Predict the winner of the F1 Grand Prix using Machine Learning

Jaspreet singh

Bachelor of Science with Honours Computer and Data Science

School of Computing and Digital Technology

Faculty of Computing, Engineering and the Built Environment

Birmingham City University

Submitted Month 2022

A thesis submitted to Birmingham City University in partial fulfillment of the requirements of the degree of Bachelor of Science.

Abstract

"Races are won at the track. Championships are won at the factory. " (Mercedes, 2019). Predicting race winners in Formula One requires the use of a training and test dataset to operate the machine learning algorithms properly. Thus, data has been collected, cleaned, and an exploratory data analysis has been done to ensure a major understanding of the possible relevant data for the prediction. Since the model is going to predict the winner for each race in different recent seasons, it's possible to see the target variable as a classification or a regression. The first is chosen and the target is mapped before modelling and its output would give a probability for each driver. During the modeling, Logistic Regression (LR), Random Forest Classifier (RFC) and Decision Tree Classifier (DTC) are the methods used in combination with both the Standard Sclaer (SS) and the Min Max Scaler (MMS) in order to achieve variety and confrontation between model results. The results demonstrate the precision of the used classification machine learning models for the prediction of the winners of the races by obtaining a higher score than initially thought could be achieved, especially with the Standard Scaler in combination with the Logistic Regression or the Decision Tree Classifier. Finally, by confronting the predictions with the betting, it's possible to say that the best scoring models formulated during this project can effectively not only beat the odds but even grant a high chance of profits for the general betters.

Acknowledgements

First and foremost, I would like to thank my family and friends for being so supportive and pushing me through this. My mother, grandmother, and friend Shaira are my motivation. Everyone has ensured that I produced a successful dissertation. Without them, I would not have been able to do this. My time at Birmingham City University has been an eventful one, and I am happy that I have created so many amazing memories and experiences. I would also like to thank my supervisor, Essa Shahra, who has provided me with excellent support and has uplifted me when giving me feedback, which has helped me to improve my research goals and given me the tool to properly present this dissertation. His suggestions and useful information during the completion of this dissertation were of extreme importance to the creation and laying off of this document. I would also like to thank every other professor during the various modules throughout the years at BCU who gave me the knowledge to create this dissertation into a more cohesive, structurally-sound project. Thank you once again.

Table of Contents

Glossary

Here an ordered list of symbols and abbreviations with expansions of any contractions to facilitate the understanding of the paper.

F1	Formula 1
GP	Grand Prix
WDC	World Driver Champtionship
IDE	Integrated Development Environment
FIA	Fédération Internationale de l'Automobile
LR	Logistic Regression
RFC	Random Forest Classifier
DTC	Decision Tree Classifier
SS	Standard Scaler
MMS	Min Max Scaler
EDA	Exploratory Data Analysis
AI	Artificial Intelligence
IEEE	Institute of Electrical and Electronics Engineers
ML	Machine Learning
TP	True Positive
TN	True Negative

FP	False Positive
FN	False Negative
TDD	Test Driven Development

List of Figures

Here an ordered list of figures and immages with description to facilitate the understanding of the paper.

List of Tables

Here an ordered list of Tables with description to facilitate the understanding of the paper.

1 Introduction

Predicting winners in sports has always been of interest to many. One such sport is Formula One (F1). While it seems quite an easy motorsport to predict as the winner will be the driver in the fastest car, this is not necessarily the case as lots of other factors contribute to the final result, such as the ability of the driver itself, other than the team's performance in the pits. The various drivers were not able to start or even finish the race. All these possibilities could lead to an unpredictable outcome where having a prediction model based on previous performances could be useful to the team principals to come up with a better strategy and understand where the team's performance lacks in order to have a specified area to improve.

As F1 requires quite extensive amounts of money, in fact some teams could spend up to $145 million per year, (Luke Smith, 2021) this information could also be vital to the sponsors when choosing which team could provide a better performance. The motorsports also allow betting on many sites and the potential for F1 betting is enormous as betting models become more sophisticated and can digest more data sets, the possibilities are endless (contenteditor, 2021) and this project could end up providing some calculated probability to bet more "safely" to the average bettors.

The goal of this project is to develop a machine learning model that can predict the winner of the next F1 Grand Prix (GP) by using previous championship data. After executing an exploratory data analysis (EDA), in order to achieve this goal, the target variable will be categorised as a classification problem. The models will be trained on all the previous data of the season to predict the winners of each race of the chosen recent season. All these steps will be executed through the spyder integrated development environment (IDE) using the Python programming language.

1.1 Problem Definition

Every major sporting event can be bet on and win money, but is there a way to correctly predict the winner of a Formula One Grand Prix in order to gain an advantage in general betting that is more efficient and trustworthy than gut feelings and emotions?

1.2 Scope

This study consists mainly of the machine learning subject in order to apply principles of data collection, cleaning, analysis, and subsequently creation of machine learning models and their testing and evaluation. The presented work's aim is not focused on presenting a new approach but rather searching for ways to improve the prediction model. The models will mostly be tested on a limited range of recent seasons for specific and more relevant accuracy to the present, despite the fact that they can be tested on any given season with the caveat that more data is missing as the season tested gets older.In the same regard, very recent seasons won't be tested as recent regulation changes took place and now the 2022 regulations represent the biggest overhaul in the sport for years. (inews.co.uk , 2022)

1.3 Rationale

Formula One's Drive To Survive series has topped global Netflix viewing and its drivers have accumulated record-breaking Twitch audiences. That's good news for a sport that's struggled with declining popularity for years. Autosport reports a new study by statistics body Nielsen that says a new audience is finding F1, and that the series could be on track for a billion fans worldwide in 2022. (Hazel Southwell, 2021)

As the number of fans of the motorsport increased dramatically in recent years, watching, playing, and even betting on F1 races is now a popular activity for many people. As being an unpredictable motorsport the betting odds could result in some very generous winnings.As a result, the aim of this project is to be able to predict the next F1 Grand Prix winner using statistical techniques.

1.4 Project Aims and Objectives

The goal of this project is to develop machine learning models that can predict the winner of the next F1 Grand Prix by using previous championships' data. In order to achieve this aim, the following objectives have been developed:

- Find publicly available historic F1 seasons datasets.
- identification of existing machine learning models for predicting race winners.
- Find the best set of variables inside the dataset.
- Utilize the best possible machine learning models.
- Evaluate the models to find the most accurate one.

1.5 Background Information

As F1 is a regulated motorsport by the Fédération Internationale de l'Automobile (FIA), it is subject to changes over the years, some of which lead to a complete redevelopment of the engines, tires, and shape, dimensions, weight of the car or other changes to improve the safety for the drivers and spectators of the race that could massively impact the teams and their overall performances. The most recent ones happened in 1994, 2009, 2014, and 2022. Looking at these seasons, there may be a correlation with a possible change in winning teams from the previous season. This theory will be later tested during data visualisation and analytics.

2 Literature Review

2.1 Literature Search Methodology

Regarding the different subjects that will be studied during this review and subsequently, in order to complete the project, these different terms will be studied and searched: Data analysis, machine learning to forecast the winner of a race or sporting event, and, of course, F1. related to the field of research will be sourced through key words and, taking into account the vastness of the sources present on the web regarding these topics, such as past research papers published on IEEE, articles and reports on Google scholar and other trusted IT-focused websites, will be used as a reference in a Harvard format in order to better understand and sustain the information in this report. As a process to select related studies from various fields, the main inclusion criteria is that the studies should be linked to F1 and if they are not focused on it, they should at least add value to solve the issue by being related to the prediction of a winner of a race or sports event. Lastly, the studies should only be considered if they are in the language of English; otherwise, they will be excluded from being used for this project.

2.2 F1

As for the main theme of the whole project, the F1 world could still be a mystery to the majority of the population, even if research found that 77 per cent of the growth was driven by the 16 to 35 age group, which as of 2021 accounted for 46 per cent of Formula One's interest pool. (Georgina Yeomans, 2021) This project could be appealing to the new followers of the sport, but for those that are not aware of it, F1 is an extremely complicated sport where until recently there were great inequalities between the team's budget, number of employees, and subsequently to their machine's performances. Understanding the dynamic flow of the sport makes it not only very entertaining as it basically consists of 20 of the fastest and most advanced engines in the world, making it very challenging, but also its structure makes it hard to predict a clear winner for each grand prix as well as for the world driver champion.

Important factors in how the race will play out could be a mix of casual events (flags, incidents, dangerous weather, etc.) and possibilities such as the pilot's position in the grid, the circuit, and so on. All this data makes this simple sport reveal itself quite surprising regarding the 3 V's that can be found and gathered in different types. Some insights here:

- The F1 car has around 300 sensors streaming data back to the garages. (Variety)
- Around 750 billion pieces of data are sent in total from all cars during the race weekend (Volume)
- The raw unstructured data collected for one single car over a race weekend is around 15 GB.
- The peak data transfer (throughput) during the race is about 2 MB/s. (Velocity)
- On average, for every lap, a Formula 1 car produces 35 megabytes of data. (Kiril Varbanov, 2014)

This information is analysed accurately by the teams and used subsequently to improve the machine's performance and the team's score. In fact, each team must meet certain targets in order to gain sponsorship and be able to develop and build machines capable of repeating the circle, potentially allowing the team to achieve higher results in the grid.Sponsors and teams aren't the only ones looking for money in the sport. In fact, as in other popular games, F1 results can be gambled upon, making the possible accurate prediction something not only useful but also interesting.

2.3 Machine Learning

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. (IBM Cloud Education, 2020) They are considered the working horse in the new era of the so-called big data. Techniques based on machine learning have been applied successfully in diverse fields ranging from pattern recognition, computer vision, spacecraft engineering, finance, entertainment, and computational biology to biomedical and medical applications. (El Naqa, I. and Murphy, M.J., 2015) Machine learning differentiates itself into two categories, Supervised machine learning, Unsupervised machine learning. Unsupervised learning is a method to find the pattern of unlabelled dataset which means the dataset have no corresponding output value. In supervised learning, on the other hand, make prediction based on some already known examples or facts. (Kurama V., 2018) Because this study uses a labelled dataset, only supervised learning will be considered. "Regression" and "classification" issues are two types of supervised learning problems. The

major distinction between regression and classification is the label data type. If the label value is continuous, it is a regression problem; if the label value is discrete, it is a classification problem. Only classification methods will be discussed further in this thesis because the purpose of this study is to predict the F1 GP winners, which are discrete values. Machine learning is the main activity in order to create a piece of code able to predict a relevant output starting from a base of data regarding a certain topic. This process is not completely straightforward, as predicting the winner of an F1 GP is never an easy task. But there are always some unique aspects or match conditions that may favour some teams rather than others, such as car settings, driver health and focus, weather conditions, race strategies, etc. As machine learning technology advances, it certainly makes lives easier. However, implementing these algorithms within businesses has also raised an incredible number of ethical concerns surrounding AI technologies and their applications that have continued to thrive over the last few years due to their large digital footprints and the scope of their capabilities.

2.4 Data Analytics

Data analytics is the science of analyzing raw data to make conclusions about that information. Many of the techniques and processes of data analytics have been automated into mechanical processes and algorithms that work over raw data for human consumption. (Frankenfield, J., 2020) Data analysis is important in research because it makes studying data a lot simpler and more accurate. It helps researchers straightforwardly interpret the data so that researchers and businesses don’t leave anything out that could help them derive insights from it. (Anon, n.d.)

The process involved in data analysis involves several different steps. First, find out the data requirements or how the data is grouped and its values may be numerical or be divided by category. Second, collect it from a variety of sources, such as computers, online sources, cameras, environmental sources, or through personnel. Once the data is collected, it must be organised so it can be analysed and finally cleaned up before analysis. There are four distinct forms of data analytics. Descriptive analytics is a type of analytics that describes what has transpired over a specific time period. Diagnostic analytics is primarily concerned with the reasons behind events. Predictive analytics looks ahead to what will most likely occur in the near future. Finally, prescriptive analytics recommends an action plan. Data analytics can then result in a very useful tool to understand better data and subsequently use it for a variety of goals, such as data visualisation and machine learning.

Between the various visual analytic tools that can be used to create visualizations, this research contains visualisations that have been done in the Python programming language using matplotlib libraries in order to maintain the same form throughout the project.

2.5 Data Pre-Processing

This process consists of merging data sources into a single one, filling or deleting gaps in the dataset, removing data that will not be needed across the project, and locating outliers and either explaining or deleting them. Feature engineering is the process of using domain knowledge to extract new variables from raw data that make machine learning algorithms work. (Alteryx Innovation Labs., 2018) while Feature selection is the process of reducing the number of input variables when developing a predictive model. (Brownlee, J., 2019) Data preprocessing is an important step to execute during the development of the study in order to achieve results that matter and avoid problems of overfitting or underfitting, as well as decrease computational cost and maybe improve the model's performance.

2.6 Betting Odds

As one of the most watched sports in the world, Formula One (F1) racing is a huge hit for sports bettors worldwide. (The Sports Geek, n.d.) As a result, the options for how and what to bet on are virtually limitless, with the most well-known betting types being to-win, podium finish, and pole position bets.These aren't the only ones. In fact, it is even possible to bet on them, such as the driver mashup, the prop bets, and the future bets.

- To-Win Bets: The simplest, the driver who is thought to win the race is betted upon.
- Podium Finish Bets: Simplistic as the previous, a driver who is thought that will be present in the podium is betted upon.
- Pole Position Bets: Similar to the first one, the driver who is thought to have obtained the pole position during the qualifying sessions is betted upon.
- Driver Mashup Bets: One driver is chosen between pairs and a bet is placed on the one that is thought to finish the race ahead of the other one in the pair.
- Prop Bets: These are usually divided into skilled and unskilled. The first regards betting on predictions that can be somewhat known with a piece of knowledge in the field, while the ladders are bets on predictions that can't be properly calculated with certain knowledge.
- Future Bets: Bets on who will be the winner of the world driver's championship (WDC) or the winner of the constructor's championship at the end of the starting or

ongoing season.

Betting odds can be written in many formats. Currently, the most common types of odds are fractional, decimal, and American. (Online-Betting.me.uk, n.d.) In the case of this study, the chosen betting odds type will be fractional as it is the most commonly used by the betting companies when concerning F1 bets. A fractional listing would mean that it is possible to win a certain amount for every amount wagered, in addition to the initial amount used to bet. In other words, this is the ratio of the amount of profit won to the initial bet amount.

2.7 Performance Evaluation

Following the successful training of the models, the subsequent phase is to evaluate the model's classification performance using test data. The methods listed below can be used to assess the performance of a ML classification algorithm. In this thesis, classification accuracy, classification reports, and confusion matrices will be utilised to assess the performance of each model created.

The confusion matrix or also known as an error matrix, is a specific table structure that provides visualisation on how the supervised learning model performed on the data. Each row of the matrix represents the instances in an actual class, while each column represents the instances in a predicted class, or vice versa. (Wikipedia, 2019) The error matrix may be used to visualise the number of predicted data misclassifications that ultimately resulted in the incorrect classification.

- True Positives (TP): the model predicted the winner correctly.
- True Negatives (TN): the model predicted the driver as not the winner, and the driver is not the winner.
- False Positives (FP): the model predicted the driver as the winner, but the winner was someone else.
- False Negatives (FN): the model predicted the driver as not the winner but it actually is the winner.

The classification report is a performance evaluation metric in machine learning which is used to show the precision, recall, F1 score, and support score of your trained classification model. (Aman Kharwal, 2021). This evaluation method contains different metrics such as precision, recall, F1 score, and support. Here they are listed with a brief description:

- Precision: The ratio of true positives to the sum of true and false positives
- Recall: The ratio of true positives to the sum of true positives and false negatives

- F1 score: It presents a synthesis of Precision and Recall measures. When Precision equals Recall, it reaches its peak
- Support: is the number of times the class appears in the dataset. It diagnoses the process of performance evaluation and it does not differ between models

2.8 Challenges In The Prediction

Even though ML algorithms have been used to solve many classification problems in recent years, accurately predicting an F1 race remains difficult.There are several instances where a driver driving a mechanically slower car for a team that isn't a championship contender might win. It is because many unexpected things may happen during a race, such as red flags, unpredictable weather conditions, and occasionally pure pure luck, especially when large incidents involve vehicles leading in front of the pack.

2.9 Review Of Literature

Over the past two decades, machine learning (ML) techniques have been increasingly utilised for the purpose of predicting outcomes in sports. (Bunker, R. and Susnjak, T., 2022) This trend has been growing as more people with knowledge of machine learning and artificial intelligence have developed a liking for sports, which has increased their global fans and influence.This converged not only in artefacts like the one shown in this paper, but also in large-scale public and commercial projects developed by senior data analysts and data scientists.One of these massive projects is the one usually viewed during the race weekend events of F1 around the globe. On race day, F1 now uses Amazon’s cloud-based machine learning systems to make sense of all that data for viewers, generating real-time graphics about things like the impact of pit-stops on the race, modelling hundreds of data inputs covering such diverse factors as tyre wear, the weather, track conditions, traffic conditions, and historical data about how fast each team gets through a pit-stop. (John Davidson, 2021) During the development of the project, strict lines of requirements were established and various special factors have been considered thanks to the fact that this artifact’s creators were able to use all kinds of data that’s not usually and completely publicly available. As a result, having the ability to create a model that is more similar to a real-life scenario than a study conducted from a one-man army ad, as shown next in this paper.There’s an old saying in the academic world that data can be made to say anything you like. But Rob Smedley is unequivocal about the results the ‘Fastest Driver’ algorithm has unearthed. (www.formula1.com , 2020) Unfortunately a complete and informational description of how

the model was created is not present, thus making the public rely on the fact that the artefact was created by skilled developers and trusted authorities rather than the result being directly published on the official main website of F1.

In a study named "F1PredictWeb" (villekuosmanen, 2021) the prediction of the position of a certain driver in a F1 GP is calculated and shown through a simple web application. Although the visive aspect of the project is easy to understand and creates a certain value for every hardcore fan of the sport, the actual process of the machine learning model behind it is not only very complex in the amount of data and processes, but it does not even permit a complete understanding of the model itself. Various insights make it understand that previous drivers' performances are taken into account and there is the possibility to generate the prediction with or without a previous prediction of the qualifications on which the subsequent ML operation will operate to execute the new results.

Asterios Stergioudis (2021) provides insight into the significance of data visualisation in the correlation of the various strategies that could be used during a race by the various teams and drivers, thus inquiring how these new data could change the outcome of the race and its winner.This product gives in-depth information that could eventually help betters that prefer to operate on driver mashups and prop bets regarding the sport.

(Grace Wilding, 2020) explores the unpredictability of any race through the seasons by quoting how different aspects such as the weather, crashes, penalties, and mechanical or technical failures together with track and rule changes could impose a threat to the correct prediction of a ML model into the sport's outcome and how these random factors could always impose an aura of uncertainty on the produced outputs.

Research carried out by (Horvat, T. and Job, J., 2020) discusses the review of existing ML algorithms in predicting sports outcomes. The study quotes that over 100 papers were analysed in order to understand in which way these problems are usually solved. The author explains that some sort of feature selection and feature extraction are executed prior to the model with chronologically distributed data. As evaluation, data segmentation and k-cross-evaluation are being used together with neural networks for problems that are mostly categorised as classification problems.

3 Method and Implementation

3.1 Design and Methods

In this section, it will be explained which methodology has been used and why it was chosen, as well as other points of focus regarding the project, such as the limitations and options of the various themes that make it, as well as their design specifications or user requirements. A concept solution will be developed and subsequently tested to find the best fit methodology for the project. Once chosen, the one that fits all the logical and practical requirements, will be the basis for the design and development of the artefacts that will be successively tested extensively in order to confirm which is undoubtedly the best possible model. At the end, the whole process will be summarised and conclusions will be drawn from it.

3.2 Methodology

A project's methodology refers to a technique or set of procedures and resources utilised to develop and execute the project from conception to completion. When it is used in the creation of software, it is referred to as a software development methodology. There are three types of software development methodologies: classic, agile, and hybrid. Traditional techniques, such as the waterfall model, are recognised for being formal and very hard to change, whereas agile methodologies differ only in the fact that they are informal. Lastly, Hybrid methodologies combine the two previous types of methodologies into a single technique. Early-stage planning is required by the waterfall technique to discover any defects before the development stage. This model's name derives from its structural arrangement. Each step follows the completion of the preceding one, with the output of the previous phase becoming the input of the next. While in the specific phases of each step of the plan, test driven development (TDD) could be applied as needed to ensure the correct progress of the project. Test-driven development (TDD) is a software development process relying on software requirements being converted to test cases before software is fully developed and tracking all software development by repeatedly testing the software against all test cases. This is as opposed to software being developed first and test cases being created later. (Wikipedia, 2022) The waterfall methodology follows a chronological process and works based on fixed dates, requirements, and outcomes. With this method, the individual execution teams aren’t required to be in constant communication and, unless

specific integrations are required, are usually self-contained.(www.workfront.com.,n.d.) This methodology would be the preferred one as the project will only be able to be completed successfully by following a certain order, starting from the collection of data, cleaning of the entire dataset, and subsequent analysis in order to be fitted for the machine learning models and undergo proper tuning as needed. This method is straightforward, well-defined and will permit planning for the duration of the project and for each stage of its development. On the other hand, throughout the software development process, the Pandas' 'info', 'head', and'shape' methods were frequently utilised to guarantee that the code generated what the author intended it to do in the first place, thus granting a successful passage to the next step of the plan. It could be possible to say the whole project has then adopted a hybrid methodology.

3.3 Limitations and Options

Having a look into the various steps of the creation of the artefact, there could be different limitations and options for each phase that could considerably jeopardise the project. Starting from data collection, it is important to choose the best way to execute the action through a method that will give real and unchangeable data, such as records and documents that are easy to collect and do not require any costs, such as interviews, surveys, questionnaires, focus groups, or observations. It is important to notice that even if the method presents itself as fast, it could require some time to review all the datasets that have to obviously be publicly available other than being possibly complete and without any missing data. Thus, the data collected requires a cleaning phase to sort out any possible incomplete values or fix them accordingly. While analysing the data can be done in a variety of ways, it’s important to choose a method that would not only make sense and be relevant but even useful to a possible business or choice. The major limitation of this phase is the fact that it could require not only a decent sized amount of data but even a decent amount of time. Thus, a lack of commitment could flaw the entire process, which needs quite some patience. Finally, the data used needs to be accurate as it could introduce a certain bias in the subsequent use of it. As complete perfection of the dataset could be unachievable, corrections and statements need to be put together to inform of any imperfection of the output produced or given input. Another aspect to look at should be the privacy concerns that need to be addressed. Luckily, if the dataset is public, this shouldn't pose a problem. While creating the model through machine learning, it is important to take into account ethics, any deterministic problems, as well as any missapplication or interpretability concerns that need to be addressed while building the artefact.

3.4 Design Specification / User Requirements

To build the project, certain requirements will be necessary, primarily a good amount of knowledge about the IT and data fields, especially in machine learning and artificial intelligence, knowing how to programme and visualise graphs in the Python language and a clear understanding of how each model works and their possible tweakable parameters in order to execute a complete testing and evaluation of the artifact. The environment used is Anaconda Spyder, which allows you to easily see the result, errors, and the code without excessive scrolling and permits you to save the whole project as a "py" file. A variety of Python's own standard libraries have been utilised in this project, but even many other third-party libraries for solving numerous domain-driven problems have been utilised to finalise the artifact. Some of these included pandas for data processing and analysis of data-frames, mathplotlib, and seaborn for data visualisation and plotting figures. NumPy for scientific computing, Scikitlearn was chosen as the machine learning library because it has a large variety of machine learning methods. Finally, some understanding of the bets and how they work is suggested in order to grasp the value that the product could bring to the process.

3.5 Concept Solution

As a solution for the creation, this simple diagram has been created and shown as Figure 3.1.

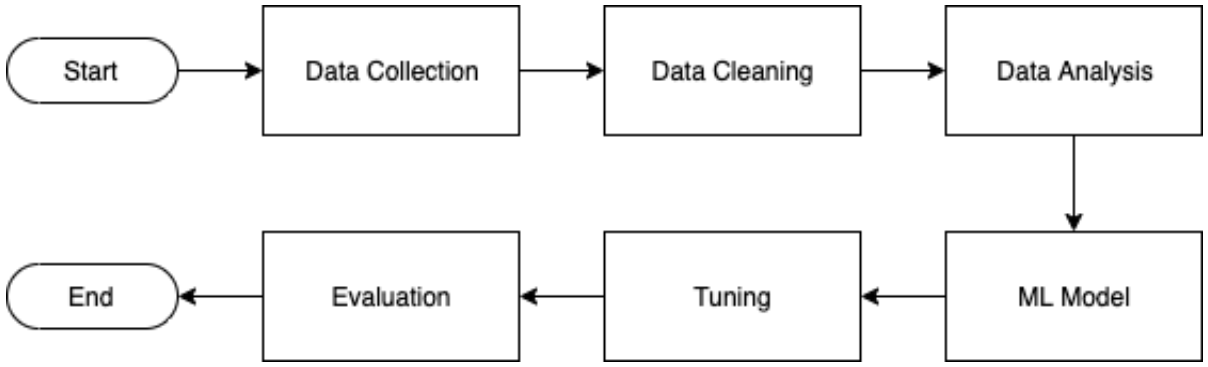

Figure 3.1: Waterfall diagram of the project's development.

Another simple diagram to show the life cycle TTD of each step of the development process has been created and is shown in Figure 3.2.

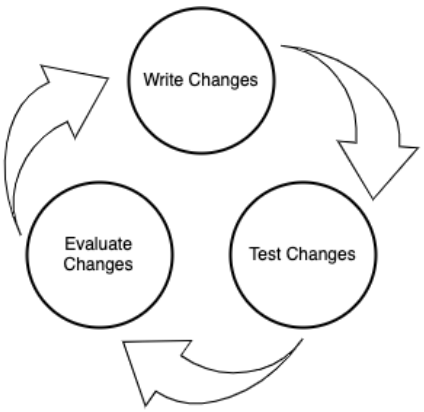

Figure 3.2: TTD diagram of the Agile method used for each development phase.

3.6 Dataset Operations

The main data-set used in this project was downloaded from the Kaggle dataset ‘Formula 1 World Championship (1950-2022)’ (Vopani, Kaggle.com, 2022). This data set was collected and refined mainly from two sources; the first is the official F1 website that contains historical data since 1950 regarding drivers, circuits, constructors, and their results, the second is from the ergast API, which provides an historical record of motor racing data. This second source gives more in-depth details regarding results, times, drivers, etc. These two will mostly contain the same data, but by having a dataset composed of both, great accuracy and completeness will be acquired.

Since the data is received in various CSV files containing multiple data frames, these will be required to be read and merged using common keys into one large dataframe on which all the next operations will take place. In order to make the data more user-friendly, most of the column names should be changed. These operations, together with the dropping of columns that won't be used during the development of the paper, would constitute the so-called data wrangling steps. Then it is possible to move on to the feature engineering that is showcased in the creation of new variables that could prove useful during the modelling process. Firstly the datetime function of Pandas is used to convert date stings to a data format on which it is possible to operate mathematical operations. As in the artefact, the date of the race and the date of birth of each driver are used to calculate each driver’s age during each race. Then each row that presents an age quite impossible for a driver to be on the grid is dropped. In this case, 50 is chosen as to limit the number of outliners inside this new feature. Checking the data regularly helped with the correct development of the cleaning process. Some checks revealed some numeric data, such as the fast_lap_rank, to be a different type, thus

the need for the to_numeric panda function in order to convert the values of the column into numeric values. Further observations reveal the presence of missing or null data that needs to be sorted as well, especially in the qualify_position and again in the fast_lap_rank columns. A fillna approach has been chosen in order to fill the missing values with the number 20, as it's the last spot on the grid in recent years and historically a place where drivers in the past could not reach the end of the race most of the time. This approach has also been taken in order to prevent the value 0 from interfering with the numeric nature of the columns itself where the number close to 0, in this case 1, has the best probability of winning the race when disregarding any possible influence of other values. This same approach has been taken when dealing with the 0s present in the start_position column, thus maintaining the influence on the model of the famous pole position. The latest addition on the column axis is the overtake feature that has been created by subtracting the finish_position from the start_position, giving its values a positive impact if more than 1, and a negative impact if it contains a negative numeric value. As a last editing touch, the status column has been simplified with a mapping that makes the column express if the driver has finished a race with a 1 and if not, with a 0. Some other id columns have been dropped.

3.7 Design and Development

All the processes will follow the same passages of the data cleaning and be shown through the chosen IDE's terminal. This will consist of printing missing data and other error-finding codes on the cleaned dataset to make sure it's been completely fitted properly. The data frame has been iterated to get information about all the drivers' results, including features such as the grid and finishing position of each driver, their teams, and other less relevant variables such as date of birth, nationality, and finishing status, which could be later reviewed to check whether there could be some correlations. Some of these relevant questions could be if there is indeed a correlation between the age of the drivers and their performance, if racing in their home country could have any psychological impact, or if some drivers are more prone to crashes than others. To gain more information from the data, it's possible to create a lookup function to shift the points from previous races within the same championship for each driver and know the standings in the WDC of each competition. This process can even be applied to the teams in order to better understand the various constructors' championships. Other data could be relevant to the final result, such as the qualifications and the weather. That could be found online, but it would surely require other extensive types of collection, such as scraping or dictionaries, that won't be covered in this paper. Eventually, some variables could be dummified, such as the circuit, nationality, and

team variables, then dropping the features that will not be significantly present. In the case of this study, a label encoding method has been chosen in order to convert the labels into a numeric form from their categorical values so as to convert them into a more machine-readable friendly form. This is an important preprocessing step for the structured dataset in supervised learning. In an effort to do so, categorical and numerical features have been divided and the label encoding has been applied. As a last check before initiating the creation of the machine learning model, the skewness of all the remaining features has been printed and, apart from qualify_position and fast_lap_rank, that briefly surpass the threshold limit of 1/-1, all the other values seem completely acceptable, so all the columns will be kept for the modelling process.

3.8 Machine Learning Models

The machine learning models will be executed with the proper features and logic to reach an acceptable percentage of accuracy in prediction and then confronted with the original data. Since the purpose is to predict the first place on the podium for each race, it is possible to treat the target variable as either a regression or a classification. When evaluating the precision score of a regression, it's possible to predict the result of each race of a championship and calculate the percentage of predictions that are the same as the actual results. In the case of a classification, the model could wrongly output more than a winner if the accuracy is too low. In this case, a specification where only the driver with a higher probability will result in a winner should be implemented. As for the train and test split, it is possible to utilise only the last championship's races for the testing of the various algorithms such as logistic and linear regressions, random forests, support vector machines, and neural networks for both regression and classification problems, and subsequently test them on the test dataframe. In regards to this study, only the classification with Logistic Regression, Random Forest Classifier, and Decision Tree Classifier accompanied by a standard and a minmax scaler with some hyperparameter tuning will be applied without the necessity of a probability column as the models are able to achieve a high standard of accuracy.

Firstly, the split for the train occurs where a train dataframe is created with all the values of the previous season minus the one that's going to be predicted. Here, X_train will contain the previous cited data without the driver and the finish_position columns, while y_train will contain only the finish_position column of the train DF. Here, in order to train and test swiftly, all the models are first executed without any tuning. The models adapted would first be standardised by the standard scaler and the minmax scaler that are fitted on the training set with pandas. The ladder is later dropped as even without any tuning, it produced very

disappointing results. Once the model is fitted, the predition takes place on the test set and results start to emerge. Other columns are added to the prediction DF, which is then merged with a second DF to effectively understand the data and who won the race, and is then transformed into an CSV file on the machine. at the very end of the process, evaluations are made and it’s given a score as well as other metrics to evaluate the model itself. Aside from data splitting, another thing to consider while looking for the optimum method is parameter value selection, also known as hyperparameter optimization, in which every algorithm has different hyperparameters. Now, since it may be possible to achieve better results with some hyperparameter tuning, the DTC and LR are being chosen as they seem to obtain score results in a better and more stable range. On the first one, a series of simple loops were employed to find the best suitable parameters for the max_leaf_nodes, which has been revealed to be 1610, and a value of 17 for the max_depth in order to achieve a better validation accuracy for the model. This process can be a time-consuming operation, especially if each method has several hyperparameters, so it's advisable to use an algorithm like grid-search to identify the optimum hyperparameter combination automatically. The Grid Search algorithm may be developed using the scikit-learn package by importing the GridSearchCV class. As for the second model, the LR, a grid search has been done using a Repeated Stratified KFold that required a reasonable amount of time to finish its matching even though some combinations were omitted to cut back on the warnings and errors. Once they inserted the up-to-date parameters on each model, their score increased visibly.

4 Evaluation

This chapter shows the prediction results using the chosen algorithms. in order to see how different models behave and ultimately choose the best performing one, and that it is not affected by bias such as overfitting or underfitting, and thus could be reliable for the aim of the project. This step will use the same evaluation metrics on the two possible problems that have been analysed during the development: a multiclass classification for the positions of each driver on the grid as well as a binary classification in order to predict the winner of the race with purely 1s and 0s.

4.1 Evaluation Methodology

To evaluate the results of each model at its simplest, the precision_score function has been used, and looking at the higher general results, the two highest models that gave the best range of values have been chosen to implement more changes and hyperparameter tuning as well as checks on the reliability of the subsequent models that they will serve to generate. In regards to the untouched models, they will continue inside the loop of calculations and printing on the IDE console but won't be further assessed if not as a curiosity. Two scalar methods were used, and even in this case, only the one with models with better scores will be analysed further. Once done with tuning and evalation methods, the remaining models will be the ones taken most into consideration for a possible test of the models on other seasons of the F1 WDC in order to consolidate that the models are constant in the higher scores and accuracy of prediction.

4.1.1 Evaluation Metrics

The metrics that were being used to access this study are the classification report, the confusion matrix, and a model accuracy test to make sure the algorithm is not biased, such as plotting the model accuracy for each tree depth of the decision tree classifier model to check if it is overfitted or not, as shown in Figure 4.1, and even the ability of the model to beat standard betting odds.

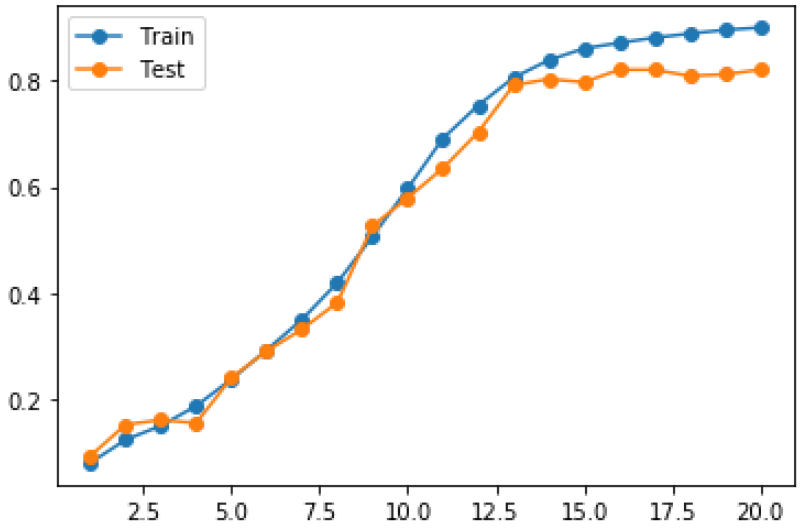

Figure 4.1(Plot of the increasing depth to check if the DTC model is overfitted.)

These various performance metrics include accuracy, F1 score, precision, recall, and support; while most of these metrics are included in the report, they can also be calculated using the decision matrix.This metric is also known as an error matrix in machine learning, and it's intended specifically for classification issues since it displays how models behave when they get confused when making predictions. When in regards to beating the odds, a simple array of betting odds will be chosen as, unfortunately, betting sites do not have any accessible database where previous season's odds could be extracted. As such is the case, random odds are generated on which a sane amount of money is supposedly staked, and by using the accuracy of the models, it is possible to have an insight on the question of whether it is really possible to obtain a profit by betting on the predictions of the model itself.

4.1.2 Baseline systems

Apart from the model presented in this paper, another approach was taken into consideration and brought forward. This alternative proposed consists of the conversion of the categorical values into binary features instead of using the label encoding as recorded in the paper. This variant was acceptable, but it had several flaws, including the fact that not only did the dataframe grow significantly in size, from 15 features to over 200, but that some of the new columns were circuits, driver nationality, and constructors specific to each possibility, making some of them very skewed.On top of these facts, the large quantity of values has shown a slower development time as well as longer times for saving and loading the CSV dataframes.

4.1.3 Dataset

In the case of the confusion matrix, it is possible to obtain the same evaluation metrics on each of the various models that are the same, so they can be easily confronted and conclusions can be drawn upon them in a fairly easy and straightforward way.

4.2 Results

Even from the results of the early created models, the accuracy was already decent and later on with the hyperparameter tuning of the two best models, in this case the logistic regression and the decision tree classifier, the accuracy increased by nearly 10%. Unfortunately for the random forest classifier, it did not show great results on the multiclass classification even if completely optimised for the binary classification, thus the need to cast it aside to concentrate on the other models, as even with tuning, the accuracy score for the multiclass was already lower than the others. As follows, in Figure 4.2, are the results of the models with the binary classification with the standard scaler, as the min max scaler did not live up to the expectations, thus models with it will not be analysed further.

```
Scores of every model by: StandardScaler()
The score of the model is:
82.0
The accuracy of the model is:
98.0
[[320   3]
 [  3  14]]
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       323
           1       0.82      0.82      0.82        17

    accuracy                           0.98       340
   macro avg       0.91      0.91      0.91       340
weighted avg       0.98      0.98      0.98       340

The score of the model is:
88.0
The accuracy of the model is:
99.0
[[321   2]
 [  2  15]]
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       323
           1       0.88      0.88      0.88        17

    accuracy                           0.99       340
   macro avg       0.94      0.94      0.94       340
weighted avg       0.99      0.99      0.99       340

The score of the model is:
100.0
The accuracy of the model is:
100.0
[[323   0]
 [  0  17]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       323
           1       1.00      1.00      1.00        17

    accuracy                           1.00       340
   macro avg       1.00      1.00      1.00       340
weighted avg       1.00      1.00      1.00       340
```

Figure 4.2 (Model results with binary classification using the standard scaler)

As it is quite easy to see, the results of these models are quite surprising. While the accuracy of the prediction is always higher than the score, apart from the logistic regression that achieved an unbelievable 100%, the models seem quite promising in their objective and their range of precision, reacall, support, and f1 score are all in an acceptable range. This could be because the training set is quite larger than the test itself and it could've improved the overall performance.

Regarding the multi class classification results, these are shown with a confusion matrix as well in Figure 4.3 here below.

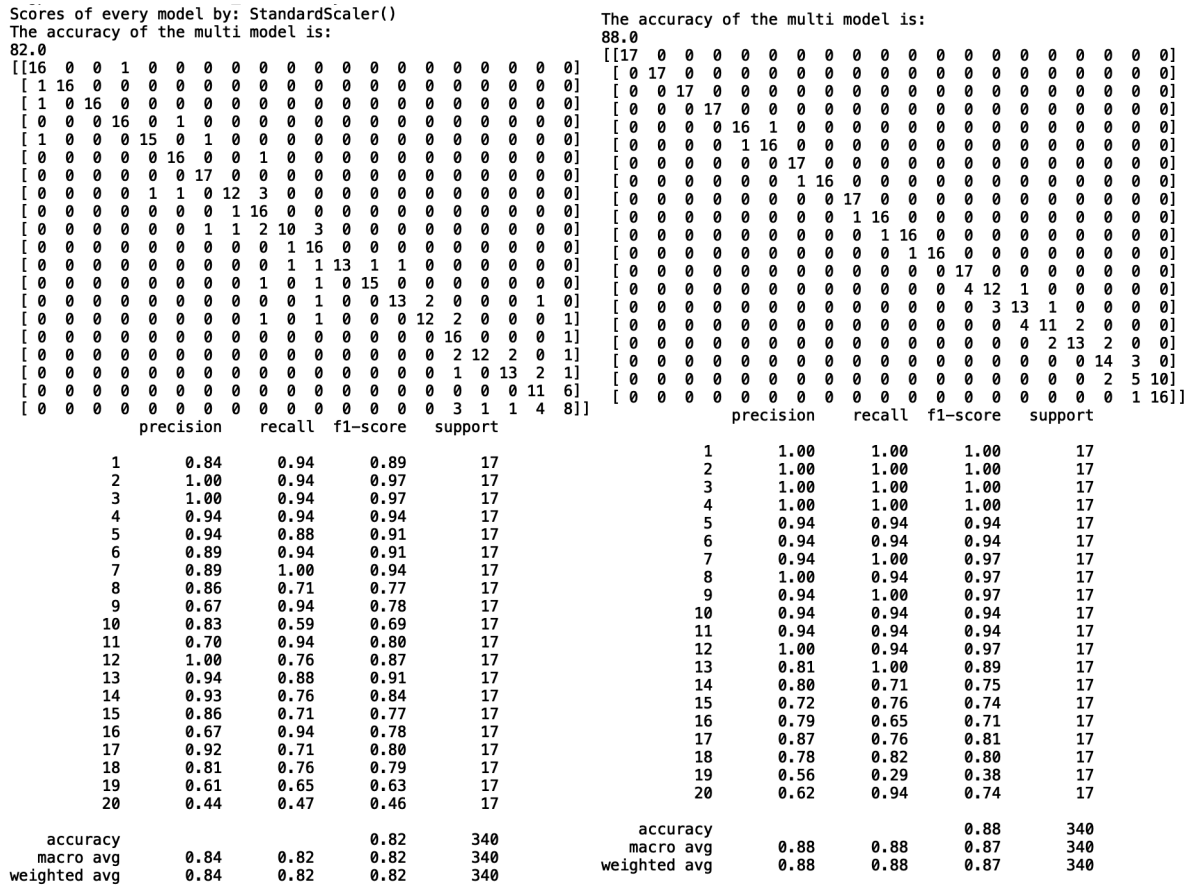

Figure 4.3 (LR(right) and DTC(left) confusion matrix for multi class classification.)

Even in this case, the LR and DTC had reached good precision and accuracy but not completely perfect. This means that the models are unfortunately not smart enough to put a single value unique for each driver, as there happen to be more drivers that are in the

number 1 position. This requires the development of the binsary classification in order to sort out the possibility of noise and have a clear answer for the prediction and the betting as well. Here is Table 4.1 with the simple betting odds with the same precision as the prediction model.

Round	Predicted	Odds	Winnings
1	Right	2	10
2	Right	2	20
3	Right	2	30
4	Right	2	40
5	Right	2	50
6	Right	2	60
7	Right	2	70
8	Right	2	80
9	Right	2	90
10	Right	2	100
11	Wrong	2	90
12	Right	2	100
13	Wrong	2	90
14	Right	2	100
15	Right	2	110
16	Right	2	120
17	Right	2	130

Table 4.1 (Betting odds with ML model.)

This table shows the chances of beating the odds on an entire season of F1 with bets that are quite simple, but shows that the models are capable of correctly predicting the winner of the race and even gaining a profit of 130 pounds while spending 10 pounds on each bet and having the chance of winning the double of the initial bet. As simple as the betting process is, there is a good chance that a model with such high accuracy could be useful even with higher betting odds, because the algorithm is capable of predicting the majority of cases, and it would be difficult to lose money if the betted amount is always the same and the model correctly predicts at least 50% of the bettings.

4.3 Discussion

Results show that the LR and DTC performed the best across all tests, with DTC not far behind, illustrating that although the method is more complex when compared to LR, it could not really perform better than the simple LR. Taking this into consideration, the DTC clearly outperformed the RTC across all the tests carried out. Both models' results reveal that LR and DTC may surely be used to classify the winner of a Formula 1 grand prix, provided the models have been optimised and both have been appropriately trained, like in this study where the test set was always run as one season, keeping the number of rows under 500 while the training set was composed of all the other rows.

The performance of these models is now hard to properly compare with many other papers as not many have used the same data that will eventually become obsolete soon as every weekend race adds new data to the Formula 1 and ergast databases. Luckily, on the same racing term, it was possible to find other models of racing, such as, for example, the study of Anandaram Ganapathi, where similar data study machine learning models achieved an average accuracy of nearly 100% (±5%) while using the same dataset applied in this paper.

5 Conclusions

This paper presents an empirical analysis of the predition on the F1 GP winner. To address the challenge, we gathered all pertinent historical racing data from 1950 to 2020 and created a big dataset. To conduct the analysis and predict the winner of the GP, various branches of data science have been converged, including preprocessing of data, visualisations of data, preparation of data, feature selection, and implementing different machine learning models for the predictions. The dataset was preprocessed to ensure consistency by eliminating missing values and encoding variables into numerical representation. The best attributes were chosen by using a heatmap to visualise skewness and correlation. Several machine learning models were used on chosen features to predict the winner, and the results were spectacular. Three alternative strategies for making predictions were investigated and compared: LR, DTC, and RFC with standard scaler and minmax scaler. The taken approach, according to the analysis, aids the model in making consistent predictions, while also improving the model's betting performance. First of all, the LR model was applied, which predicted the GP winner with good, astonishing accuracy. To improve the performance of the model, the parameters were fine-tuned with the gridCV, while for the DTC model loops were used and good results were achieved. These 2 models' performances were enhanced confortably by 80%–100% depending on the season and randomness.

6 Recommendations for future work

Different new experiments using various models can be operated in the future, or modifying existing models and comparing them can also provide meaningful data to work on. Despite the trained models showing quite valid results, there is still room for improvement, whether that's adding new meaningful data to the models or training them for longer times, if not creating a new model altogether. An increase in the amount of data available and relevant, such as pitstop times, tyre degradation on the machines, driver's reaction time at the start of the race or at speedtraps on the track, and other driver's parameters could help to improve the model's accuracy and relevance to the new regulations and changes. If more data or models cannot be processed, it's still possible to increase the utility of the project by creating a webapp to permit public access to the model. Acquiring new data is an underutilised approach for improving the training of machine learning models. P. Norvig, the Director of Research at Google, once said: 'We don't have better algorithms." We just have more data' (Kdnuggets.com, 2019).

7 References

www.mercedesamgf1.com. (n.d.). Mercedes-AMG Petronas Formula One Team. [online] Available at: https://www.mercedesamgf1.com/en/.

inews.co.uk. (2022). From the safety car to bigger wheels, F1 rule changes for 2022 explained. [online] Available at: https://inews.co.uk/sport/formula-one/f1-rules-2022-safety-car-bigger-wheels-formula-one-regulation-changes-explained-1519838.

Kiril Varbanov (2014). F1 Framework: F1 and Big Data. [online] F1 Framework. Available at: http://f1framework.blogspot.com/2014/03/f1-and-big-data.html.

Kurama, V. (2018). Unsupervised Learning with Python. https://towardsdatascience.com/unsupervised-learning-with-python-173c51dc7f03. [Online].

Australian Financial Review. (2021). How Formula 1 uses machine learning to spice up races. [online] Available at: https://www.afr.com/technology/how-formula-1-uses-machine-learning-to-spice-up-races-20210716-p58abx#:~:text=On%20race%20day%2C%20F1%20now.

www.formula1.com. (n.d.). Hamilton? Schumacher? Senna? Machine learning reveals the fastest F1 driver of the past 40 years | Formula 1®. [online] Available at: https://www.formula1.com/en/latest/article.hamilton-schumacher-senna-machine-learning-reveals-the-fastest-f1-driver-of.3DwwPLW4glCmlunjciH1Cz.html.

Online-Betting.me.uk (n.d.). Understanding betting odds to beat them. https://www.online-betting.me.uk/articles/betting-odds-explained.php.

Wikipedia Contributors (2019). Confusion matrix. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Confusion_matrix.

Bunker, R. and Susnjak, T. (2022). The Application of Machine Learning Techniques for Predicting Match Results in Team Sport: A Review. Journal of Artificial Intelligence Research, 73, pp.1285–1322. doi:10.1613/jair.1.13509.

villekuosmanen.github.io. (n.d.). F1Predict - Formula 1 Grand Prix predictions. [online] Available at: https://villekuosmanen.github.io/F1PredictWeb/?repost#/race.

www.f1-predictor.com. (n.d.). F1 predictor – Machine-learning based F1 race prediction engine. [online] Available at: https://www.f1-predictor.com/.

Capgemini UK.(2020). The unreliability of machine learning in Formula 1.[online]Available at: https://www.capgemini.com/gb-en/2020/09/the-unreliability-of-machine-learning-in-formula-1/

Horvat, T. and Job, J. (2020). The use of machine learning in sport outcome prediction: A review. WIREs Data Mining and Knowledge Discovery, 10(5). doi:10.1002/widm.1380.

Wikipedia Contributors (2018). Test-driven development. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Test-driven_development.

www.workfront.com. (n.d.). Waterfall Methodology - A Complete Guide | Workfront. [online] Available at: https://www.workfront.com/project-management/methodologies/waterfall.

Stobierski, T. (2021). Data Wrangling: What It Is & Why It's Important. [online] Business Insights - Blog. Available at: https://online.hbs.edu/blog/post/data-wrangling.

Data Science | Machine Learning | Python | C++ | Coding | Programming | JavaScript. (2021). Classification Report in Machine Learning. [online] Available at: https ://thecleverprogrammer.com/2021/07/07/classification-report-in-machine-learning/#:~:text= Classification%20Report%20using%20Python&text=It%20is%20a%20performance%20eva luation

Kdnuggets.com. (2019). In Machine Learning, What is Better: More Data or better Algorithms. [online] Available at: https://www.kdnuggets.com/2015/06/machine-learnin g-more-databetter-algorithms.html

8 Bibliography

Singh, C. (n.d.). 3 biggest regulation changes in F1 history. [online] www.sportskeeda.com. Available at: https://www.sportskeeda.com/f1/3-biggest-regulation-changes-f1-history.

Chandra, V. (n.d.). Comparison between Various Software Development Methodologies. International Journal of Computer Applications, [online] 131(9), pp.7–10. Available at: https://www.academia.edu/29931163/Comparison_between_Various_Software_Development_Methodologies?auto=citations&from=cover_page.

www.altair.com. (n.d.). What is data wrangling and what are the steps? - Altair. [online] Available at: https://www.altair.com/what-is-data-wrangling/#:~ :text=Data%20wrangling%20is%20the%20process.

kaggle.com. (n.d.). F1 Champ | EDA | Classification 100% accuracy. [online] Available at: https://www.kaggle.com/code/anandaramg/f1-champ-eda-classification-100-accuracy/notebook

9 Appendices

Figure 9.1 (Project Gantt chart)

```
#!/usr/bin/env python3

# -*- coding: utf-8 -*-

"""

@author: ID 19150299

"""

#importing libraries

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt
```

```
import numpy as np
import plotly.graph_objects as go
from plotly.offline import iplot
from datetime import datetime
import math as ma
from sklearn.model_selection import train_test_split
import warnings
warnings.simplefilter("ignore")
#importing other libraries for ML
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler,MinMaxScaler
from sklearn.metrics import accuracy_score,precision_score,confusion_matrix,classification_report,roc_auc_score, roc_curve
from sklearn import tree
from sklearn.tree import plot_tree, export_text
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import GridSearchCV
def classification_operations(model,df,scaler,num):
    #values for evaluation
    score = 0
    accuracy = 0
```

```
con_mat = 0
accuracy_multi = 0
con_mat_multi = 0
pred_df2 = pd.DataFrame()
for gp in df[df.season == 2020]['round'].unique():
    #split test
    test = df[(df.season == 2020) & (df['round'] == gp)]
    X_test = test.drop(['finish_position','driver'], axis = 1)
    y_test = test.finish_position
    X_test = pd.DataFrame(scaler.transform(X_test), columns = X_test.columns)

    #make predictions
    pred_df = pd.DataFrame(model.predict(X_test), columns = ['pred_finish'])
    pred_df['act_finish'] = y_test.reset_index(drop = True)
    pred_df['actual'] = pred_df.act_finish.map(lambda x: 1 if x == 1 else 0)
    pred_df.sort_values('pred_finish', ascending = True, inplace = True)
    pred_df.reset_index(inplace = True, drop = True)
    pred_df['predicted'] = pred_df.index
    pred_df['predicted'] = pred_df.predicted.map(lambda x: 1 if x == 0 else 0)

    #add useful columns to predictions
    pred_df['round'] = gp
    pred_df['season'] = 2020
    inversed = pd.DataFrame(scaler.inverse_transform(X_test))
    pred_df['start_position'] = round(inversed[0])
    pred_df2 = pred_df2.append(pred_df, ignore_index = True)
```

```
    """print(prediction_df.head())"""
    if num < 3:
        #evaluation for multiclass
        accuracy_multi += accuracy_score(pred_df.act_finish, pred_df.pred_finish)
        con_mat_multi += confusion_matrix(pred_df.act_finish, pred_df.pred_finish)

    #evaluation for binary
    score += precision_score(pred_df.actual, pred_df.predicted)
    accuracy += accuracy_score(pred_df.actual, pred_df.predicted)
    con_mat += confusion_matrix(pred_df.actual, pred_df.predicted)

    df_pred = df[(df.season == 2020) & (df['round'] == gp)]
    df_pred2 = df_pred[['driver','round','start_position']].copy()
    df_pred3 = pd.merge(df_pred2, pred_df,on = ['round','start_position'])

pred_df2.to_csv('pred_df.csv')
df_pred3.to_csv('prediction_df.csv')

#evaluation for multi
if num < 3:
    report_multi = classification_report(pred_df2.act_finish, pred_df2.pred_finish)
    accuracy_multi = accuracy_multi / df[df.season == 2020]['round'].unique().max()
    print('The accuracy of the multi model is: ')
    print(round(accuracy_multi*100))
    print(con_mat_multi)
```

```
        print(report_multi)



    #evaluation for binary
    report = classification_report(pred_df2.actual, pred_df2.predicted)
    model_score = score / df[df.season == 2020]['round'].unique().max()
    accuracy = accuracy / df[df.season == 2020]['round'].unique().max()
    print('The score of the model is: ')
    print(round(model_score*100))
    print('The accuracy of the model is: ')
    print(round(accuracy*100))
    print(con_mat)
    print(report)
def main():

    #getting data from csv into data frames
    status_df = pd.read_csv('Data/status.csv')
    qualifying_df = pd.read_csv('Data/qualifying.csv')
    drivers_df = pd.read_csv('Data/drivers.csv')
    driver_standings_df = pd.read_csv('Data/driver_standings.csv')
    races_df = pd.read_csv('Data/races.csv')
    results_df = pd.read_csv('Data/results.csv')
    constructor_standings_df = pd.read_csv('Data/constructor_standings.csv')
    constructors_df = pd.read_csv('Data/constructors.csv')

    #print all data frame columns
```

```
pd.set_option('display.max_columns', None)


#merging all data frames into one
df1 = pd.merge(results_df, races_df, how='inner', on ='raceId').drop(['url','time_y',
        'time_x','fastestLap','circuitId','number','position','positionText',
        'fastestLapTime','fastestLapSpeed','points','milliseconds'], axis = 1)
                df2 = pd.merge(df1, drivers_df, on = 'driverId').drop(['number','code','forename','surname','url'],1)
            df3 = pd.merge(df2, driver_standings_df, on = ['driverId', 'raceId']).drop(['positionText','position'],1)
                df4 = pd.merge(df3, constructors_df, on = ['constructorId']).drop(['constructorRef','url','nationality_y'],1)
        df5 = pd.merge(df4, constructor_standings_df, on = ['constructorId', 'raceId']).drop(['positionText'],1)
            df6 = pd.merge(df5, qualifying_df, how='left', on = ['driverId', 'raceId']).drop(['q1','q2','q3','number',
                                                    'constructorId_y'], axis = 1)
final_df = pd.merge(df6, status_df, on ='statusId')
final_df = final_df.drop(['resultId','driverId','driverStandingsId','constructorStandingsId',
                'qualifyId','constructorId_x'],1)
print(final_df.columns)


# changing columns labels to understand data easier
col_lab2 = {'grid':'start_position','positionOrder':'finish_position','rank':'fast_lap_rank',
        'year':'season','name_x':'gp','driverRef':'driver','nationality_x':'driver_nationality',

'points_x':'points_driver_season','wins_x':'driver_season_wins','name_y':'constructor',
        'points_y':'constructor_season_points','position_x':'constructor_season_position',
```

```
        'wins_y':'constructor_season_wins','position_y':'qualify_position'}
final_df.rename(columns=col_lab2,inplace=True)

#check data frame info
print(final_df.info())
print(final_df.shape)
print(final_df.columns)

#change string to date format
pd.to_datetime(final_df.date)
final_df['dob'] = pd.to_datetime(final_df['dob'])
final_df['date'] = pd.to_datetime(final_df['date'])

#get driver's age at every race and insert it in a new column then adjust data frame
difference = final_df['date']-final_df['dob']
age = difference.dt.days/365
final_df['age'] = round(age)
final_df = final_df.drop(['dob','date'], axis = 1)

#drop drivers with an unrealistic age to race in the date of the GP
final_df.drop(final_df[final_df['age'] >= 60].index, inplace = True)

#convert fast_lap_rank into numeric as it's not a string
final_df['fast_lap_rank'] = pd.to_numeric(final_df['fast_lap_rank'],errors='coerce')

#looking for null data and in %
```

```
print(final_df.isnull().sum())
print(final_df.isnull().sum() / len(final_df) * 100)

#fill null values of fast lap rank and qualify_position with 20s as it's the last position on the grid
final_df['qualify_position'] = final_df['qualify_position'].fillna(20)
final_df['fast_lap_rank'] = final_df['fast_lap_rank'].fillna(20)

#Change where start_position is 0 to finish_position to improve overtakes performance in model
final_df['start_position'] = np.where(final_df['start_position'] == 0, final_df['finish_position'], final_df['start_position'])

#adding new column overtake to improve useful data
overtakes = final_df['start_position'] - final_df['finish_position']
final_df['overtakes'] = overtakes

#mapping status column if finished or else
final_df.status = final_df.status.map(lambda x: 1 if x == 'Finished' else 0)

#check data frame info
print(final_df.info())
print(final_df.shape)
print(final_df.columns)
print(final_df.isnull().sum())
print(final_df.isnull().values.any())
print(final_df.skew())
```

```
#drop some columns
final_df = final_df.drop(['raceId','statusId'],1)
#DATA VISUALISATION
#nationality of winning drivers
fig = go.Figure(data=[go.Pie(labels=final_df[(final_df['finish_position']==1)].sort_values(by=['driver_nationality'])['driver_nationality'].unique(),
values=final_df[(final_df['finish_position']==1)].groupby('driver_nationality')['finish_position'].value_counts(),
hole=.3)])
fig.show(renderer="svg")
#show heatmap of data
plt.figure(figsize=(10,8))
sns.heatmap(final_df.corr(),annot=True, linewidths=2, linecolor='yellow',cmap="YlGnBu")
plt.show()

#drop some more columns
final_df = final_df.drop(['points_driver_season','driver_season_wins','constructor_season_wins','constructor_season_points'],1)
print(final_df.shape)

#seperating categorical and numerical columns for understading
num_data = []
cat_data = []
for data in final_df.columns:
    if final_df[data].dtypes == 'O':
        cat_data.append(data)
```

```
    else:
        num_data.append(data)

#label encoding categorical columns to use them into the ml model
print(final_df.info())
encoder = LabelEncoder()
for i in cat_data:
    if i != 'driver':
        final_df[i] = encoder.fit_transform(final_df[i])
print(final_df.info())

print(final_df.skew())
print(num_data)
print(cat_data)

#MACHINE LEARNING
num = 0

#split train
train = final_df[final_df.season <2020]
X_train = train.drop(['finish_position','driver'], axis = 1)
y_train = train.finish_position

#ml models list
log_reg = LogisticRegression(solver = 'saga', penalty = 'l1', C = 100)
random_for_class = RandomForestClassifier()
```

```
                    dec_tree_class = DecisionTreeClassifier(max_leaf_nodes=1605,
max_depth=17,criterion='gini',random_state=42)
    model_list = [dec_tree_class,random_for_class,log_reg]


    """
    #code for cheking overfitting and evaluation models
    scaler = StandardScaler()
    X_train = pd.DataFrame(scaler.fit_transform(X_train), columns = X_train.columns)

    #search best hyperparameteres for logistic regression
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
    param = dict()
    param['solver'] = ['saga','liblinear']
    param['penalty'] = ['l1','l2,''none']
    param['C'] = [0.01,1,100]
    search = GridSearchCV(log_reg, param, scoring='accuracy', n_jobs=-1, cv=cv)
    result = search.fit(X_train, y_train)
    print('Higher Score: %s' % result.best_score_)
    print('Hyperparameteres to choose: %s' % result.best_params_)

    test = final_df[(final_df.season == 2020)]
    X_test = test.drop(['finish_position','driver'], axis = 1)
    y_test = test.finish_position
    X_test = pd.DataFrame(scaler.transform(X_test), columns = X_test.columns)

    #search best hyperparameteres for decision tree
```

```
dec_tree_class.fit(X_train, y_train)


for max_l in range(1600,1620):
  model = DecisionTreeClassifier(max_leaf_nodes=max_l,random_state=42)
  model.fit(X_train, y_train)
    print('The Training Accuracy for max_depth {} is:'.format(max_l), model.score(X_train, y_train))
            print('The Validation Accuracy for max_depth {} is:'.format(max_l), model.score(X_test,y_test))
  print('')
train_scores, test_scores = list(), list()
values = [i for i in range(1, 21)]
for max_d in values:
                                                    model = DecisionTreeClassifier(max_depth=max_d,max_leaf_nodes=1605,random_state=42)
  model.fit(X_train, y_train)
    print('The Training Accuracy for max_depth {} is:'.format(max_d), model.score(X_train, y_train))
            print('The Validation Accuracy for max_depth {} is:'.format(max_d), model.score(X_test,y_test))
  print('')
  #evaluate on the test and train dataset
  train_yhat = model.predict(X_train)
  train_acc = accuracy_score(y_train, train_yhat)
  train_scores.append(train_acc)
  test_yhat = model.predict(X_test)
  test_acc = accuracy_score(y_test, test_yhat)
```

```
        test_scores.append(test_acc)
    #plot scores on tree depth
    plt.plot(values, train_scores, '-o', label='Train')
    plt.plot(values, test_scores, '-o', label='Test')
    plt.legend()
    plt.show()
    print(dec_tree_class.score(X_train, y_train))
    print(dec_tree_class.score(X_test, y_test))

    """

    stds = StandardScaler()
    mms = MinMaxScaler()
    scaler = [stds,mms]
    for scaler in scaler:
        X_train = pd.DataFrame(scaler.fit_transform(X_train), columns = X_train.columns)
        print('Scores of every model by:',scaler)
        for model in model_list:
            model.fit(X_train, y_train)
            classification_operations(model,final_df,scaler,num)
            num = num + 1
if __name__ == "__main__":
    main()
```

Figure 9.2 (Project code)