine the optimum hyperplane for separating data into distinct classes. The hyperplane is a line that divides data into two zones in a two-dimensional space. A hyperplane is a surface that divides data into multiple areas in a higher dimensional space. Figure 4 provides a visual depiction of the algorithm.

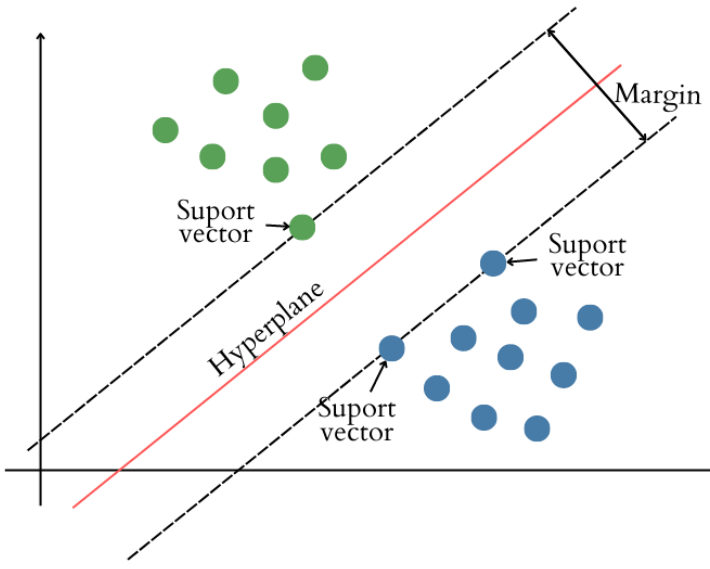


Figure 4: SVM visualization. Inspired by [28].

The SVM algorithm initiates by transforming the input data into a higher-dimensional feature space. Its primary goal is to find the hyperplane that maximizes the margin

between the two classes. The margin is the distance between the hyperplane and the nearest data points from each class. By maximizing this margin, the SVM algorithm aims to enhance its predictive accuracy and generalization capabilities, allowing it to proficiently classify new and unseen data [29].

To select the best hyperplane, SVMs employ a mathematical optimization approach known as quadratic programming. The optimization problem entails lowering a cost function that penalizes misclassifications while simultaneously maximizing the margin. A set of weights that define the hyperplane is the solution to the optimization problem. The SVM optimization problem can be phrased as follows:

$$\begin{aligned} &\text{minimize } \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\max(0, 1 - y_i(w^T x_i + b)) \\ &\text{subject to } y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, \ldots, n \end{aligned} \tag{12}$$

where C is a parameter that controls the balance between maximizing the margin and minimizing misclassifications, y_i is the label of the ith data point (either -1 or 1), ξ_i is a slack variable that allows for misclassifications, w is the weight vector, and b is the bias term.

The linear hyperplane equation can be written as, where x is the input data.:

$$w^T x + b = 0 \tag{13}$$

Kernel functions are utilized by SVMs to transform data into a higher-dimensional space, enabling the separation of non-linearly separable data. These functions facilitate the mapping of data onto a higher-dimensional space, where a linear hyperplane can effectively classify the data [30]. Kernel functions fall into different categories, including linear, polynomial, and radial basis functions (RBF). The kernel function is typically represented as:

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) \tag{14}$$

where ϕ is the feature map that maps the data into a higher dimensional space.

The distance between the hyperplane and a data point x can be calculated using the following equation:

$$\text{distance} = \frac{|w^T x + b|}{||w||} \tag{15}$$

where $||w||$ is the Euclidean norm of the weight vector.

Finally, the decision function for SVM can be written as:

$$f(x) = \text{sign}(w^T \phi(x) + b) \tag{16}$$

where the sign function, denoted as sign, assigns a value of -1 when the argument is negative, 0 when the argument is zero, and 1 when the argument is positive.

SVM performance is controlled by several critical hyperparameters. The kernel, which might be linear, polynomial, or radial, defines how the data is translated into a higher-dimensional space. The regularization parameter (C) determines the balance between maximizing the margin and minimizing training errors. A lower C value provides for a wider margin but may allow for more errors, whereas a higher C value imposes tougher margin limitations. Furthermore, kernel-specific hyperparameters such as degree for polynomial kernels or gamma for RBF kernels may be required [31].

To summarize, SVM is a powerful machine learning approach used for classification and regression tasks. It works by identifying a hyperplane that separates data into distinct classes. SVM can handle data that is not linearly separable by employing kernel functions to transform it into a higher-dimensional space. The key strengths of SVM include its effectiveness, accuracy, and interpretability, making it a popular choice across different domains.

2.2.3 Artificial Neural Networks

Neural networks are a type of machine learning that utilizes interconnected nodes or neurons arranged in layers, resembling the structure of the human brain. These networks consist of algorithms that aim to discover patterns and relationships within a dataset, imitating the functioning of the human brain [32]. There exist various types of neural networks, each with its own suitability for specific purposes and target applications. Some widely recognized models include feedforward neural networks [33], convolutional neural networks [34], and recurrent neural networks [35].

2.2.3.1 Architecture

Neural networks consist of interconnected nodes, or neurons, organized into layers, including the input layer, hidden layers, and output layer. This layered architecture enables the neural network to process data hierarchically, progressively extracting more abstract representations from the initial input data [36].

The input layer serves as the starting point for data input. Each neuron in the input layer represents a particular feature or attribute of the input data. Unlike the hidden layers, the input layer does not perform any computations. Its main function is to transmit the input features to the neurons in the subsequent hidden layers [37].

The hidden layers, located between the input and output layers, are essential for learning and extracting intricate patterns from the input data. These layers contain neurons that perform computations and transformations on the data received from the input layer. The number of hidden layers in a neural network can vary, depending on the complexity of the problem at hand [38].

The output layer of a neural network generates the final output or prediction. The number of neurons in the output layer is determined by the specific task at hand. For instance, in binary classification problems, there is typically a single neuron in the output layer that represents the probability or likelihood of belonging to a particular class. Each neuron in the output layer for multi-class classification represents the likelihood of falling into a particular class. Depending on the issue being solved, various activation functions can be used by the output layer [39].

2.2.3.2 Activation Functions

Since they introduce non-linearity into the network and aid in determining the output of neurons, activation functions are crucial in neural networks [40]. In neural networks, a variety of activation function types may be employed, some of which include:

- Sigmoid function: also known as the logistic function, is characterized by an S-shaped curve and maps the input to a value within the range of 0 and 1. This function is commonly employed in binary classification problems since it can compress the output into a probability value between 0 and 1, indicating the likelihood of belonging to a specific class. The sigmoid function is mathematically represented by the following formula:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{17}$$

- Tanh (Hyperbolic Tangent) function: shares similarities with the sigmoid function but maps input values to the range of (-1, 1). It is frequently utilized in the hidden layers of neural networks because it is centred around zero and can assist with gradient-based optimization. It is defined as:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{18}$$

- ReLU (Rectified Linear Unit) function: is a piecewise linear function that returns the input value if it is positive and zero otherwise. In other words, it "rectifies" negative values to zero while leaving positive values unchanged. It is computationally efficient and has become a popular choice for activation functions in deep neural networks due to its ability to mitigate the leakage gradient problem. It is defined as:

$$f(x) = \max(0, x) \tag{19}$$

- Leaky ReLU function: is a ReLU function variant that accepts negative input values with a small, non-zero gradient. This may lessen the likelihood of dead neurons, which develop when a cell continuously produces zero output. It is given by the formula:

$$f(x) = \max(\alpha x, x) \quad \text{where } \alpha \text{ is a small constant (e.g., 0.01)} \tag{20}$$

- Softmax function: is generally employed in multi-class classification issues' output layer. It transforms a vector of real values into a probability distribution, where each component represents the likelihood that a certain class will be part of it. Softmax is appropriate for multi-class classification applications since it makes sure that the projected probability adds up to 1.

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \quad \text{for each element } x_i \tag{21}$$

The activation functions employed in neural networks are only a few examples. Every activation function has advantages and disadvantages, and the selection is based on the particular issue that needs to be resolved.

2.2.3.3 Training and Learning Algorithms

To train neural networks, they are provided with a substantial amount of labeled data, enabling them to learn patterns and make predictions or classifications based on the input. The first step in this training process is to define the network's architecture, which involves determining the number of layers, the number of nodes in each layer, and the activation functions used. The architecture chosen is influenced by the specific problem at hand and the complexity of the data being processed.

After the architecture has been established, the network must be initialized. The connections between the nodes are given random weights in this process. The network can start unbiasedly by giving weights that are randomly assigned [41]. These initial weights allow different paths and combinations of features to be explored, which is necessary for the network to converge to optimal solutions during training.

The network is ready to be trained using a method known as forward propagation after initialization. Here, the network receives input data, and information travels across each layer, from input to output. Each node in the network computes an output by applying the activation function to a calculation using the weighted sum of the inputs. This process is repeated at each layer until the final output is obtained.

After the neural network generates an output, it is compared to the desired or expected output, and an error or loss function is calculated. This error function measures the discrepancy between the predicted output and the actual output. The specific error function chosen depends on the type of problem being addressed. In regression tasks, the mean squared error (MSE) is frequently employed. The MSE computes the average of the squared differences between the predicted output and the true output across all training examples [42].

$$E_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 \tag{22}$$

In classification tasks, where the output is a probability distribution over multiple classes, the cross-entropy loss is often employed.

$$E_{\text{CE}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \log(p_{ij}) \tag{23}$$

Once the error is calculated, the network adjusts its weights to minimize the error

using a process called backpropagation [43]. Backpropagation computes the gradient of the error function with respect to each weight in the network. This gradient specifies the direction and magnitude of the weight adjustment required to decrease the error. An optimization algorithm, such as gradient descent, is then used to iteratively update the weights and find the optimal values that minimize the error. The update rule for a weight using gradient descent is expressed by the following formula.

$$w = w - \eta \cdot \frac{\partial E}{\partial w} \tag{24}$$

Where w represents the weight being updated, η is the learning rate, which controls the step size of weight updates, and $\frac{\partial E}{\partial w}$ is the partial derivative of the error function with respect to the weight.

The process of forward propagation, error calculation, and backpropagation is repeated for each training example in the dataset. This is known as an epoch. Training typically involves multiple epochs to allow the network to learn from the data and improve its performance iteratively. The network gradually learns to adjust its weights in a way that minimises the overall error. This iterative process of updating weights based on the calculated gradients gradually improves the network's ability to make accurate predictions or classifications.

To ensure optimal performance on unseen data, it is essential to prevent overfitting during the training process. Overfitting occurs when the neural network becomes excessively tailored to the training data, leading to subpar performance when presented with new, unseen data. To mitigate overfitting, several techniques can be employed [44], some of which are regularization, dropout and early stopping.

Regularization is a technique that discourages the network from placing excessive importance on individual weights by introducing a penalty term to the error function. Dropout randomly deactivates selected neurons during training, preventing the network from relying too heavily on particular neurons and promoting robust representations. And early stopping consists of monitoring the performance of the network on a validation set, by stopping training early if the validation error starts to increase. The goal of adopting these strategies is to achieve a compromise between fitting the training data well and generalizing it to previously unseen data, eventually boosting network performance while avoiding overfitting.

The training procedure is repeated until a stopping requirement is reached, such as completing a particular number of epochs or performing satisfactorily on the validation

set. Once trained, the neural network can be used to make predictions or classifications on unseen data.

2.3 Previous works

Predicting outcomes in sports using machine learning has gained significant attention in recent years. Several previous works have focused on developing models and algorithms to forecast various aspects of sports events, such as game outcomes, player performance, and injury predictions [45, 46, 47].

There is a great variety of sports, with very different characteristics and rules, which gives rise to very different studies. Team sports are among the most popular, so we can find multiple jobs related to these types of sports such as football [48, 49, 50, 51], basketball [52, 53], American football [54, 55] or even volleyball [56]. We can find many different approaches, including SVM, Logistic Regression, Naive Bayes or Deep Neural Networks. In individual match sports, multiple works can be also found, especially in tennis [57, 58].

Due to the vast number of different sports and the increasing popularity of sports-related prediction, it is possible to find numerous papers attempting to forecast various elements within these sports. Therefore, the focus will be redirected towards the sports categories that encompass Formula 1. Formula 1 can be broadly classified as a racing sport, falling within a larger classification of sports that share the common objective of achieving a first-place finish. Despite apparent dissimilarities, several sports within the racing genre share this overarching goal. Illustrative examples of such sports include predictive analyses conducted for horse racing [59, 60], triathlon events [61], and cycling competitions [62].

More specifically, Formula 1 can be categorised as motorsport. Within this type of sport, we can also find multiple prediction works. There are different works which use different techniques to predict different aspects of the world of motorsport.

One form of predictive analysis in motorsport involves forecasting the race outcome and the final positions of drivers using either machine learning algorithms [63] or neural networks as the primary focus [64]. Another approach is found in [65] where artificial neural networks and Monte Carlo tree search are used for Formula E energy management.

There are also some works related to pit stop prediction in the motorsport world. National Association for Stock Car Auto Racing (NASCAR) is a motorsport sport

which has its biggest audience in the United States, both NASCAR and Formula 1 are popular motorsport series, but NASCAR primarily features oval track racing with stock cars, while Formula 1 focuses on road courses with open-wheel, purpose-built race cars, [66, 67] different machine learning algorithms are used for pit stop prediction in this sport. More specifically for formula 1, [68] integrates machine learning, optimisation and agent-based simulation together with a digital twin and [69] uses two neural networks. The first network is responsible for determining whether a driver should make a pit stop, while the second network decides the appropriate choice of tire compound if a pit stop is deemed necessary.

3 Methods

This section serves as a core component of the thesis, providing a comprehensive elucidation of the steps undertaken in its development. The primary objective is to outline the detailed process employed to achieve the research goals. It begins with the collection of relevant data, followed by rigorous pre-processing and cleaning procedures. These initial steps are crucial for ensuring data integrity and eliminating inconsistencies or biases that could hinder accurate analysis.

Subsequently, the section explores the creation and implementation of three specific machine learning algorithms: Support Vector Machines, Random Forest, and Artificial Neural Networks. To optimize the performance of these algorithms, extensive experimentation and configuration have been conducted. Hyperparameter tuning, a critical aspect of algorithm configuration, has been approached using cross-validation techniques. This allows for the identification of the most effective hyperparameter settings for the target variables, enhancing the accuracy and reliability of the models.

Importantly, it should be noted that while this section focuses on explaining the methods employed, the presentation and analysis of the results obtained from these methods will be covered in the subsequent section of the thesis. This separation allows for a clear distinction between the methodology and the outcomes of the research. Overall, this section establishes the foundation for the subsequent chapters, providing a comprehensive account of the methodology employed in the thesis. It offers transparency and clarity regarding the data collection, pre-processing, and the specific implementation of the algorithms.

3.1 Data Preparation

This subsection focuses on the crucial process of data preparation for machine learning models. It involves comprehensive data handling tasks, including feature selection, engineering, handling missing values, addressing outliers, and encoding categorical variables. The objective is to optimize the data for seamless integration into future machine learning models.

The data preparation process ensures that both input and output variables, which are essential components in machine learning, are properly formatted and transformed. Input variables, also known as features or attributes, represent the relevant information or characteristics of the data that are provided as input to a machine learning model. These variables capture patterns, relationships or properties that can influence the

model's predictions or results. Output variables, on the other hand, represent the desired goal or outcome that the machine learning model intends to predict or classify based on the input variables. By performing these data preparation steps, the data becomes more suitable for further development of the machine learning model, which improves the performance and accuracy of the model.

3.1.1 Data exploration

Several types of data are required for the problem address to be solved. A database was developed using data from several sources. Data from two different sources have been combined, the Ergast Developer API [70] and FastF1 [71].

Both sources are used because the first source does not contain all of the necessary information, despite being a very extensive source with information on seasons, tracks, and races dating back to 1950. The second feed from the first also collects data straight from The official F1 data stream [72], which provides additional information such as the types of tires used or whether there is any VSC in each lap. So although some data is found in both sources, other data is only found in one of them. There is data that has been extracted directly, and additional techniques have been built to extract novel data, such as calculating the time disparities between the car ahead and behind it, or identifying if a pit stop has already been done by these cars.

3.1.1.1 Input variables

Several variables have been acquired from the two sources of data, either directly or by doing calculations using the existing information, which will be utilized as input to predict. Numerous elements influence the decision of when to make a pit stop; considering these aspects, the following variables have been collected. These variables have been collected for each of the laps of all the races of different seasons. For the explanation of the variables, I will group them taking into account the point of evaluation of the variables.

The following variables have the same value for the totality of the laps of all drivers in the same race throughout the same season:

- *Year*: A categorical variable, which refers to the year of the season. It has been decided to collect data from the 2019 racing season up until the most recent complete season, which is 2022. Therefore, this variable will have 4 different possible values, referring to each of the seasons.
- *RaceNumber*: The race number within a season. In other words, the first race

of the season is number 1, the second race is number 2 and so on. As the season progresses, the teams learn more about the performance of the car and the competitors.

- *RaceName*: The name of the circuit. Not all seasons have the same circuits. And even if it is the same track in two different seasons, there may be minor changes between each of the tracks, but this is something that is not being taken into account here. It differs from the previous variable in that not all seasons have the same circuits, and they do not occur in the same order within the season.

The variables listed below are specific to each driver and remain constant throughout a race, for all laps. However, the values will differ between drivers. While the previous variables were indicative of the race, these variables are indicative of individual drivers. The information is known before the race starts, and it remains constant throughout the race.

- *Driver*: To identify which driver is being referred to, a 3-letter code associated with the driver is used. This code is usually derived from the beginning of the driver's last name. Another option is to use the driver's number, but over the years, numbers have been repeated among drivers, and a driver can change their number. For example, the winner of the previous championship may choose to wear number 1. This variable thus also allowed access to both data sources, so it was used to combine the data, although it is initially extracted from FastF1.

- *Team*: Name of the team in which the driver is in that race. It is normal for the driver to stay with the same team throughout the same season, although it is common to change from one year to the next, but there have also been team changes in the middle of the season. The performance of a driver might vary depending on the team he is in because the outcome of a race is not just dependent on the driver, but also heavily on the team, particularly when choosing a strategy and when developing the car. Hence, a driver's performance can vary depending on the team he is in.

- *QualyPositon*: This is the position in which the driver has qualified to start the race. It is indicative of the driver's performance of the particular circuit. The better his position in the standings, the better the car can be assumed to perform. This isn't always the case, though, as other events may have taken place that impacted the qualifying outcome, and a car's performance during a qualification lap differs from its performance throughout a race due to strategy and tire wear. It does, however, offer a quite accurate broad approximation. Only the result of

the qualifying session is taken into account, not the result of the sprint races that were introduced at some circuits in addition to qualifying in 2021 [73], these sprint races are infrequent, while the qualifying session remains consistent and accurately reflects performance.

- *GridPosition*: This is the position in which the driver started the race. This value is not always equal to how they finish in the classification, since due to different reasons you may get a penalty that results in a grid place penalty [74]. Even if a driver qualified first, he may start the race in last place if he makes more car modifications than are permitted during the season, for instance.
- *DriverStandings*: This variable contains the driver's position in the driver's championship. Knowing a driver's position in the championship is interesting because it provides insight into how well he performed throughout the season. If a driver is in a good position in the championship, he is more likely to have a successful race and vice versa. This information may be less helpful early in the season; for instance, there is no such information in the first race, and all drivers will have a value of 0.

To conclude the explanation of the input variables, the following variables pertain to each individual lap. They allow us to understand the driver's particular situation during each lap of a race.

- *LapNumber*: The lap indicator serves the purpose of identifying the driver's lap number and is an essential variable required for calculating and accessing other variables.
- *Position*: This variable refers to the driver's starting position at the beginning of the lap. It may happen that along the same lap, there is a change in the positions, but since the data is being collected by counting a lap in its entirety, only the position at the beginning is collected.
- *Compound*: This is a categorical variable indicating the type of tire being used. Nowadays there are three distinct types of compounds available for use in races under dry conditions; soft, medium and hard. The teams decide which compound to use in accordance with the race strategy they intend to adopt. The soft compound tire offers more speed but degrades more quickly, necessitating an earlier replacement. The hard compound tire, on the other hand, is slower but lasts longer. As its name implies, the medium compound tire is in the middle of the spectrum between soft and hard compounds. In rainy conditions, there are two types of tyres, intermediate and wet, whose use differs depending on the

intensity of the rain.

For the 2019 Formula 1 season, a new tyre nomenclature was introduced, eliminating the compound names used in previous years [75]. Instead of terms such as superhard, hard, medium, soft, supersoft, ultrasoft and hypersoft, a classification based on five main compounds was introduced: C1, C2, C3, C4 and C5. These compounds vary in terms of hardness and performance. During each Grand Prix, Pirelli, the tyre supplier, selects three of these compounds to be assigned as "hard compound", "medium compound" and "soft compound" for that race weekend, as mentioned above. These options might differ depending on the track's characteristics and forecasted conditions, allowing the tyre choices to be adjusted to each unique circuit. Because of these nomenclature and compound selection modifications, there is no direct equivalence between 2019 tyre types and previous years.

- *TyresChange*: Drivers are required to use at least two different types of dry tyres, according to the guideline [74]. If the driver has already switched tire types during the current lap, this variable is calculated based on the information from the previous variable. It is a binary variable, in which it can be either yes or no, this is represented as 1 or 0 respectively.

- *NumberPitStops*: This is the number of stops the driver has made up to this point, at the beginning of the lap. The number of times drivers stop during a race varies depending on the strategy they choose to follow. However, typically, the difference in the number of stops is only one. Some circuits require more stops on average due to high tire degradation, while others only require one stop.

- *TyreLife*: This variable refers to the number of laps the current set of tyres has been used. This parameter does not solely indicate the number of laps completed during the race, as the tyres may have also been used during practice or qualifying sessions.

- *TrackStatus*: It contains information about whether the track is clear or whether there is a yellow or red flag, as well as information about whether there is a safety car or virtual safety car. There are 7 different values, which can occur at the same time. There are seven distinct values, and they can occur simultaneously. For instance, number 2 represents a 'yellow flag' while number 4 represents a 'Safety Car', but we can also have 24 to indicate that both of these events are happening simultaneously during the current lap. During SC and VSC phases cars have to slow down and are prohibited from overtaking, which results in the

time lost during a pit stop being reduced, it is therefore likely that teams will use these periods for pit stops [76]. As in a race under normal circumstances, whether to pit during this time relies on each driver's strategy and that of the other drivers. It is customary to act in the opposite direction of the direct rival.

- *TimeLap*: This particular variable, which has a datetime type, stores the lap time for the driver's most recently completed lap. That is, it is the lap time from the previous lap to the current lap. Furthermore, other variables are derived from this variable.
- *TimeDiffAhead*: This is the distance in time separating the driver from the driver one position ahead, also known as the interval. Since he is the pilot's direct competitor, the distance may affect the strategy to be followed. Although it is not being considered in this work, it might be something to take into account in subsequent work, but you can activate the DRS, or Drag Reduction System, which assists in overtaking when you are less than a second behind the car directly ahead [77].
- *TimeDiffBehind*: In this scenario, the distance being referred to is the distance between the driver behind, in other words, the pilot to be defended against. In contrast to the preceding variable, when the driver in the first place is the one with no value, in this case, it is the driver in the last place.
- *TimeDiffToLeader*: also known as the gap, refers to the distance in time to the first position. It's useful to be aware of the time gap between your car and the leading car in the current race, as it provides insight into the pace at which you're competing compared to the current leader.
- *DriverAheadPit*: This is a binary variable, which records whether the car in front has already made a pit stop or not. In particular, if a pit has been made in the stint in which the driver is in, a stint refers to the period during a race when a driver is on a particular set of tires without pitting for a change. With this you can see the strategy being followed by the competitor, and if it is your direct competitor. Sometimes, the driver in front of you may appear to have a significant lead, but in reality, they may not have pitted yet, while you have already completed a pit stop.
- *DriverBehindPit*: The same thing occurs as it did in the preceding variable, but this time, information about the driver behind is recorded.

After researching factors that can affect decision-making in a race, we have identified

the variables listed above. These variables were selected based on theoretical considerations, analysis of past races, and evaluation within the models to ensure their ability to provide informative insights. To facilitate a more global view of all the variables that will be used, they are listed in Table 1. The name of the variable, its type, its source of information and its evaluation point is collected. In total, there are 21 variables, of which there are categorical, numerical, binary and time variables. In addition, in Appendix A it is possible to visualise the distribution of the variables, separated by type of variable.

Table 1: Input variables set.

Variable	Type	Source	Evaluation Point
Year	Categorical	FastF1	Constant race and driver
RaceNumber	Numerical	FastF1	Constant race and driver
RaceName	Categorical	FastF1	Constant race and driver
Driver	Categorical	FastF1	Constant driver
Team	Categorical	FastF1	Constant driver
QualyPositon	Numerical	Ergast	Constant driver
GridPosition	Numerical	FastF1	Constant driver
DriverStandings	Numerical	Ergast	Constant driver
LapNumber	Numerical	FastF1	Beginning lap
Position	Numerical	Ergast	Beginning lap
Compound	Categorical	FastF1	Beginning lap
TyresChange	Binary	Calculated	Beginning lap
NumberPitStops	Numerical	FastF1	Beginning lap
TyreLife	Numerical	FastF1	Beginning lap
TrackStatus	Categorical	FastF1	Beginning lap
TimeLap	DateTime	Ergast	Previous lap
TimeDiffAhead	DateTime	Calculated	Beginning lap
TimeDiffBehind	DateTime	Calculated	Beginning lap
TimeDiffToLeader	DateTime	Calculated	Beginning lap
DriverAheadPit	Binary	Calculated	Beginning lap
DriverBehindPit	Binary	Calculated	Beginning lap

3.1.1.2 Output Variable

In order to determine whether a lap is an appropriate time for a pit stop, it is important to be able to verify that the model's decision was accurate. The decision of whether to pit stop or not on the lap is a binary classification problem.

It is known in a race whether a driver has stopped or not on a lap. Therefore, the model's output will be evaluated based on its prediction of whether a pit stop should

be made on a specific lap in a race, and compared to what happened on that lap in the race.

However, the choices made by race teams aren't always correct, so it would be testing whether it is a good time to pit stop with values that do not reflect whether it is a good time. Due to the complexity of the problem, there is no perfect time to make a pit stop, so there is no way to check if the decision made by the models is correct in its entirety.

Taking all this into account, the variable to be used as the target variable is not only whether the driver has made a pit stop during the race on that lap, but also the calculated variable indicating whether that pit stop was a good pit stop. For each element that is being evaluated, i.e. for every lap of all the drivers on a circuit, we have the information on whether a pit stop was made and whether it was a good one. If the position has improved or stayed the same after the stop, either in that stint or if it was the last one until the end of the race, then it indicates that the stop was successful. Therefore, the different models will be evaluated using both variables.

Most Formula One races last 50-70 laps to cover the required 305 kilometres [78]. The number of pit stops in these laps is usually between one and three, depending on the circuit and the teams' strategy. This means that on most laps a driver does not perform a pit stop. Specifically, of all the laps for which information is available, a pit stop is performed in only about 3% of them [79]. This difference in the distribution of the data can be seen graphically, both for the actual pit stops and those considered as good pit stops, in Figure 5. Additionally, of all the pit stops that are performed, only about half of them meet the criteria established as good pit stops, so the difference between no pit stop and good pit stop distribution is even greater.

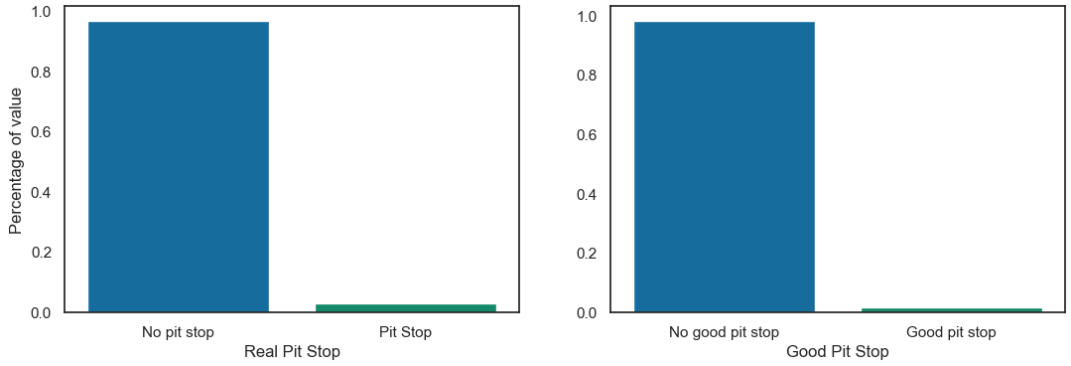


Figure 5: Percentage of laps without actual pit stops versus those that did, and for those considered a good pit stop.

3.1.2 Data Cleaning

Preprocessing has been done on the acquired data to either remove values that are not pertinent to this scenario and determine what to do with null values.

The data extracted from the two sources include information on all races and drivers who participated in the 2019 through 2022 seasons. It comprises 82523 laps and covers a total of 77 races held during the selected period.

To begin with, any races that were held under wet conditions have been removed. The rationale behind this decision is that pit stops under wet conditions serve a different reason compared to other races. Specifically, the requirement to switch to a different type of tire at least once is eliminated and the decision to change tires is primarily influenced by the state of the track and the driver's sensory perceptions, variables for which we lack relevant data [80].

Since the data for each type of tire on each lap is known, races in which a driver used the "Wet" or "Intermediate" type on a lap have been eliminated, as these are the types of compounds used when it rains, with the difference depending on the intensity of the rain. Figure 6 shows the distribution of laps with each type of compound used over the years; as can be seen, the types for rain are much less commonly used. Specifically, of the 77 races for which information has been collected, 12 of them were run in rainy conditions.

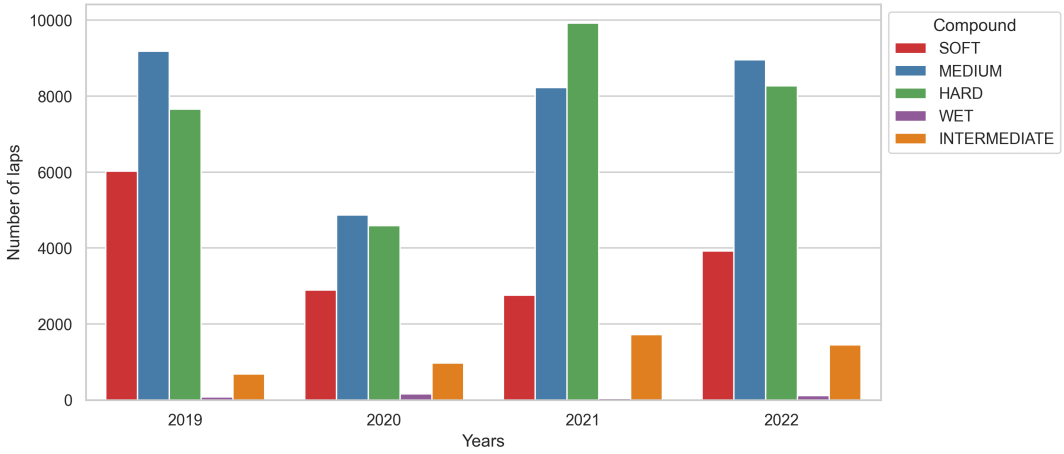


Figure 6: Distribution of laps with each of the tire compounds over the years.

There is an available variable in one of the data sources that contains the status of the driver at the end of the race. The data in question include the term "Finished" if the driver completed the race, but it's possible that the driver finished the race with additional laps completed due to being lapped. In this scenario, the value would be

denoted as "+1 Lap", "+2 Lap", and so on, depending on how many laps they were behind. Additionally, if the driver did not finish the race for various reasons, those reasons would also be listed as values in this variable. Therefore, pilots who have not completed the race due to different problems have been eliminated. Pilots who have overlapped more than 3 times have also been eliminated because their performance in the race was poor or they had some problem. This exclusion is likely implemented to ensure that only valid and complete race data is taken into account for analysis and evaluation purposes.

The information of drivers who have made more than four pit stops in a race has also been removed; if they have had to make that many stops, it is because a problem occurred during the race. Figure 7 shows the number of times a driver's information has been deleted in a race for each of the possible reasons. For visibility of the graph, all those elements that have only occurred once have been grouped under the variable "Others", within this variable we find factors such as "Radiator", "Transmission" or "Differential'.

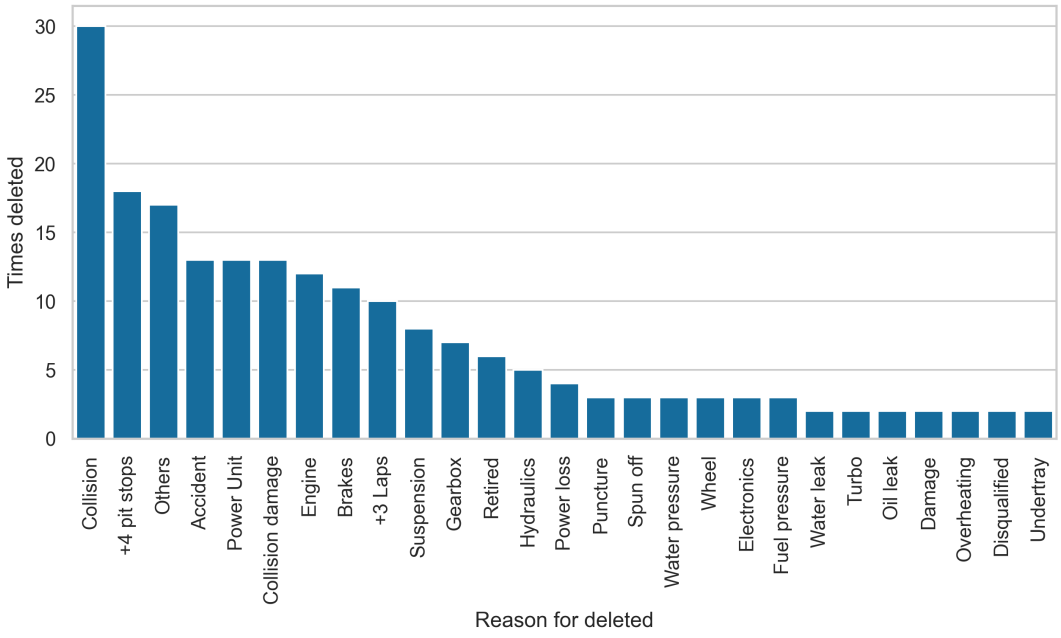


Figure 7: Number of times a driver's information has been deleted and the reason.

The following step is to examine the potential null values within the variables and determine the appropriate course of action. Only 2 variables have null values, these are *QualyPosition* and *DriverStandings*.

There is no information on the quality results of three pilots competing in three different races. Analyzing which pilots and races are involved, the reason for the

lack of information is that these pilots do not participate in the respective qualifying session. These missing values are assigned to their starting position, as in these cases, the drivers start in the last position, which would be their qualifying position if they had participated in the qualifying session.

Although the *GridPosition* variable does not have null values as such, some drivers have this value as 0, which does not make sense in the context of the problem. Since this variable refers to the driver's starting position, 0 values are substituted for the driver's position on the first lap of that driver in the race where the information is missing.

When one of the main drivers is unable to compete in a race, a substitute driver takes his place, although it is not common for the pilot not to be able to drive. In these cases, the variable *DriverStandings* is set to null. Since these drivers have no points in the championship, this null value is replaced by a value of zero.

Figure 8 demonstrates that the variables requiring the aforementioned adjustments have a minimal percentage of null values. The highest proportion of null values is found in the GridPosition variable, accounting for only 2 per cent of the data. Due to the small proportion of values and the feasibility of identifying the actual value, it was possible to replace these values instead of discarding them.

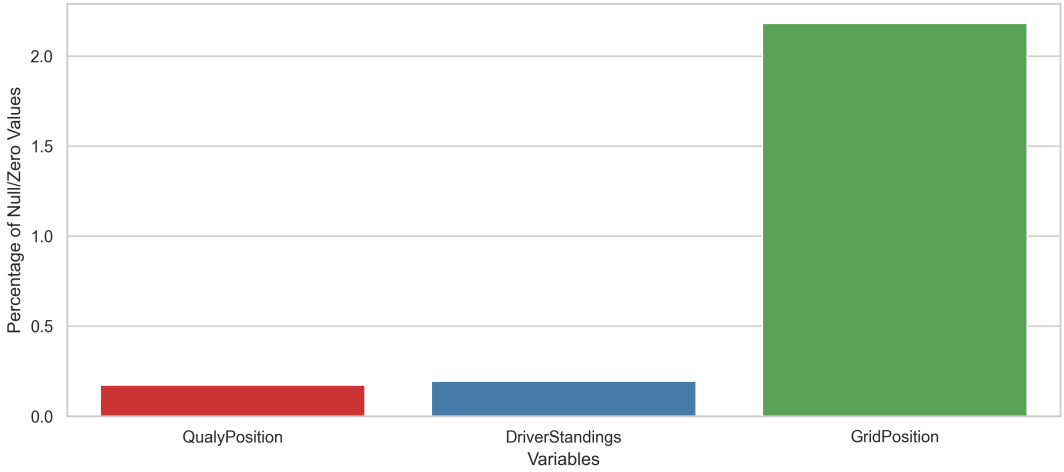


Figure 8: Distribution of null values within variables

After going through all the necessary data preprocessing steps that were previously outlined, the resulting number of laps that can be utilized in the subsequent analysis is 62566. This is the dataset that will be used in the following steps, which will first be transformed for further use in the machine learning algorithms.

3.1.3 Data Preprocessing

To construct predictive models using the data, it is crucial to preprocess the data first. This involves manipulating the raw data, which comprises categorical, binary, numerical and datetime variables, to make it more appropriate for machine learning algorithms. The list of the type of each of the variables can be seen in Table 1.

To employ categorical variables in machine learning algorithms, it is essential to perform preprocessing. Categorical variables have a limited set of discrete values, and to prevent any information loss, we need to convert them into a numerical format. One of the widely used techniques is one-hot encoding, which is employed here. It involves transforming each category into a binary column, where the presence of the category is represented by 1, and its absence is represented by 0 [81].

Since the binary variables only take two values, in this case, 0 or 1, it is unnecessary to make any changes to them, they can be used directly as input for the automatic learning algorithms.

Normalizing numerical variables is a crucial step in ensuring that all features are scaled similarly and have equal significance during analysis. The scaler that has been used for this purpose is the standard scaler [82]. This scaler performs the normalization by subtracting the mean and dividing it by the standard deviation of the input variable. This process ensures that the transformed variable has a mean of 0 and a standard deviation of 1. The standard scaler offers compatibility, distribution preservation, outlier robustness, and improved interpretability. After evaluating various combinations of algorithms and scaling techniques, including min-max scaler and robust scaler, it was determined that the standard scaler produced the most favourable results for the data. Considering its numerous advantages, such as compatibility, preservation of the distribution, robustness against outliers, and improved interpretability, the decision was made to adopt the standard scaler for the analysis.

The final category of variables that we encounter in our dataset is date-type variables. To preprocess these variables, we first convert them into seconds, thereby converting them into numerical variables. Then, we apply the same scaling technique used for the other numerical variables discussed earlier. This ensures that the date-type variables are normalized and have a similar scale to the other numerical variables.

3.1.3.1 Dimensionality Reduction

Upon completion of the preprocessing stage, the dataset comprises 102 variables. It has been decided to apply Principal Component Analysis (PCA) and perform the

algorithms with the original dataset and the one that has been applied PCA to compare the results. PCA is a popular method for reducing the dimensionality of data. It seeks to retain the most important information while reducing the number of variables. By transforming the original variables into a smaller set of uncorrelated components called principal components, PCA enables the identification of underlying patterns and structures within the data.

In this scenario, a 95% threshold was employed to determine the number of principal components to be retained. By applying this threshold, it was ensured that the chosen principal components collectively accounted for 95% of the total variance found in the original dataset. By keeping only these components, the dimensionality of the data was effectively reduced while still preserving a substantial amount of information. As a result, the dataset now comprises 48 variables, specifically the principal components that were retained.

3.2 Machine learning methods

Upon completion of the data preparation phase, the subsequent step involves the application of machine learning algorithms. To tackle this matter, three distinct algorithms, Random Forest, Support Vector Machines, and Artificial Neural Networks, were employed. Each algorithm underwent experimentation with different configurations and hyperparameters. This section presents a detailed account of the specific configurations utilized for these algorithms.

Before starting with the construction of machine learning algorithms, a crucial step is to split the data into distinct training and test sets. This split has two main objectives: to evaluate the algorithm's performance on previously unseen data and to assess its generalisability. In the context of this study, a cautious approach was adopted, employing a 70-30 split between training and testing. Therefore, 70% of the data was used to train the models, allowing them to learn patterns and relationships. While the remaining 30% of the data was reserved for testing and validation, which provided a robust assessment of the effectiveness of the algorithms in instances that have not been seen before by the algorithm.

The approach described below will be used to analyze data in two scenarios: one without applying PCA, and the other with PCA applied. The objective is to determine if there are any improvements or differences in the results between the two scenarios. Additionally, since there are two target variables related to pit stops (whether a pit stop was performed and if it was a good pit stop), the analysis will be conducted for

both variables. This will help identify the best outcomes and facilitate a comparison between them.

3.2.1 Random Forest

The first machine learning algorithm employed for data processing is Random Forest. As previously mentioned, this algorithm utilizes multiple decision trees to generate predictions by combining their outcomes. The implementation of Random Forest incorporates hyperparameter fitting, a process that seeks to identify the best configuration of hyperparameters for the model. Hyperparameters refer to customizable settings defined by the user that influence the algorithm's behaviour and effectiveness.

To adjust the hyperparameters, a grid search method is used. The grid search consists of systematically evaluating the model performance on a predefined grid of hyperparameter values. The choice of hyperparameters for grid search and optimization depends on several factors, including their relevance to the algorithm and their potential impact on the model's performance. In the case of Random Forest, the selected hyperparameters were chosen based on their significance and known effects within the algorithm. The hyperparameters that have been chosen and the different values are as follows [83]:

- *n_estimators*: represents the number of decision trees included in the random forest ensemble. The values tested are 100, 200, and 300. This selection was made to investigate the impact of increasing the number of trees on the accuracy and generalization capability of the model.
- *max_depth*: this hyperparameter determines the maximum depth allowed for each decision tree in the random forest. The options considered were no maximum depth (None), as well as depths of 5 and 10. The trade-off between model complexity and generalization performance was examined using these values.
- *min_samples_split*: specifies the minimum number of samples required to split an internal node while building the decision tree. In this study, the values tested for this hyperparameter were 2, 5, and 10. These values were selected to explore the impact of different splitting criteria on the model's ability to generalize to unseen data.

In addition, in response to the imbalance in the data, specifically within the classes of the target variable, the Random Forest algorithm configuration incorporated the parameter *class_weight* set to "balanced". The inclusion of this parameter is intended

to address the problem of unequal representation of classes in the dataset. When there is an imbalance in the data, assigning equal weights to each class during model training can result in biased predictions that favour the majority class. By specifying this configuration, the algorithm automatically adjusts the weights proportionally to the inverse of the class frequencies. This adjustment ensures that the model assigns more weight to the minority class, resulting in a more equal and unbiased representation of all classes during the training process.

The optimal hyperparameter configuration that maximises the F1 score was determined by grid search and cross-validation. Specifically, the analysis employed a five-fold cross-validation method. This involved dividing the dataset into five subsets or folds of equal size. The Random Forest model was trained on four of these folds and evaluated with the remaining fold, iteratively repeating the process for each of the folds. By exploring the hyperparameter combinations using grid search and evaluating the F1 score on all cross-validated folds, the optimal hyperparameter configuration for the data was identified.

After conducting the grid search, the table showcasing the best hyperparameter configurations for the output variables "has pit stop" and "good pit stop" can be found in Table 2. These hyperparameter configurations represent the optimal settings identified through the grid search process for achieving the best predictive results for both output variables.

Table 2: Best Hyperparameter Configuration for the RF Classifier.

Hyperparameters	Has pit stop	Good pit stop
max_depth	None	10
min_samples_split	10	2
n_estimators	100	300

The differences in hyperparameter configurations between the two output variables "has pit stop" and "good pit stop" highlight the unique characteristics and objectives associated with each prediction task, enabling the model to optimize its performance based on the specific requirements for each variable.

3.2.2 Support Vector Machine

The following machine learning algorithm applied is Support Vector Machines (SVM). Similar to Random Forest, SVM also involves adjusting hyperparameters to optimise its performance. In addition, an SVM classifier is also created with the *class_weight* parameter set to "balanced", to account for class imbalance. To determine the optimal

hyperparameter settings for the SVM model, a grid search method is also employed. The parameter grid is defined, specifying different values for the following hyperparameters [84]:

- *C*: this hyperparameter represents the regularization parameter in SVM, controlling the trade-off between minimizing training error and allowing for a larger margin. The values tested for C were 0.1, 1, and 10. These values were selected to assess the impact of varying levels of regularization on the model's performance.
- *kernel*: defines the type of decision boundary to be learned by the SVM model. The options considered were 'linear', 'rbf' (radial basis function), and 'poly' (polynomial). Exploring different kernels allows us to examine how different types of decision boundaries affect the model's ability to capture the underlying patterns in the data. Exploring different kernels allows for an examination of how different types of decision boundaries can affect the SVM model's ability to capture the underlying patterns in the data.
- *gamma*: this hyperparameter controls the flexibility of the decision boundary in SVM. It determines the influence of individual training examples on the decision boundary. The values tested for gamma were 0.1, 1 and 10, which allows the evaluation of how the model performance varies with different degrees of flexibility in the decision boundary.

Similar to what has been explained for the previous algorithm, cross-validation is employed to assess the model's generalization performance, dividing the dataset into five folds where the SVM model is trained on four and evaluated on the remaining one. Through grid search and evaluation of the F1 score, the optimal hyperparameter configuration for the SVM model is determined to maximize its performance. The resulting hyperparameter settings from the grid search are then used to evaluate the SVM model's accuracy on the dataset, ensuring accurate and generalized predictions. The best hyperparameter configurations for "has pit stop" and "good pit stop" are presented in Table 3, representing the settings that yield the highest accuracy based on the grid search process. This comprehensive cross-validation approach provides a thorough assessment of the model's ability to generalize.

In this case, unlike Random Forest, the best hyperparameters are common for both output variables. This suggests that, based on the analysis performed through grid search, the SVM model finds the same decision boundary or optimal solution for both variables "Has pit stop" and "Good pit stop". This implies that the model perceives a

Table 3: Best Hyperparameter Configuration for the SMV Classifier.

Hyperparameters	Has pit stop	Good pit stop
C	0.1	0.1
gamma	1	1
kernel	poly	poly

similar relationship or pattern between the input characteristics and these two output variables. The different hyperparameter settings between SVM and Random Forest can be attributed to the intrinsic differences in their underlying algorithms and the different behaviours these classifiers exhibit when adapting to the dataset.

3.2.3 Artificial Neural Networks

Finally, the performance of the neural networks in predicting both target variables will be evaluated. The neural network structure is derived from the work presented in [69], which has been thoroughly analysed and refined through experimentation with various combinations of hyperparameters and network architecture.

To handle class imbalance in the dataset, a method known as class weighting is employed. This technique assigns higher weights to the minority class and lower weights to the majority class. By doing so, the model gives more importance to the underrepresented class during training, which helps improve its ability to correctly classify instances from both classes. This approach aims to mitigate the impact of disproportionate class representation and improve the model's performance in handling imbalanced data. In this instance, the class weights have been calculated based on the distribution of classes in the training set. By adjusting the weights, the learning algorithm gives more importance to the minority class during the training process, which allows it to effectively capture the underlying patterns and improve the predictive performance of the minority class. The use of class weights helps mitigate potential bias towards the majority class, ensuring a more balanced learning process and improving the model's ability to handle unbalanced data.

A grid search method is used to determine the ideal hyperparameter settings and architecture to optimize the performance of the ANN model. The hyperparameter grid has distinct values for different hyperparameters. The following hyperparameters are being examined [85]:

- *Hidden layer configuration*: The hidden layer configuration determines the number and size of the ANN hidden layers. Specifically, three configurations are considered: [64, 64], [64, 64, 64] and [128, 64, 32]. Each configuration

represents a different arrangement of hidden layers, and the numbers indicate the number of nodes (neurons) within each layer. By varying the structures of the hidden layers, the aim is to investigate how the complexity and capacity of the network, influenced by the number and size of the hidden layers, affect the model's ability to capture and learn from the underlying patterns in the data.

- *L2 regularization parameter*: This parameter controls the amount of regularisation applied to the weights in the ANN model, helping to prevent over-fitting. Values of 0.0001, 0.0005 and 0.001 are tested to assess the impact of different levels of regularisation on model performance.
- *Dropout rate*: Dropout is a regularization technique to mitigate overfitting by randomly deactivating a portion of the input units during each training iteration. By doing so, dropout reduces the network's dependence on specific features, promoting more robust and generalized learning. Different dropout rates of 0.2, 0.3, and 0.4 are considered to evaluate the effect of dropout on the model's performance.
- *Batch size*: It refers to the number of training samples processed in one forward and backward pass during the training of a model. It plays a role in balancing computational efficiency and the quality of weight updates. By experimenting with different batch sizes, such as 128, 256, and 512, the aim is to assess how the choice of batch size affects the model's performance and identify the trade-offs between computation time and the effectiveness of weight updates during training.
- *Number of epochs*: It represents the total number of iterations through the training dataset during the training process. It directly affects the convergence and learning capacity of the model. The impact of different epoch values, specifically 20, 30, and 40, is investigated to assess how the number of training iterations influences the performance of the model.

After conducting the grid search with cross-validation, and evaluating different combinations of hyperparameters, the optimal configuration for the ANN model is determined based on the highest F1 score. The resulting best hyperparameter configurations for both of the output variables are then presented in Table 4, representing the specific values that yield the highest F1 score based on the grid search process.

In addition to hyperparameter exploration, some parameters remain fixed in all model configurations. The "nadam" optimiser is used, which combines the advantages of adaptive momentum estimation (Adam) and Nesterov momentum. The activation

Table 4: Best Hyperparameter Configuration for the ANN model.

Hyperparameters	Values
Hidden layer configuration	[64, 64, 64]
L2 regularization parameter	0.0001
Dropout rate	0.2
Batch size	256
Number of epochs	30

function "relu" is applied to all hidden layers, while the output layer uses the activation function "sigmoid" for binary classification. The loss function is "binary_crossentropy", suitable for binary classification tasks. The F1 score is selected as the evaluation metric to assess model performance in terms of accuracy and recall. Furthermore, to prevent overfitting, early stopping is utilized as a regularization technique. This involves monitoring the validation loss during the training process and restoring the best model weights based on a predetermined patience threshold, such as 5. If the validation loss does not improve for five consecutive epochs, the model training will stop, preventing further training that may lead to overfitting.

4 Results

The following section presents the results obtained from applying three distinct machine learning methods, namely random forest, SVM, and neural networks. These methods were employed to predict two target variables: "has pit stop" and "good pit stop." This section aims to provide a comprehensive analysis and evaluation of the outcomes obtained from each method, highlighting their respective performances in predicting the target variables.

The target variable "has pit stop" indicates whether a driver made a pit stop during a specific lap in a race. It is a binary variable that represents whether a pit stop occurred or did not occur on that particular lap. On the other hand, the variable "good pit stop" assesses the performance of a pit stop. It indicates whether a pit stop was successful or not based on the change in the driver's position following the stop. If the position either improved or remained the same after the pit stop, it is considered a positive outcome. Conversely, if the position worsened, it is classified as a non-optimal outcome. By considering both variables, we can examine the presence of pit stops and evaluate their effectiveness in terms of influencing the driver's position during the race.

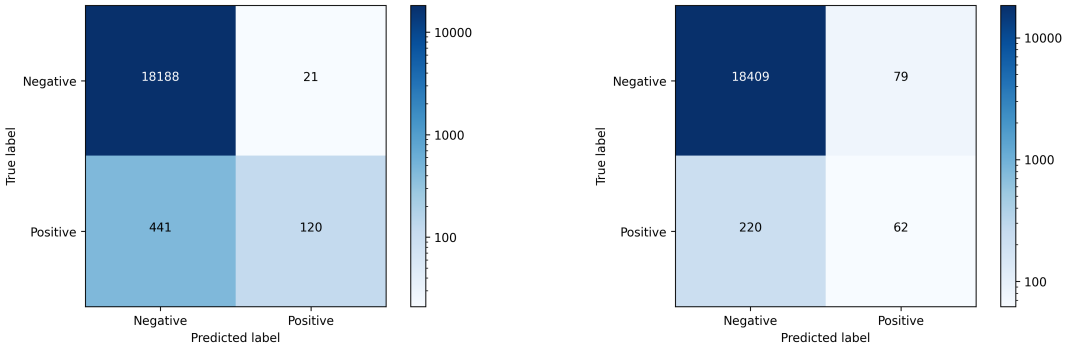
The analysis involved training and testing the machine learning models using different hyperparameters. Two scenarios were considered: one with the original data and another with the data transformed using Principal Component Analysis. However, upon comparing the results, it was observed that the models trained on the original data consistently outperformed those trained on the PCA-transformed data. The performance metric, F1 score, of the models trained without PCA was consistently up to 15% better. As a result, the analysis in this section focuses exclusively on the results obtained using the original data, as it yielded superior performance in predicting the target variables.

Three machine learning algorithms are used: Support Vector Machines (SVM), Random Forest and Neural Networks. These algorithms have been specifically selected for their effectiveness in classification tasks and their ability to capture complex patterns and relationships in data. By comparing and evaluating the performance of these algorithms, the study aims to identify the most suitable approach to accurately predict pit stop requirements. For each algorithm, the dataset is pre-processed and then this data is used to train and optimise the models, considering different hyperparameter settings and techniques such as cross-validation and grid search.

4.1 Random Forest

Presented here are the achieved results utilizing the random forest algorithm, with a specific emphasis on the optimal combination of hyperparameters determined through the grid search methodology as discussed earlier. The subsequent analysis centres on evaluating the outcomes obtained from employing the best combination of hyperparameters for the random forest model.

The confusion matrix serves as a valuable tool to assess the Random Forest classifier's performance on the given variables. It enables a comprehensive evaluation of the model's ability to correctly classify instances and identify any misclassifications, contributing to a better understanding of its overall effectiveness. By analyzing the true positives, true negatives, false positives, and false negatives, we can gain insights into the classifier's strengths and weaknesses in predicting the target variables accurately.



(a) Has pit stop target variable

(b) Good pit stop target variable

Figure 9: Confusion Matrix for Random Forest Classifier.

In the case of the "has pit stop" variable, the model correctly identified many instances with pit stops but had a relatively higher number of false positives. For the "good pit stop" variable, the model exhibited a similar trend, correctly identifying a substantial number of good pit stops but also having a relatively higher number of false positives. Considering the data imbalance, where the number of instances without a pit stop or non-optimal pit stops outweighs the number of instances with pit stops or good pit stops, it is crucial to take this into consideration when interpreting the results. The observed higher number of false positives may be attributed to the model's tendency to predict the majority class more frequently. However, it is essential to note that the model still achieved a considerable number of true positives, indicating its ability to correctly identify instances with pit stops or good pit stops.

Additionally, Table 5 displays the metrics associated with the confusion matrix, such

as precision, recall, and F1 score. The combination of the confusion matrix plot and the metric results provides valuable insights into the predictive capabilities and overall performance of the random forest classifier for the given problem.

Table 5: Performance Metrics for Random Forest Classifier.

Metrix	Has pit stop	Good pit stop
Accuracy	0.977	0.984
Precision	0.812	0.462
Recall	0.241	0.25
F1-score	0.372	0.325

Considering the class imbalance in the dataset, the high accuracy achieved by the random forest classifier should be interpreted with caution. While accuracy is commonly used as a metric to evaluate overall correctness, it can be misleading when dealing with imbalanced data. In this case, the high accuracy may be driven by the model's ability to accurately predict the majority class (instances without a pit stop or non-optimal pit stops), rather than its performance in identifying the minority class (instances with pit stops or good pit stops).

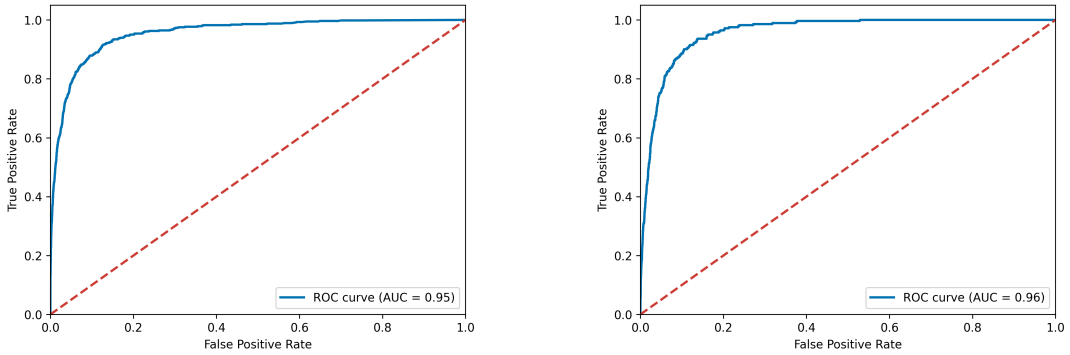
Upon examining the additional evaluation metrics of precision, recall, and F1 score, it becomes apparent that the model's performance is less satisfactory in certain areas. For the "has pit stop" variable, the recall score indicates that the model struggles to capture a significant number of instances with pit stops, resulting in a higher rate of false negatives. Similarly, for the "good pit stop" variable, both precision and recall scores are relatively low, indicating that the model struggles to accurately identify instances of good pit stops, leading to a higher number of false positives and false negatives. The F1-score, which considers both precision and recall, provides a balanced evaluation of the model's performance. The lower F1 scores for both variables suggest that the model's ability to correctly identify instances with pit stops or good pit stops is compromised. This means that the model may miss a substantial number of true instances and produce a higher rate of incorrect predictions.

To evaluate the performance of the machine learning model, two key metrics are also closely examined: training error (E_t) and validation error (E_v). The training error measures the accuracy of the model on the training data, representing how well the model fits the available information. On the other hand, the validation error quantifies the model's ability to generalize to new, unseen data, providing insights into potential overfitting.

For the "Has pit stop" variable, the trained machine learning model demonstrates

an exceptionally low E_t of approximately 0.00048, indicating a highly accurate fit to the training data. However, a relatively higher E_v of around 0.0245 is observed, suggesting some challenges in generalizing the model's performance to new data. This disparity could be attributed to the model potentially memorizing specific patterns or noise in the training data, leading to limited generalization. Similarly, for the "Good pit stop" variable, the model achieves a comparable E_t of approximately 0.00043 but showcases a lower E_v of around 0.0162. This suggests that the model for "Good pit stop" performs better in generalization to unseen data compared to the "Has pit stop" model. The findings underscore the need for further evaluation and optimization to enhance both models' ability to capture the underlying complexity of the data and achieve improved generalization for more accurate predictions.

In addition to having a more comprehensive analysis of the random forest classifier's predictive capabilities, in Figure 10, the receiver operating characteristic (ROC) curve and the corresponding area under the curve (AUC) score for the random forest classifier are presented. The AUC score provides a measure of the classifier's performance in distinguishing between positive and negative instances, with a higher score indicating better classification performance.



(a) Has pit stop target variable

(b) Good pit stop target variable

Figure 10: ROC Curve and AUC Score for Random Forest Classifier.

The random forest classifier demonstrates strong discrimination ability with high AUC scores of 0.95 for the "has pit stop" variable and 0.96 for the "good pit stop" variable. However, it is important to consider the impact of data imbalance on the interpretability of these AUC scores. The dominance of the majority class in the dataset can influence the AUC metric and potentially obscure the model's true performance in accurately identifying instances with pit stops or good pit stops. Similar to the accuracy metric, the AUC may not provide the most informative analysis in this context. Therefore, for more meaningful comparison and evaluation, the F1-score metric is preferred as it

considers both precision and recall, providing a balanced assessment of the model's performance in identifying instances with pit stops or good pit stops.

The random forest classifier serves as a powerful tool for uncovering the significance of individual features when predicting the target variables. The analysis of Figures 11 and 12 showcases the top 20 feature importances specifically for the "has pit stop" and "good pit stop" variables, respectively. This detailed examination allows for a deeper comprehension of the distinct contributions made by various features in predicting each output variable. By carefully comparing the feature importance presented in both figures, valuable insights emerge regarding the specific factors that hold significant influence over the occurrence and quality of pit stops.

Notably, the shared presence of the first three variables between both target variables suggests their universal importance in the predictive process. However, as we delve into the remaining features, we observe distinct variations in their order of importance, with some even being unique to each variable. This disparity in feature importance underscores the fact that different factors are weighed when predicting the occurrence of a pit stop compared to predicting a good pit stop. These variations shed light on the specific aspects that receive prioritization for each output variable.

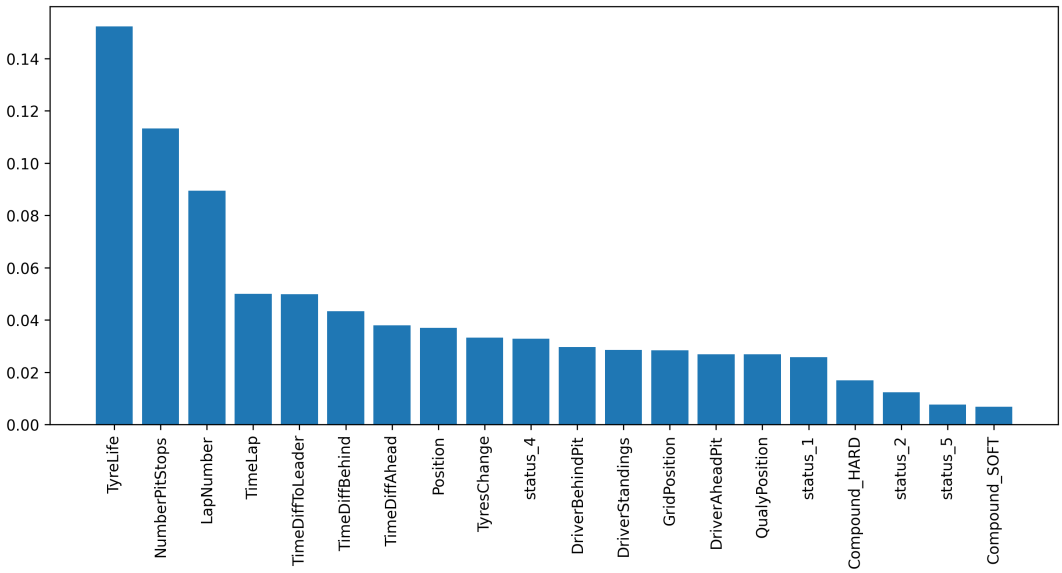


Figure 11: Top 20 Feature Importances for Has pit stop target variable.

Based on the achieved results for the random forest algorithm, it becomes evident that utilizing an appropriate metric is crucial for data comparison. Given the imbalanced nature of the data, certain metrics can create a misleading impression of significantly better results, whereas, in reality, the algorithm's predictive ability is not as strong for

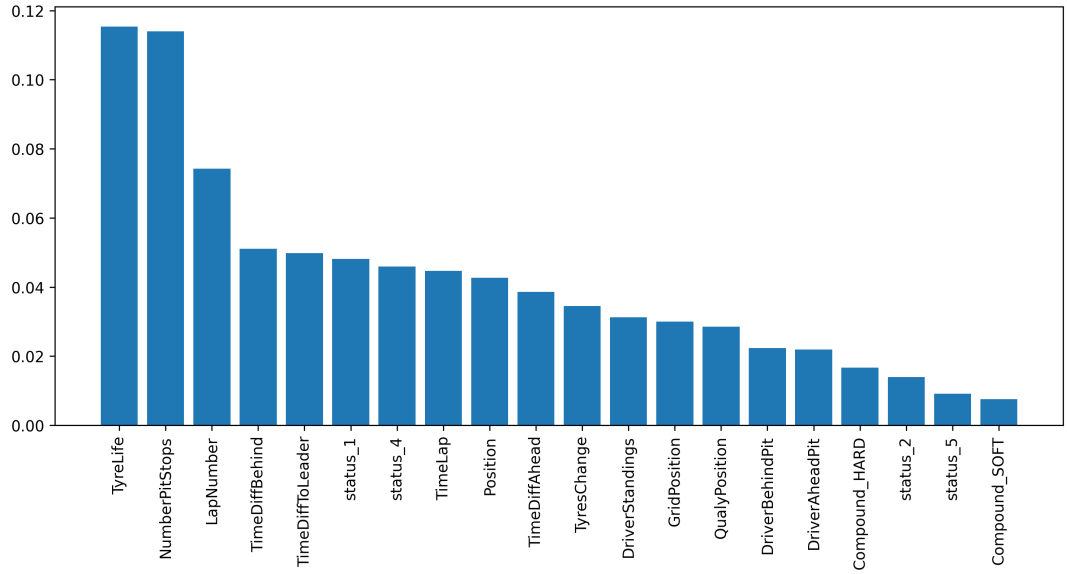


Figure 12: Top 20 Feature Importances for Good pit stop target variable.

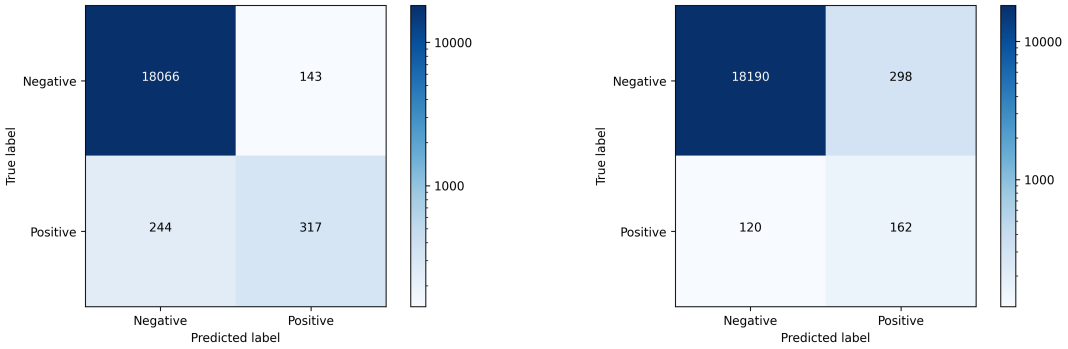
both variables. However, it does exhibit a slightly better performance when predicting the "has pit stop" variable. But there is still room for improvement in the predictive capabilities of the model for both variables.

4.2 Support Vector Machine

The SVM algorithm was utilized to predict the two output variables. The SVM model was trained using the best combination of hyperparameters determined through the grid search methodology explained earlier. These optimal hyperparameters were selected to maximize the SVM model's predictive performance.

For the SVM classifier, the confusion matrix depicting its predictions for both variables is illustrated in Figure 13. This matrix offers a visual representation of the classifier's performance, highlighting true positives, true negatives, false positives, and false negatives.

The results observed in the confusion matrices follow a similar pattern to what could be observed in the Random Forest algorithm. Taking into account the imbalance in the data, where the majority class outperforms the minority class, the SVM model had difficulties in accurately predicting the minority classes. This resulted in a relatively higher number of false positives, as the model tended to predict the majority class more often. However, it is important to note that the SVM model still achieved a significant number of true positives, indicating its ability to correctly identify instances with pit



(a) Has pit stop target variable

(b) Good pit stop target variable

Figure 13: Confusion Matrix for SVM Classifier.

stops or good pit stops.

To further assess the performance of the SVM classifier, Table 6 presents the associated metrics derived from the confusion matrix, such as precision, recall, and F1 score. Analyzing these metrics in conjunction with the confusion matrix aids in understanding the predictive capabilities and overall performance of the SVM classifier. Due to the imbalanced nature of the data, these are the metrics that provide the most information on the performance of the models, as derived from the Random Forest method results.

Table 6: Performance Metrics for SVM Classifier.

Metrix	Has pit stop	Good pit stop
Accuracy	0.979	0.978
Precision	0.689	0.352
Recall	0.565	0.574
F1-score	0.621	0.437

When examining the performance metrics for the SVM classifier, the results indicate relatively better performance for the "has pit stop" variable compared to the "good pit stop" variable. Although the accuracy achieved for both variables is high, which may suggest that the model generally makes correct predictions, the reality is that given the imbalance in the data, this metric cannot be used as a comparator on its own.

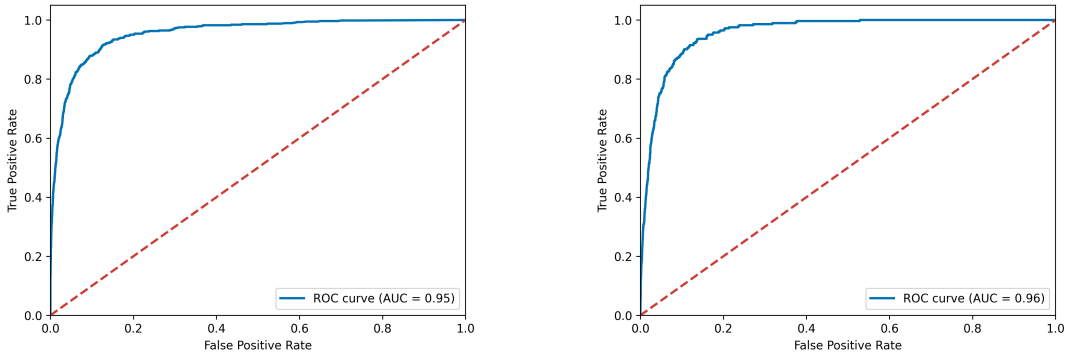
By analysing the precision, recall, and F1-score, we can obtain more information about the performance of the SVM model. For the variable "has pit stop", the precision score suggests that the model can correctly identify cases with pit stops with a reasonable level of confidence. The recall score indicates that the model captures a significant portion of the cases with pit stops, although there is still room for improvement. The F1 score, which combines precision and recall, provides a balanced assessment of the

model's performance, in this instance compared to the others, it is relatively higher indicating a better performance in predicting this variable.

In contrast, the performance metrics for the "good pit stop" variable reveal some problems. The precision score suggests that the model's predictions of good pit stops may not be as accurate. The recall score indicates that the model may miss a considerable number of cases with good pit stops. The F1 score highlights that the model is not able to correctly predict cases of good pit stops.

The comparison of the training error (E_t) and validation error (E_v) for the SVM model provides valuable insights into its performance for the "Has pit stop" and "Good pit stop" variables. For the "Has pit stop" variable, the SVM achieves a low E_t of approximately 0.00023, indicating an accurate fit to the training data. However, the E_v is relatively higher at around 0.0206, suggesting that the model may not generalize well to unseen data. For the "Good pit stop" variable, the SVM model demonstrates a higher E_t of approximately 0.0134, potentially indicating challenges in fitting the training data effectively. However, the E_v remains comparable at around 0.0223, suggesting a relatively better ability to generalize compared to the "Has pit stop" variable. Overall, the comparison highlights the need for further optimization and refinement of the SVM model to enhance its generalization and predictive accuracy for both variables.

Furthermore, including the ROC curve and the corresponding area under the curve score in Figure 14 allows for a more comprehensive analysis of the SVM classifier's performance. The ROC curve and AUC score serve as additional metrics that complement the information provided by the confusion matrix and associated metrics.



(a) Has pit stop target variable

(b) Good pit stop target variable

Figure 14: ROC Curve and AUC Score for SVM Classifier.

As with the Random Forest results, there are very good results when analysing the ROC curve and the AUC score, giving a misleading picture of how good the model is

at predicting both variables.

Overall, the SVM classifier exhibits superior performance for the "has pit stop" variable compared to the "good pit stop" variable It demonstrates promising accuracy in identifying instances with pit stops, as evidenced by reasonable precision and recall scores. The SVM model successfully captures a significant portion of instances with pit stops, highlighting its effectiveness in this aspect. However, there is still potential for improvement in accurately predicting instances of good pit stops.

When comparing the performance of the SVM classifier with the random forest classifier, it becomes apparent that the SVM model outperforms the random forest for both the "has pit stop" and "good pit stop" variables. However, it is important to note that there is a notable disparity in performance between the two variables. The SVM classifier accurately identifies instances with pit stops, showcasing a significant improvement over the random forest classifier, although this does not mean that it predicts perfectly, there is still room for improvement. On the other hand, while the SVM classifier also demonstrates relatively better performance for the "good pit stop" variable, there is room for further refinement in correctly predicting instances of good pit stops.

4.3 Artificial Neural Networks

Finally, in this section, the analysis revolves around the results obtained using the best hyperparameters and network structure for both target variables. Unlike the previous approaches, the emphasis is placed on examining the evolution of the loss and F1 score over the epochs. This metric serves as a key indicator of the algorithm's performance, offering valuable insights into the learning process and improvement of the ANNs over time.

Starting with the "has pit stop" variable, Figure 15 shows the performance of the ANN training and testing with this target variable, the evolution over the epochs of both the loss and the F1 score.

The loss function acts as a measure of the difference between the predicted and actual values of the target variable. Throughout the training process, the main goal is to minimize this loss, as it indicates a better alignment between the model's predictions and the true values. As the training epochs advance, a decreasing loss indicates that the model is learning from the data and improving its predictive capabilities. On the other hand, the F1 score is a metric that combines precision and recall, offering a well-balanced evaluation of the model's overall performance. It considers both false

positives and false negatives, making it a valuable metric for assessing the model's ability to handle class imbalances and make accurate predictions. The desired trend for the F1 score is an upward trajectory, indicating that the model is becoming more adept at correctly classifying positive and negative instances. As the epochs advance, an increasing F1 score demonstrates the model's improving ability to capture meaningful patterns and make reliable predictions.

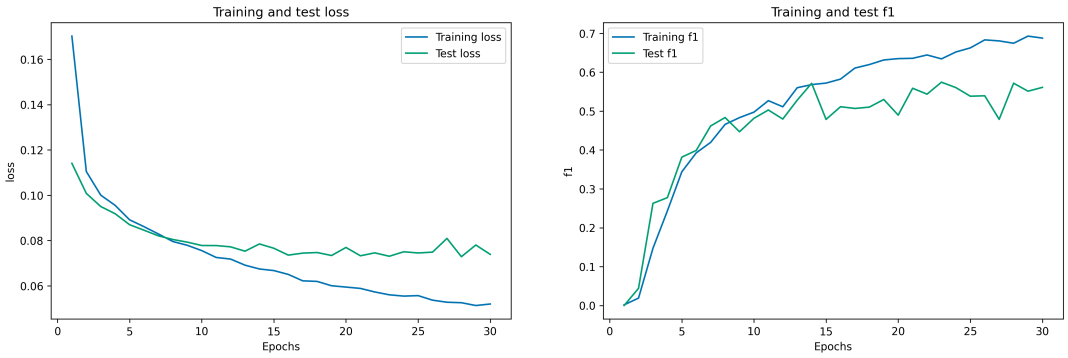


Figure 15: Training and Testing Performance for ANN with Pit Stop Target Variable.

The results obtained with the trained ANN for this variable show promising trends. The loss, which represents the deviation between predicted and actual values, decreases systematically on both training and test data sets. This reduction indicates that the model is effectively learning from the training data and successfully generalising it to the unseen test data.

Furthermore, the F1 scores, which at the end of training reach a value of almost 0.6, indicate that the neural network can correctly identify cases with pit stops with a reasonable level of confidence, especially considering the results we have obtained for the other algorithms.

Although the results obtained are relatively promising, a closer examination reveals the presence of some overfitting in the neural network, despite the implementation of various measures to address this issue and enhance the generalizability of the model. This is evidenced by the disparity between the performance of the training data and that of the test data, where the training performance outperforms the test data.

On the other hand, the results for the second target variable, "good pit stop", are presented in Figure 16. This figure provides an overview of the performance of the neural network in predicting the Good pit stop variable throughout the training and testing process.

Despite observing a similar trend for the Good pit stop variable compared to the

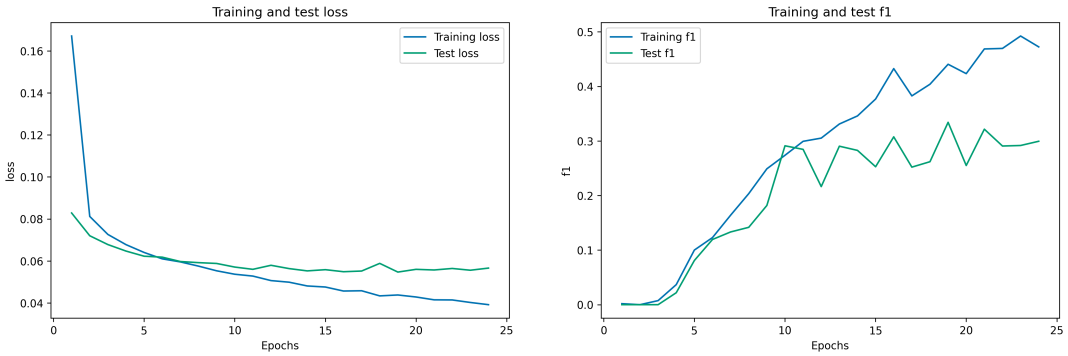


Figure 16: Training and Testing Performance for ANN with Good Stop Target Variable.

previous variable, the results are notably worse. Higher overfitting becomes evident, with a larger disparity between the performance on the training and testing data. In terms of the F1 score, we observe a lower value of 0.36 for this variable. This indicates that the neural network's ability to correctly identify cases with good pit stops is relatively limited.

Analyzing the comparison between the training error and validation error for the target variables using the trained ANN model yields valuable insights into their predictive performance. For the "Has pit stop" variable, the ANN exhibits a training error of approximately 0.0479 and a validation error of around 0.0726. The relatively higher validation error suggests potential challenges in generalizing effectively to unseen data, indicating a possible presence of overfitting, something that has already been indicated above. In contrast, for the "Good pit stop" variable, the ANN demonstrates a training error of approximately 0.0344 and a validation error of around 0.0545. Although the validation error remains higher than the training error, the smaller gap compared to the "Has pit stop" variable indicates a relatively better ability to generalize. Despite this, a closer examination reveals a higher degree of overfitting in the "Good pit stop" model, as evidenced by the larger disparity between the performance on the training and test data sets.

The analysis of the trained Artificial Neural Network models for the "Has pit stop" and "Good pit stop" variables provides valuable insights into their predictive performance. The ANN for the "Has pit stop" variable shows promising trends with decreasing loss and increasing F1 scores, indicating successful learning and improved predictive capabilities. However, some overfitting is evident, necessitating further optimization to enhance generalizability. Conversely, the ANN's performance for the "Good pit stop" variable is notably worse, with higher overfitting and limited ability to identify

cases with good pit stops accurately. Balancing underfitting and overfitting remains a challenge.

4.4 Comparative Analysis

To provide an overview of the results obtained from the analysis of the three algorithms, Table 7 presents the F1 score results achieved using the best combinations for both target variables.

Table 7: Comparison of F1 score.

Algorithms	Has pit stop	Good pit stop
RF	0.372	0.325
SVM	0.621	0.437
ANN	0.593	0.360

The analysis reveals that SVM outperforms the other algorithms in predicting both pit stop occurrence and pit stop quality. It achieves the highest F1 scores for both variables, indicating its superior performance in capturing the patterns and characteristics of pit stops. Neural networks, although slightly behind SVM, also demonstrate competitive results, in particular for the variable "Has pit stop". On the other hand, the random forest shows lower F1 scores for both variables, indicating a comparatively weaker performance in pit stop prediction.

It is important to note the notable differences between the results obtained for the two target variables. While the algorithms show promising results in predicting the occurrence of pit stops (variable "Has pit stop"), neither of them achieves satisfactory values for assessing the quality of pit stops (variable "Good pit stop"). This suggests that accurately determining whether a pit stop will lead to a beneficial outcome is a more difficult task, as it depends on several factors beyond the scope of the analysed data, such as team strategy and race dynamics.

The best-performing algorithm for predicting whether to pit stop or not, SVM, was utilized to predict the outcomes of an entire race. Since the thesis coincides with the ongoing 2023 season, race data from this season was excluded from the training and testing process of the models. However, the first race of the 2023 season was employed to visualize how the model would perform in a real race scenario.

Figure 17 presents a comprehensive overview, illustrating the predictions made by the model for each driver and each lap. It showcases whether a pit stop was predicted and

executed on a given lap, only predicted but not executed, as well as instances where a pit stop was executed but not predicted accurately.

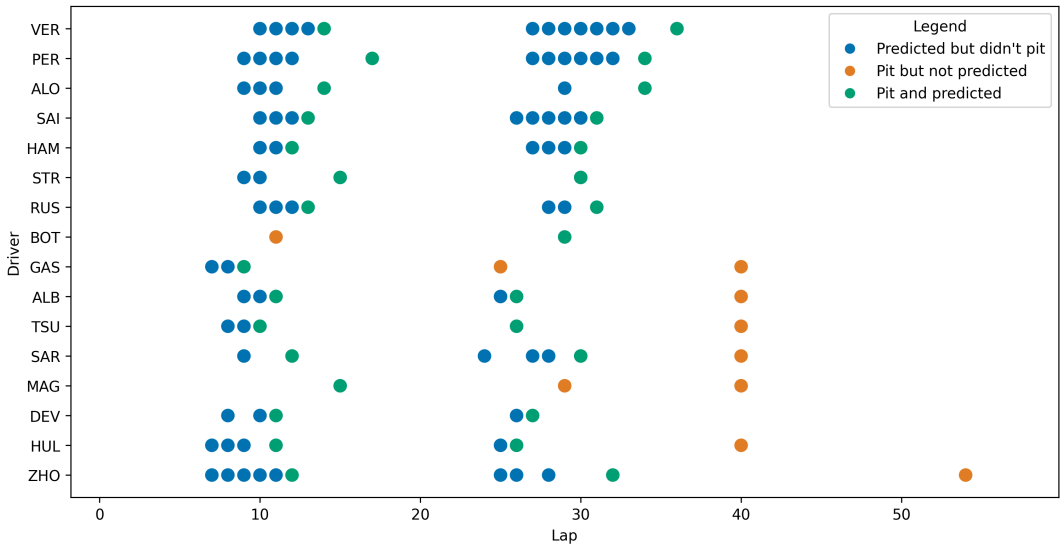


Figure 17: Visualization of Pit Stop Predictions and Executions in the First Race of the 2023 Season.

The analysis of the algorithm's performance in predicting optimal pit stop timing for the first race of the 2023 season reveals some interesting patterns. In the majority of cases, the algorithm consistently predicts the optimal time for a pit stops a few laps prior to its actual execution. This suggests that the algorithm is capturing significant indicators and trends in the race data, enabling it to make proactive predictions.

However, it is important to note that there are specific instances, particularly during the third pit stop for drivers in less favourable positions, where the algorithm fails to accurately predict the appropriate timing. This discrepancy can be attributed to the unique strategies adopted by drivers in disadvantaged positions. These drivers may opt for unconventional approaches, taking calculated risks to improve their race outcomes. As a result, the algorithm's predictions may not align with their strategic decisions, leading to inaccurate timing predictions.

In the remaining cases, although the algorithm accurately predicts the exact lap for a pit stop in some instances, the majority of predictions occur a couple of laps earlier than the actual stop. This suggests that the algorithm has a tendency to anticipate the need for a pit stop a few laps before it is ultimately executed. This behavior can be beneficial in providing drivers with ample time to plan and make necessary adjustments in their race strategy.

Overall, the analysis highlights both the strengths and limitations of the algorithm in predicting optimal pit stop timing. While it demonstrates consistent and proactive predictions in many cases, it may encounter challenges in accurately predicting timing for drivers in disadvantaged positions who adopt unconventional strategies. These results serve to highlight the complexity of the pit stop strategy.

5 Conclusions

In this thesis, various hyperparameter configurations were explored to obtain the best results for three different machine learning algorithms: Random Forest, Support Vector Machines (SVM), and Neural Networks. The objective was to predict whether a driver should make a pit stop in a given lap of a Formula 1 race using historical race data from the seasons spanning 2019 to 2022.

Two target variables were considered, both based on the real outcomes of the races analysed: "has pit stop" and "good pit stop". The "has pit stop" variable indicates the occurrence or non-occurrence of a pit stop on a specific lap, while the "good pit stop" variable assesses the performance of a pit stop based on the change in the driver's position after the stop, positive outcomes indicate an improvement or maintenance of position, while negative outcomes signify a decrease in position.

To optimize the performance of the algorithms, different hyperparameter configurations were tested. The F1 score was used as the primary measure to evaluate the models' performance. By comparing the results obtained from the different configurations, the study aimed to identify the best approach to pit stop prediction in Formula 1. The goal is to provide valuable insights for optimizing race strategies and making informed pit stop decisions in Formula 1.

5.1 Discussion

The findings of this thesis provide insights into the challenging nature of predicting pit stops in Formula 1 races. Firstly, the results for the "has pit stop" variable demonstrate promising potential. The models exhibited reasonable accuracy in predicting whether a driver would make a pit stop during a specific lap. However, it is important to note that there is still room for improvement in achieving more precise predictions. Pit stop prediction is a highly complex task, the interaction of multiple variables and the inherent uncertainties involved make it challenging to capture all the nuances and accurately predict the occurrence of pit stops. The findings reaffirm that achieving perfect predictions in this area is not realistic, but the models can serve as valuable tools to assist decision-making processes.

On the other hand, the results for the "good pit stop" variable indicate the inherent difficulty in determining whether a pit stop will result in a favourable outcome for a driver. It is not possible to know with certainty if a pit stop will be successful or not. Teams make the best decisions based on available information and strategic

considerations, but there is no guarantee of a positive outcome. Moreover, the competitive nature of Formula 1 means that if one driver gains a position due to a pit stop, another driver may lose a position. It is impossible for all drivers to benefit from pit stops simultaneously in a race.

Therefore, it is important to recognise that the models developed in this study cannot be relied upon solely to accurately predict when to perform a sting. However, they can serve as valuable tools to support decision-making processes alongside other relevant information and expert knowledge. By incorporating predictions from the models, teams can improve their understanding of pit stop dynamics and make more informed decisions about race strategies.

In conclusion, this thesis illuminates the potential of machine learning algorithms in enhancing the prediction of pit stops in Formula 1 races. Despite the inherent challenges posed by the complex and uncertain nature of racing dynamics, the models developed in this study offer valuable insights to support race strategists. While achieving absolute perfection in predictions remains elusive, the models serve as essential tools in the decision-making process. As technology and data analytics continue to advance, there lies a promising opportunity to further refine and augment these models to overcome their limitations. By combining the power of accurate predictions with human expertise, teams can effectively fine-tune their race strategies and elevate the overall competitiveness of Formula 1 races. Embracing a data-driven approach in conjunction with the instincts and knowledge of race teams can lead to more strategic and thrilling races, ultimately enriching the Formula 1 experience for fans and participants alike.

5.2 Future Work

Building upon the insights gained from this study, several avenues for future research can be pursued to further advance the field of pit stop prediction in Formula 1. These directions aim to enhance the accuracy and effectiveness of the models while considering the unique challenges and complexities associated with predicting pit stops.

One promising area for future research is the expansion of the variables used in prediction models. By incorporating additional factors that are unique to the teams, models can capture a more complete picture of pit stop dynamics. Collaboration with the teams would be essential to access this unique data and gain insight into their decision-making processes. Integrating these variables into models could lead to more

accurate predictions and better optimisation of race strategies.

In addition, a change of perspective could be explored by incorporating information from previous laps into the prediction models. By analysing historical data and multi-lap patterns, models can capture the evolving trends and dynamics of pit stop strategies. This approach could provide a more complete understanding of pit stop dynamics and improve the accuracy of predictions. Taking into account not only the current lap but also information from previous laps could allow a more complete analysis for the prediction of pit stops and their potential impact on race results.

Moreover, a significant advancement in the field would involve the development of a real-time pit stop prediction system. By analyzing data during live races and integrating real-time data feeds with advanced machine learning algorithms, this system could provide teams with dynamic and timely pit stop predictions. Such a real-time predictor could adapt to changing race conditions, weather, and team strategies, enabling teams to make informed and strategic decisions during the race itself. This real-time predictor has the potential to revolutionize pit stop strategies and enhance team performance, ultimately contributing to more competitive and thrilling Formula 1 races.

References

[1] Fédération Internationale de l'Automobile, "2019 F1 Technical Regulations," 2019.

[2] Liberty Media, "2022 Annual Report 2023 Proxy Statement," 2022.

[3] Fédération Internationale de l'Automobile, "Formula 1 Financial Regulations ," 2022.

[4] "Evolution of Sports through Operation Research," 2022.

[5] E. Alpaydın, *Introduction to Machine Learning*. The MIT Press, 2022.

[6] A. Jung, *Machine Learning: The Basics* . Springer, Singapore, 2022. [Online]. Available: https://alexjungaalto.github.io/MLBasicsBook.pdf

[7] P. Cunningham, M. Cord, and S. J. Delany, "Supervised learning," *Cognitive Technologies*, pp. 21–49, 2008. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-540-75171-7_2

[8] G. Kesavaraj and S. Sukumaran, "A study on classification techniques in data mining," *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, 2013.

[9] J. Dj Novakovi, A. Veljovi, S. S. Ili, Zeljko Papi, and M. Tomovi, "Evaluation of Classification Models in Machine Learning," *Theory and Applications of Mathematics & Computer Science*, vol. 7, no. 1, pp. 39–46, 2017.

[10] M. Usama, J. Qadir, A. Raza, H. Arif, K. L. A. Yau, Y. Elkhatib, A. Hussain, and A. Al-Fuqaha, "Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges," *IEEE Access*, vol. 7, pp. 65 579–65 615, 2019.

[11] T. S. Madhulatha, "An Overview on Clustering Methods," *IOSR Journal of Engineering*, vol. 02, no. 04, pp. 719–725, 5 2012. [Online]. Available: https://arxiv.org/abs/1205.1117v1

[12] C. O. S. Sorzano, J. Vargas, and A. P. Montano, "A survey of dimensionality reduction techniques," 3 2014. [Online]. Available: https://arxiv.org/abs/1403.2877v1

[13] X. J. Zhu, "Semi-Supervised Learning Literature Survey," 2005. [Online]. Available: https://minds.wisconsin.edu/handle/1793/60444

[14] X. Ning, X. Wang, S. Xu, W. Cai, L. Zhang, L. Yu, and W. Li, "A review of research on co-training," *Concurrency and Computation: Practice and Experience*, p. e6276, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/cpe.6276https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.6276https://onlinelibrary.wiley.com/doi/10.1002/cpe.6276

[15] S. Sun, "A survey of multi-view machine learning," *Neural Computing and Applications*, vol. 23, no. 7-8, pp. 2031–2038, 12 2013. [Online]. Available: https://link.springer.com/article/10.1007/s00521-013-1362-6

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[17] H. K. Gianey and R. Choudhary, "Comprehensive Review On Supervised Machine Learning Algorithms," *Proceedings - 2017 International Conference on Machine Learning and Data Science, MLDS 2017*, vol. 2018-January, pp. 38–43, 3 2018.

[18] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 6 2006.

[19] D. M. W. Powers and Ailab, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," 10 2020. [Online]. Available: https://arxiv.org/abs/2010.16061v1

[20] G. Tourassi, "Receiver Operating Characteristic Analysis: Basic Concepts and Practical Applications," *The Handbook of Medical Image Perception and Techniques: Second Edition*, pp. 227–244, 1 2018. [Online]. Available: https://www.cambridge.org/core/books/handbook-of-medical-image-perception-and-techniques/receiver-operating-characteristic-analysis-basic-concepts-and-practical-applications/FF2BC4BEC0B24A799218564AF2C09D2B

[21] M. Stone, "Cross-validation:a review," *https://doi.org/10.1080/02331887808801414*, vol. 9, no. 1, pp. 127–139, 1 2007. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/02331887808801414

[22] D. Berrar, "Cross-Validation Call for Papers for Machine Learning journal: Machine Learning for Soccer View project Cross-validation." [Online]. Available: https://doi.org/10.1016/B978-0-12-809633-8.20349-X

[23] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[24] S. Dash, "Decision Trees Explained — Entropy, Information Gain, Gini Index, CCP Pruning | by Shailey Dash | Towards Data Science," 2022. [Online]. Available: https://towardsdatascience.com/decision-trees-explained-entropy-information-gain-gini-index-ccp-pruning-4d78070db36c

[25] Y. Liu, Y. Wang, and J. Zhang, "New machine learning algorithm: Random forest," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7473 LNCS, pp. 246–252, 2012. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-34062-8_32

[26] P. Probst, M. N. Wright, and A. L. Boulesteix, "Hyperparameters and tuning strategies for random forest," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 3, p. e1301, 5 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/widm.1301https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1301https://wires.onlinelibrary.wiley.com/doi/10.1002/widm.1301

[27] C. Cortes, V. Vapnik, and L. Saitta, "Support-vector networks," *Machine Learning 1995 20:3*, vol. 20, no. 3, pp. 273–297, 9 1995. [Online]. Available: https://link.springer.com/article/10.1007/BF00994018

[28] A. Mammone, M. Turchi, and N. Cristianini, "Support vector machines," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 1, no. 3, pp. 283–289, 11 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/wics.49https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.49https://wires.onlinelibrary.wiley.com/doi/10.1002/wics.49

[29] I. Zoppis, G. Mauri, and R. Dondi, "Kernel Methods: Support Vector Machines," *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, vol. 1-3, pp. 503–510, 1 2019.

[30] A. Patle and D. S. Chouhan, "SVM kernel functions for classification," *2013 International Conference on Advances in Technology and Engineering, ICATE 2013*, 2013.

[31] Ardiansyah, R. Ferdiana, and A. E. Permanasari, "Optimizing SVM Hyperparameters using Predatory Swarms Algorithms for Use Case Points Estimation,"

2022 9th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), vol. 2022-October, pp. 90–95, 2022.

[32] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, no. 8, p. 2554, 1982. [Online]. Available: /pmc/articles/PMC346238/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC346238/

[33] G. Bebis and M. Georgiopoulos, "Feed-forward neural networks," *IEEE Potentials*, vol. 13, no. 4, pp. 27–31, 1994.

[34] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 6999–7019, 12 2022.

[35] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee, "Recent Advances in Recurrent Neural Networks," 12 2017. [Online]. Available: https://arxiv.org/abs/1801.01078v3

[36] M. R. Shaeri, A. M. Randriambololona, and S. Sarabi, "Prediction Accuracy of Artificial Neural Networks in Thermal Management Applications Subject to Neural Network Architectures," *Proceedings of the 8th World Congress on Mechanical, Chemical, and Material Engineering*, 2022.

[37] F. Li, J. M. Zurada, Y. Liu, and W. Wu, "Input Layer Regularization of Multilayer Feedforward Neural Networks," *IEEE Access*, vol. 5, pp. 10 979–10 985, 6 2017.

[38] M. Uzair and N. Jamil, "Effects of Hidden Layers on the Efficiency of Neural networks," *2020 IEEE 23rd International Multitopic Conference (INMIC)*, 11 2020.

[39] O. A. Montesinos López, A. Montesinos López, and J. Crossa, "Fundamentals of Artificial Neural Networks and Deep Learning," *Multivariate Statistical Machine Learning Methods for Genomic Prediction*, pp. 379–425, 2022. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-89010-0_10

[40] A. D. Rasamoelina, F. Adjailia, and P. Sincak, "A Review of Activation Function for Artificial Neural Network," *SAMI 2020 - IEEE 18th World Symposium on Applied Machine Intelligence and Informatics, Proceedings*, pp. 281–286, 1 2020.

[41] R. de Almeida, Y. M. Goh, R. Monfared, M. T. A. Steiner, and A. West, "An ensemble based on neural networks with random weights for online data stream regression," *Soft Computing*, vol. 24, no. 13, pp. 9835–9855, 7 2020. [Online]. Available: https://link.springer.com/article/10.1007/s00500-019-04499-x

[42] D. M. Kline and V. L. Berardi, "Revisiting squared-error and cross-entropy functions for training neural network classifiers," *Neural Computing and Applications*, vol. 14, no. 4, pp. 310–318, 12 2005. [Online]. Available: https://link.springer.com/article/10.1007/s00521-005-0467-y

[43] R. Rojas, "The Backpropagation Algorithm," *Neural Networks*, pp. 149–182, 1996. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-61068-4_7

[44] I. Nusrat and S. B. Jang, "A Comparison of Regularization Techniques in Deep Neural Networks," *Symmetry 2018, Vol. 10, Page 648*, vol. 10, no. 11, p. 648, 11 2018. [Online]. Available: https://www.mdpi.com/2073-8994/10/11/648/htmhttps://www.mdpi.com/2073-8994/10/11/648

[45] R. P. Bunker and F. Thabtah, "A machine learning framework for sport result prediction," *Applied Computing and Informatics*, vol. 15, no. 1, pp. 27–33, 1 2019.

[46] T. Horvat and J. Job, "The use of machine learning in sport outcome prediction: A review," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 10, no. 5, p. e1380, 9 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/widm.1380https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1380https://wires.onlinelibrary.wiley.com/doi/10.1002/widm.1380

[47] H. Van Eetvelde, L. D. Mendonça, C. Ley, R. Seil, and T. Tischer, "Machine learning methods in sport injury prediction and prevention: a systematic review," *Journal of Experimental Orthopaedics*, vol. 8, no. 1, pp. 1–15, 12 2021. [Online]. Available: https://link.springer.com/articles/10.1186/s40634-021-00346-xhttps://link.springer.com/article/10.1186/s40634-021-00346-x

[48] N. Tax and Y. Joustra, "Predicting The Dutch Football Competition Using Public Data: A Machine Learning Approach." [Online]. Available: www.foxsports.nl

[49] D. Prasetio and Harlili, "Predicting football match results with logistic regression," *4th IGNITE Conference and 2016 International Conference on Advanced Informatics: Concepts, Theory and Application, ICAICTA 2016*, 12 2016.

[50] A. Joseph, N. E. Fenton, and M. Neil, "Predicting football results using Bayesian nets and other machine learning techniques," *Knowledge-Based Systems*, vol. 19, no. 7, pp. 544–553, 11 2006.

[51] R. Rein and D. Memmert, "Big data and tactical analysis in elite soccer: future challenges and opportunities for sports science," *SpringerPlus 2016 5:1*, vol. 5, no. 1, pp. 1–13, 8 2016. [Online]. Available: https://springerplus.springeropen.com/articles/10.1186/s40064-016-3108-2

[52] B. Loeffelholz, E. Bednar, and K. W. Bauer, "Predicting NBA Games Using Neural Networks," *Journal of Quantitative Analysis in Sports*, vol. 5, no. 1, 1 2009. [Online]. Available: https://www.degruyter.com/document/doi/10.2202/1559-0410.1156/html

[53] D. Miljković, L. Gajić, A. Kovačević, and Z. Konjović, "The use of data mining for basketball matches outcomes prediction," *SIISY 2010 - 8th IEEE International Symposium on Intelligent Systems and Informatics*, pp. 309–312, 2010.

[54] M. C. Purucker, "Neural network quarterbacking," *IEEE Potentials*, vol. 15, no. 3, pp. 9–15, 8 1996.

[55] J. A. David, R. D. Pasteur, M. S. Ahmad, and M. C. Janning, "NFL prediction using committees of artificial neural networks," *Journal of Quantitative Analysis in Sports*, vol. 7, no. 2, 2 2011. [Online]. Available: https://www.degruyter.com/document/doi/10.2202/1559-0410.1327/html

[56] A. Lalwani, A. Saraiya, A. Singh, A. Jain, and T. Dash, "Machine Learning in Sports: A Case Study on Using Explainable Models for Predicting Outcomes of Volleyball Matches," 6 2022. [Online]. Available: https://arxiv.org/abs/2206.09258v1

[57] M. Sipko, "Machine Learning for the Prediction of Professional Tennis Matches," 2015.

[58] S. Wilkens, "Sports prediction and betting models in the machine learning age: The case of tennis," *Journal of Sports Analytics*, vol. 7, no. 2, pp. 99–117, 1 2021.

[59] A. Khanteymoori, E. Davoodi, and A. R. Khanteymoori, "Horse racing prediction using artificial neural networks Multi-objective Optimization View project Inference of gene regulatory networks View project Horse Racing

Prediction Using Artificial Neural Networks," 2010. [Online]. Available: https://www.researchgate.net/publication/228847950

[60] W. C. Chung, C. Y. Chang, and C. C. Ko, "A SVM-Based committee machine for prediction of Hong Kong horse racing," *Ubi-Media 2017 - Proceedings of the 10th International Conference on Ubi-Media Computing and Workshops with the 4th International Workshop on Advanced E-Learning and the 1st International Workshop on Multimedia and IoT: Networks, Systems and Applications*, 10 2017.

[61] B. Ofoghi, J. Zeleznikow, C. Macmahon, J. Rehula, and D. B. Dwyer, "Performance analysis and prediction in triathlon," *Journal of sports sciences*, vol. 34, no. 7, pp. 607–612, 4 2016. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/26177783/

[62] B. Ofoghi, J. Zeleznikow, C. MacMahon, and D. Dwyer, "A machine learning approach to predicting winning patterns in track cycling omnium," *IFIP Advances in Information and Communication Technology*, vol. 331 AICT, pp. 67–76, 2010. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-15286-3_7

[63] M. Keertish Kumar and N. Preethi, "Formula One Race Analysis Using Machine Learning," *Lecture Notes in Networks and Systems*, vol. 540, pp. 533–540, 2023. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-19-6088-8_47

[64] E. Stoppels, "Predicting Race Results using Artificial Neural Networks," Ph.D. dissertation, 2017.

[65] X. Liu and A. Fotouhi, "Formula-E race strategy development using artificial neural networks and Monte Carlo tree search," *Neural Computing and Applications*, vol. 32, no. 18, pp. 15 191–15 207, 9 2020. [Online]. Available: https://link.springer.com/article/10.1007/s00521-020-04871-1

[66] T. Tulabandhula and C. Rudin, "Tire changes, fresh air, and yellow flags: Challenges in predictive analytics for professional racing," *Big Data*, vol. 2, no. 2, pp. 97–112, 6 2014. [Online]. Available: https://www.liebertpub.com/doi/10.1089/big.2014.0018

[67] C. Choo, "Real-time decision making in motorsports : analytics for improving professional car race strategy," 2015.

[68] A. Greasley, G. Panchal, and A. Samvedi, "The Use of Simulation with Machine Learning and Optimization for a Digital Twin-A Case on Formula 1 DSS," *2022 Winter Simulation Conference (WSC)*, pp. 2198–2209, 2022. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10015299

[69] A. Heilmeier, A. Thomaser, M. Graf, and J. Betz, "Virtual Strategy Engineer: Using Artificial Neural Networks for Making Race Strategy Decisions in Circuit Motorsport," *Applied Sciences 2020, Vol. 10, Page 7805*, vol. 10, no. 21, p. 7805, 11 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/21/7805/htmhttps://www.mdpi.com/2076-3417/10/21/7805

[70] "Ergast Developer API." [Online]. Available: https://ergast.com/mrd/

[71] "FastF1." [Online]. Available: https://theoehrly.github.io/Fast-F1/

[72] "Live Timing." [Online]. Available: https://www.formula1.com/en/f1-live.html

[73] L. Barretto, "How does the 2022 F1 Sprint format work?" 2022. [Online]. Available: https://www.formula1.com/en/latest/article.how-does-the-f1-sprint-work-the-format-explained-ahead-of-imola.4ep6uU8VYFMgftvTYxkZdI.html

[74] Fédération Internationale de l'Automobile, "2023 Formula 1 Sporting Regulations," 9 2022.

[75] Pirelli, "What's new with Pirelli's 2019 Formula 1 tyres?" [Online]. Available: https://press.pirelli.com/whats-new-with-pirellis-2019-formula-1-tyres/

[76] A. Heilmeier, M. Graf, J. Betz, and M. Lienkamp, "Application of Monte Carlo Methods to Consider Probabilistic Effects in a Race Simulation for Circuit Motorsport," *Applied Sciences 2020, Vol. 10, Page 4229*, vol. 10, no. 12, p. 4229, 6 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/12/4229/htmhttps://www.mdpi.com/2076-3417/10/12/4229

[77] J. De Groote, "Overtaking in Formula 1 during the Pirelli era: A driver-level analysis," *Journal of Sports Analytics*, vol. 7, pp. 119–137, 2021.

[78] Fédération Internationale de l'Automobile, "2023 Formula 1 Sporting Regulations," Tech. Rep., 2022.

[79] "How Many Pit Stops In An F1 Race? ," 5 2022. [Online]. Available: https://flowracers.com/blog/how-many-pit-stops-in-f1/

[80] "What Impact Does The Weather Have On F1? ," 2022. [Online]. Available: https://f1chronicle.com/what-impact-does-the-weather-have-on-f1/

[81] J. T. Hancock and T. M. Khoshgoftaar, "Survey on categorical data for neural networks," *Journal of Big Data*, vol. 7, no. 1, pp. 1–41, 12 2020. [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00305-w

[82] scikit-learn, "Preprocessing data." [Online]. Available: https://scikit-learn.org/stable/modules/preprocessing.html

[83] scikit-learn 1.2.2 documentation, "RandomForestClassifier." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[84] ——, "Support Vector Machines." [Online]. Available: https://scikit-learn.org/stable/modules/svm.html#svm

[85] Tensorflow, "Keras." [Online]. Available: https://keras.io/

A Appendix

A.1 Data Visualization

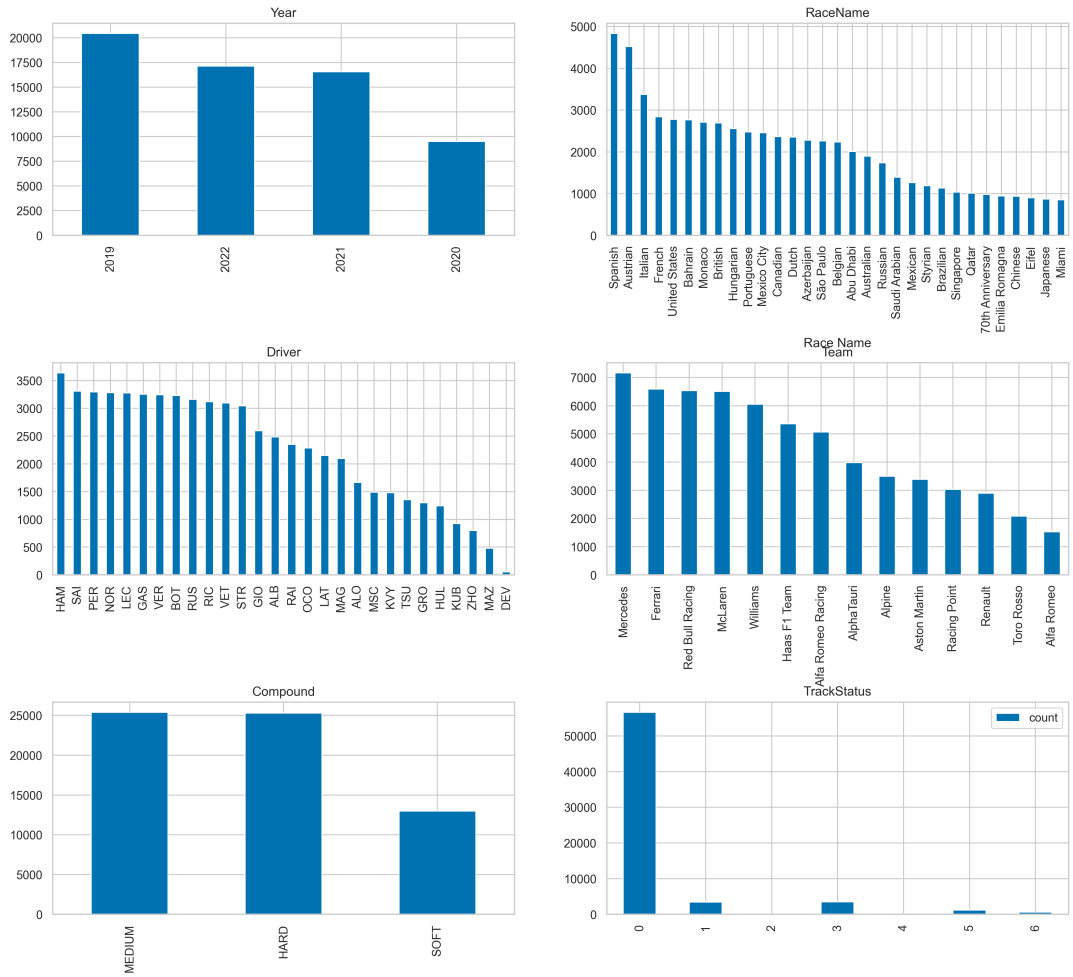


Figure A1: Distribution of categorical variables

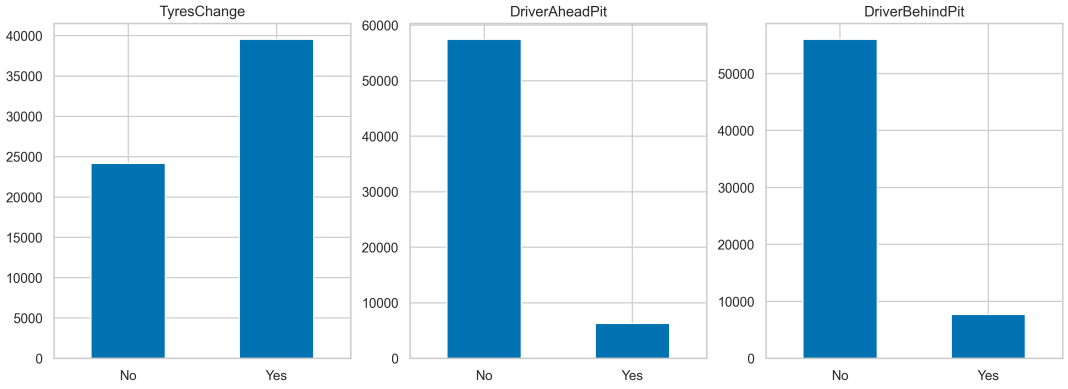


Figure A2: Distribution of binary variables

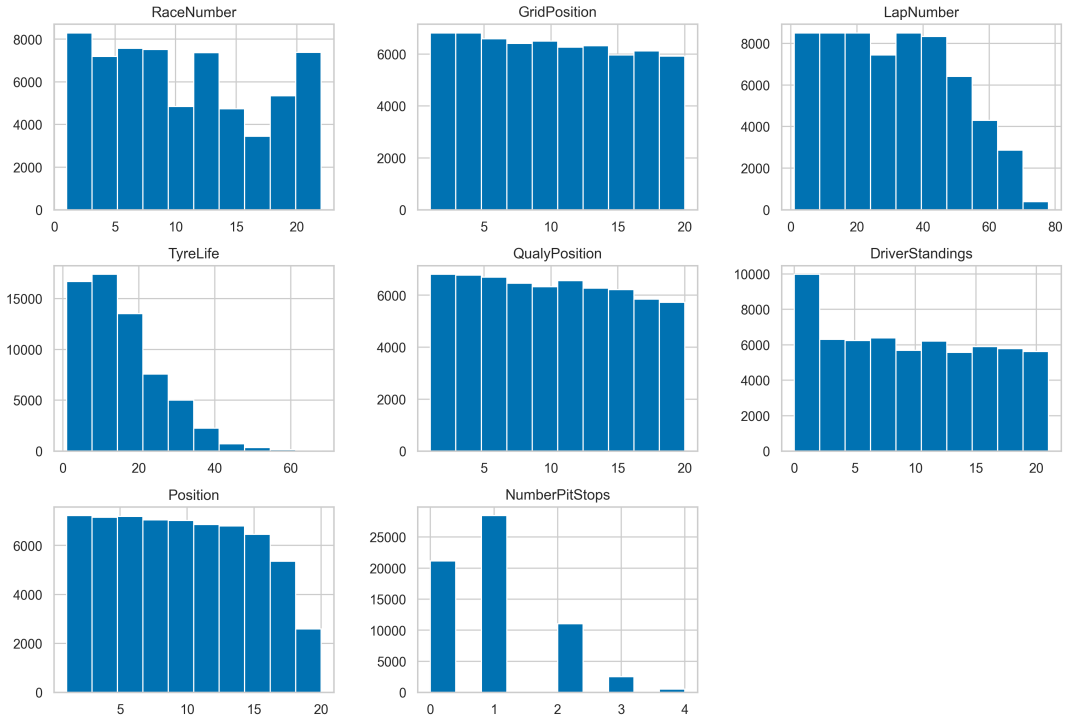


Figure A3: Distribution of numerical variables

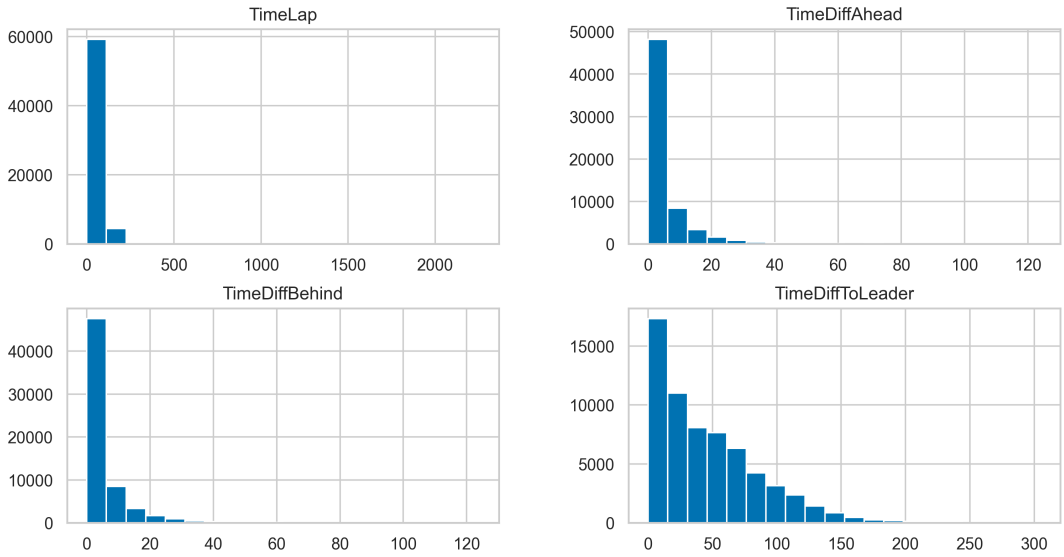


Figure A4: Distribution of date time variables