REGRESSION MODELS FOR THE PREDICTION OF FORMULA ONE GRAND PRIX:

Adapted from the Honors Thesis template by Acadia University

by

Liam Tingley (100161568)

Jodrey School of Computer Science

COMP 4983

Supervisor: Dr. Greg Lee

Acadia University

April, 2025

Table of Contents

List of Figures

Abstract

This report seeks to explore the applications of regression modelling as it is applied to the prediction of Formula One Grand Prix results. It will begin by providing background with the motivation for the project, as well as an overview of the domain of Formula One as a whole, before delving into the applications of regression modelling.

Formula 1 has undergone a substantial burst in popularity over the last half decade. As part of this boom period, the volume of analytics and the demand for them recreationally and professionally have skyrocketed. With the expansive interest for data-driven insights into forthcoming race performance, both on an individual and grid-wide level, there has never been a better time to explore the world of data analytics and machine learning through the lens of the world's most popular and prestigious motorsport. Personal motivation stems from a long-standing aspiration to work in the data analytics space with specific focus on sports in a competitive context. That is, to work with teams to find data-driven advantages to get them ahead.

The dataset with which the models were prepared spans all Grand Prix from the 2022, 2023, and 2024 seasons, coinciding with the present set of technical regulations within which teams compete. The dataset with which the best modelling performance was achieved possessed 11 predictors.

The final model ultimately achieved an adjusted R^2 of 72.29%. The model obtained a mean squared error of 3.019, indicating that predictions would typically be roughly 3 positions from the actual finishing result of a driver in a race on average. While this margin of error is greater than desired, it provides the groundwork for a far superior model to be engineered with the inclusion of more features, and a redefining of how error is tracked given the unique context in which predictions are to be considered.

CHAPTER 1: Introduction

This project seeks to showcase the predictive power of AI modelling techniques within the specific domain of Formula One Grand Prix. Formula One is widely regarded as the pinnacle of motorsports, and is often acknowledged as the congregating junction of some of the world's most proficient and innovative engineers and designers. To find oneself in the paddock would be to amongst highly esteemed company, and while this project is not a direct platform off which to jump, it serves as a stepping stone to showcase what is possible in the world of predictive analysis as it relates to Formula One.

1.1 Motivation

Considering Formula One's substantial influx of attention over the last half decade, there has been a substantial increase in demand for data and insights into how the series operates, and projections pertaining to the outcome of future Grand Prix. It is worth acknowledging the application potential in more monetarily motivated spheres, gambling amongst them, but this project intends to approach this subject from a purely curious perspective.

From the standpoint of the competitors, being able to more effectively project the performance of your own team, driver, car, etc. would serve to benefit the competitors greatly. Consider the management of expectations with an informed approach according to previous precedence and understanding underlying trends that might provide pathways for improving your performance in competition.

Another perspective to consider is that of the pundits. A commentator's portrayal of events can make or break a moment on the track, and having the most up-to-date, and accurate information is paramount. As part of this, pundits may find a particularly accurate modelling solution useful for informing how they discuss the performances of drivers throughout a race weekend, discussing form, and how drivers are performing versus expectations throughout a Grand Prix weekend.

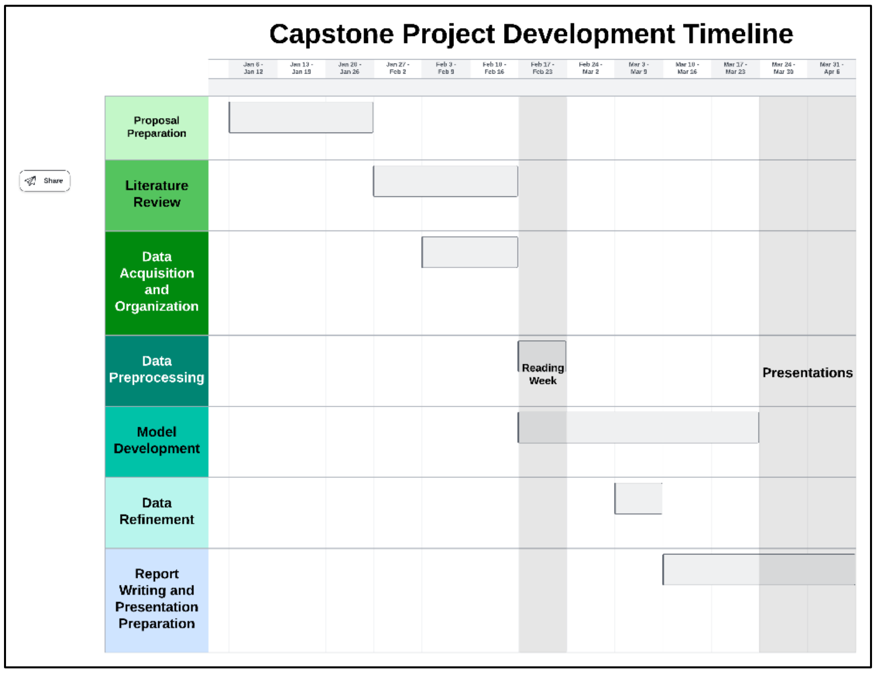
1.2 Objectives and Methodologies

The minimum objective of this project was to develop a model capable of interacting with a database, in this case housed in a CSV file, to inform its predictions. The model would be expected to generate a complete list of finishing results, provided it was supplied with a weekend entry list, and other necessary predictors to be determined.

The expected goals of the project were similar to that of the minimum goals in a slightly expanded form. Namely, the expected goals of the project were to produce multiple models whose performance could be compared against one another, enabling a choice of an optimal solution.

The stretch goals of this project were to take the most accurate model achieved within the bounds established as part of the expected goals and build upon it a user-friendly UI with which users could interact. Users would theoretically be able to prompt the model for predictions about an upcoming Grand Prix. It is important to note however, that prompting in this context would not be conducted in a manner like large language models, but more so specifying the race for which predictions should be generated, and then having those predictions returned.

Figure 1: Capstone Timeline Gantt Chart



The Gantt chart at right serves to showcase the approximate layout of the semester in which this project was developed in an ideal scenario.

Aside from the first three weeks being dedicated to the development of the proposal for this project, the literature review was slated to span the following three weeks. These will be covered in greater detail in *2.2 Related Work*. This portion of the timeline ran mostly on schedule without any interruptions.

The latter two weeks of the above outlined span were intended to be for acquiring data and organizing it as necessary ahead of preprocessing. Unfortunately, this overlapping portion of the timeline was not as successfully adhered to.

Reading Week was set aside as an opportunity for substantial progress to be made with regards to the preprocessing of the acquired data, with initial modelling following thereafter. Similarly to the previous component of this timeline, this ran slightly behind schedule due to extenuating circumstances.

Modelling was assigned a large swath of time, spanning Reading Week and five additional weeks, with refinements to the dataset interspersed part way through. It was during this phase of the project that the actual timeline and proposed timeline began to converge once again, though only to a limited extent.

Finally, there was the preparation of the presentation and this very report, spanning the final two weeks leading into the scheduled presentation window. Due to the progress of the modelling phase before it, this window was condensed to a greater extent than desired to ensure that meaningful implementation had been completed to a larger extent than would have been otherwise.

1.2.1 Application of CRISP-DM

CRISP-DM stands for Cross-industry standard process for data mining. It serves to apply a common approach to the process of data obtention and is used in a broad range of applications. The following subsections discuss each portion of the framework in greater detail as it relates to this project.

The first component of CRISP-DM is Business Understanding [1]. This portion of CRISP-DM is intended to inform the developer of the domain in which they are working, and to familiarize themself with objectives, problems, and what the project might require. In this case, we already outline the objectives and issues encountered elsewhere in this report. However, an understanding of the domain of the project is critical. In this case, I, the developer, have been a longstanding Formula One fan, and a fan of motorsport more broadly, since 2017 and 2007 respectively. As such, this domain, at a fundamental level at least, is well understood. The rules of competition, the nature of said competition, the drivers, and other factors related to informing someone about Formula One are known, with this knowledge being applied in an effective manner.

The second phase of CRISP-DM is Data Understanding [1]. In this phase, we must examine what data sources are available, and what types of refinements, transformations, etc. are necessary for it to be in a modelling-ready state. As will be further divulged throughout the report, various sources were identified at different points as having potential for utilization in this project. Amongst those were the Jolpica-f1 API, which uses a PostgreSQL database, as well as the series of Wikipedia articles outlining the results and information of each Grand Prix individually.

Data Preparation is the third phase, which is covered more extensively in Chapter 3 [1]. In brief terms, this phase was utilized for the uniformization, transformation, and other associated preparations pertaining to the dataset prior to modelling, such as the normalizing of the proportion of laps completed predictor [1]. Modelling follows on the heels of Data Preparation, as the dataset is used to inform architectural decisions pertaining to the model's structure [1]. Evaluation is then conducted after the initial Modelling phase has been completed [1]. This provides an opportunity to analyze the performance of the model, behaviours or trends in the data that were detected by the model, providing the developer with a platform on which to further enhance the model's performance. The final phase is deployment, which in the context of this project, would be to showcase the capabilities of the model as part of the final submission [1].

1.3 Report Structure

This report is composed of the following chapters. Chapter 2 seeks to inform the reader of the context in which the project is being completed. In particular, this chapter will provide background on the Formula One World Championship, what it is, who contests it, etc. This chapter will also examine the analytical potential of examining the series in greater detail. It will be rounded out by showcasing to the reader related works that serve to inform the direction of this project and the modelling processes therein.

Chapter 3 serves to show the reader the direction of development taken by the author of this report. This section elaborates on the preprocessing measures taken, as well as the modelling approaches utilized with the resultant dataset.

Chapter 4 concludes the report by providing results achieved in the implementation and modelling processes. It discusses insights gleamed during the development of this project, and what work can be done in the future to benefit the models' predictive accuracy and capabilities. It also explains some of the challenges encountered throughout the duration of the process, and reflections regarding the potential improvement of the model in the future.

CHAPTER 2: Background

The following chapter seeks to provide some background both regarding the Formula One World Championship, but also the modelling approaches used to predict its Grand Prix. The chapter will commence with an overview of the racing series, explaining what transpires throughout a typical race weekend, and a championship season more broadly. Work related to the process of generating predictions will then be covered, as carried out by Eloy Stoppels and Emma O'Hanlon in separate works pertaining to model predictions [2], [3]. Stoppels takes the approach of artificial neural networks infused with Bayesian probability for predicting individual race results [2]. Whereas O'Hanlon seeks to predict the outcome of an entire season using regression modelling in a similar vein to the approach later described in this report [3].

2.1 Formula One World Championship

The Formula One World Championship is an auto-racing championship that has been contested internationally since 1950. It is widely regarded as the grandest stage available in all of motorsports, with the largest purses, most prestigious circuits, and, arguably, the best drivers in the world. As of the 2025 season, there are slated to be 24 Grand Prix weekends, which will be contested by ten teams, each fielding two drivers per race under normal circumstances (we will briefly discuss instances in which less than 20 drivers contest a Grand Prix at a later stage) [4].

2.2 Related Work

2.2.1 Eloy Stoppels: Predicting Race Results using Artificial Neural Networks (2017)

Eloy Stoppels examines the applicability of artificial neural networks in his 2017 paper *Predicting Race Results using Artificial Neural Networks*. Stoppels utilizes a series of attributes very similar to the ones utilized in this project, including circuit length, number of laps, weather, starting position, total race length, and finishing result [2]. He also includes several additional, more complex variables. Namely the form of the driver, the recent form of their competitors, the best qualifying time, and the difference in seconds between each driver's best qualifying time and the best overall qualifying time for that session [2]. For the purposes of brevity in the context of his project, he uses a subset of four drivers to compare against one another.

Regarding the form attributes, Stoppels examined the three most recent Grand Prix to obtain each driver's own relative form and did likewise for all competitors for each driver for a cumulative, and more complex form attribute [2]. In the early stages of the season, the form attribute would be set to 0, until more than three Grand Prix had elapsed. The total race length feature was also normalized as a proportion of the max race distance in the dataset.

In preparation for modelling, Stoppels states that he divided the dataset at a 90% to 10% training to validation split [2]. For comparing models throughout development, he also states that the validation set is kept the same. A "trial-and-error" method was employed when determining the number of layers and neurons per layer [2]. The success criteria of this process were minimizing the number of incorrect predictions, as well as training costs.

Ultimately the modelling parameters selected by Stoppels were as follows: a five-layer model, with 10 neuron per hidden layer and four in the output layer, a hyperbolic tangent activation function, and cross-entropy cost function, a learning rate of 0.0005, and a maximum number of iterations of 350,000 [2]. The number of neurons in the output layer likely corresponds to the number of positions present in the four-driver subset observed. Given the nature of prediction generation, a probability table was produced by the model, which was then sorted into the most likely outcome. This ultimately yielded a training accuracy of 77% and validation accuracy of 69% on 17 training Grand Prix and 2 validation Grand Prix [2]. The model predicted 47% of races perfectly for the order in which the four drivers would finish [2].

There was a collection of interesting caveats considered by Stoppels that were either forgone or overlooked in this project, but still worth noting. Firstly, Stoppels entirely omitted instances of DNFs in his dataset [2]. This, as we will discuss, was not the case for this project. In contrast, the concept of each driver for a given race representing an individual observation helped to inform the structure of the dataset used in this project.

2.2.2 Emma O'Hanlon: Using Supervised Machine Learning to Predict the Final Rankings of the 2021 Formula One Championship (2022)

The second piece of related work comes from Emma O'Hanlon's 2022 paper *Using Supervised Machine Learning to Predict to Final Rankings of the 2021 Formula One Championship*. It is important to note the contextual differences between O'Hanlon's project and this one. Specifically, hers seeks to predict an entire championship season, whereas this project is focused on individual race predictions. That said, the application of CRISP-DM, usage of regression modelling, and establishing the precedent of transitioning from Python to R for the modelling phase serves to benefit this project all the same.

O'Hanlon's work utilizes data from a much broader range than Stoppels' or this project, noting results from 2010 until 2021 [3]. The data itself was sourced using the Ergast Developer API, the predecessor to the Jolpica-F1 API considered for usage in this project [3], [5], [6].

She employed a sports-specific adaptation of CRISP-DM, which served as a refresher for its incorporation into this project. This was particularly prevalent in her discussion surrounding feature engineering for example, and how the data was being stored and manipulated.

For the implementation of the multiple linear regression model, O'Hanlon utilized a 5% significance threshold when selecting important attributes [3]. Predictors dropped from the final dataset after being initially obtained from the Ergast API included: longitude and latitude of the circuit, qualifying position, cold versus warm weather, and driver nationality [3], [5]. Regarding qualifying position, this is measured based on classification from the qualification sessions and does not consider grid penalties, as such O'Hanlon maintained the grid predictor as it incorporates these penalties. Variance Inflation Factors (VIF) were also used to help build the model with little to no multicollinearity [3].

The artificial neural network used for comparison against the multiple linear regression was also implemented in R, this time using Tensorflow. The train/test split for this model consisted of the 2010 – 2020 seasons as the training data, and the 2021 season as test data. The best working model had the following parameters: 1 hidden layer with 5 neurons, 17 predictor variables, and 1 output layer [3]. It also used a ReLU activation function, gradient descent size of 32, 100 epochs, and a subdivision of 20% of the training set for validation [3].

The results from this project ultimately yielded an R^2 value of 93% in the case of multiple linear regression, with 1.57 RMSE on the test set [3]. Meanwhile, the best-performing artificial neural network achieved an R^2 of 96% with an RMSE of 1.66 [3]. O'Hanlon does acknowledge however, that while the R^2 metric of the ANN is higher than that of the multiple linear regression model, the multiple linear regression model was better at precisely classifying the top 10.

CHAPTER 3: Implementation

The following chapter will examine the environment in which data preprocessing and modelling took place. Beginning with hardware and software requirements of the machinery and software used to complete the work. Followed by a divulgence of the preprocessing measures enacted, as well as the modelling techniques and resultant analysis.

3.1 Hardware and Software Requirements

3.1.1 Hardware

The specifications of the system on which the implementation was conducted are as follows: the system is a Dell Latitude 5420 laptop, which has a Windows 11 Home Edition operating system, 11th Generation Intel Core i5 processor, and 16GB of available RAM.

3.1.2 Software

As outlined in prior sections, there were two primary platforms on which development occurred. The first of these was Python 3.10, which was utilized in a Jupyter Notebook format via a Google Colaboratory environment. This portion of the software, alongside their associated packages, was used for the preprocessing of the data. The primary extension to the standard Python offerings was Pandas 2.2.3. Pandas served to pull data from sources and organize it in its built-in DataFrame data structure.

For the final parts of data preprocessing, and the modelling process of the project, the R programming language and its RStudio GUI were utilized to build, compare, and contrast models. As a component of this, necessary packages included Dplyr, Car, and Metrics [7], [8], [9].

3.2 Data Preprocessing

3.2.1 Data Retrieval and Initial Omissions

Initially, a spreadsheet-based approach was used to lay out exactly what data was going to be necessary for the initial dataset. This consisted of establishing the skeleton of the dataset, selecting an initial collection of attributes, aided by the works referred to previously. Eventually, data retrieval extended beyond the reasonable scope of manual inspection of articles and websites, and required a more automated solution.

A transition was made to Google Colaboratory and Python, with a web scraping approach in mind. Initially, the primary source of these efforts was obtaining data from Racing Reference, a deep online encyclopedia housing results for all types of racing series from NASCAR to Formula One. Unfortunately, this platform refused to permit web scraping efforts, returning a 403 code, indicating successful requests, but forbidden response. Wikipedia was ultimately settled upon for sourcing data given its consistent structure for race pages. It is worth acknowledging the volatility potentially present in sourcing data from Wikipedia, but for the purposes of acquiring this data, which was then vetted using analytical tools, it proved a suitable resource.

Once data was retrieved and successfully pulled into the Google Colaboratory environment, the inspection process began. Tables obtained from Wikipedia included several extraneous pieces of information that required filtration including additional rows for fastest laps and references. Additional values, such as dedicated entries concerning the 107% rule were also identified for omitting purposes.[1] Additional data points were observed within the tables, specifically with regards to finishing position. These values included “Ret”, “WD”, and “DNS”.[2] In the latter two cases, instances with this classification were omitted from the dataset entirely as they never officially took the start. Another interesting observation that necessitated adjustment was the treatment of footnotes pulled from the tables, which were treated as ordinary components of the integers. This required handling of finishing position in a row-wise format, as opposed to pulling data from the tables directly.

3.2.2 Handling Sprint Weekends

A unique wrinkle introduced to Grand Prix weekends in recent years has been Sprint races. These are short, 100km races run on the Saturday leading into a Grand Prix Sunday. As a result of their introduction, the

[1] The 107% rule refers to a condition stipulated by Formula One that drivers must set a qualifying time within 107% of the pole-sitter. Otherwise, they may be denied entry into the Grand Prix.

[2] “WD” indicates a withdrawn entry, meaning the team does not start a given driver or car in the Grand Prix event that weekend, a decision made prior to its commencement. “DNS” indicates a driver was slated to compete in a Grand Prix, but due to a technical failure, error on the formation lap, or similar, they were unable to take the start.

formatting of Sprint weekend Wikipedia articles is slightly different, particularly with regards to the order in which data tables are presented. Up to this point, ordinary Grand Prix pages presented their race classification tables in the same position of a list returned using Pandas each time. However, in the case of Sprint races, this would not be the case. As such, a complete reworking of the fetching function, *findWikiRaceTable*, was necessary. The solution adopted for this approach was to retrieve the total number of scheduled laps for the Sunday Grand Prix from the overview table at the top of the page. Then, this value was compared to the leading entry in each table, specifically the "Laps" column of each table, until a matching value was found. The table that this value was located in was deemed to be the race classification table. Associated Sprint tables would be bypassed under this logic given that they would always be a shorter distance, and thus a lesser number of laps, than the main Grand Prix.

3.2.3 Functional Preparation

There was a collection of functions that the data preprocessing portion of the project was broken down into. In addition to the above mentioned *findWikiRaceTable*, several other functions were developed to enhance the efficiency of the preprocessing phase. These were as follows: *fetchOverviewData*, *appendSeason*, and *simplifyTeamNames*.

The first of these is quite self-explanatory and was alluded to in prior sections. The *fetchOverviewData* function serves to retrieve the opening table from the provided Wikipedia article. This includes the retrieval of the number of scheduled laps for a given Grand Prix, as well as the weather conditions on race day.

The next function alluded to is the *appendSeason* function. This is particularly the most important of any of the methods discussed. This function takes in the Pandas DataFrame to which the retrieved data will be appended, as well as a list of the web URLs from which data will be drawn. It iterates over each URL provided, within which it iterates over the race classification table therein. The desired data is extracted from each row of the table and formatted as necessary for the existing DataFrame passed as a parameter. The formatted row is then appended to the DataFrame.

The *simplifyTeamNames* function serves to simplify team names from their heavily sponsor-infused versions to their colloquialisms. This is particularly important for ensuring the common grouping of teams that have undergone rebrands during the time this dataset seeks to capture, while still being the same team under the hood. Take, for example, "Alfa Romeo", or "Stake", these teams have commonly been referred to as "Sauber" due to the underlying Sauber team's longstanding involvement in Formula One. As such, because there are no rebranding-related personnel changes, simply a new coat of paint.

3.2.4 Final Steps in R

The final preprocessing measures were taken in R before the modelling phase began. The first of these measures was the conversion of all categorical predictors to factors, as stipulated by R. This operation was conducted on the year, gpName, weather, driver, team, and eventually the dnf predictors. In the case of the team predictor, one hot encoding was applied after initial testing to simplify computational complexity and potentially enhance modelling performance. Unfortunately, this approach did not yield any improvements given the nature in which R encodes categorical variables from the outset of model creation.

3.3 Modelling

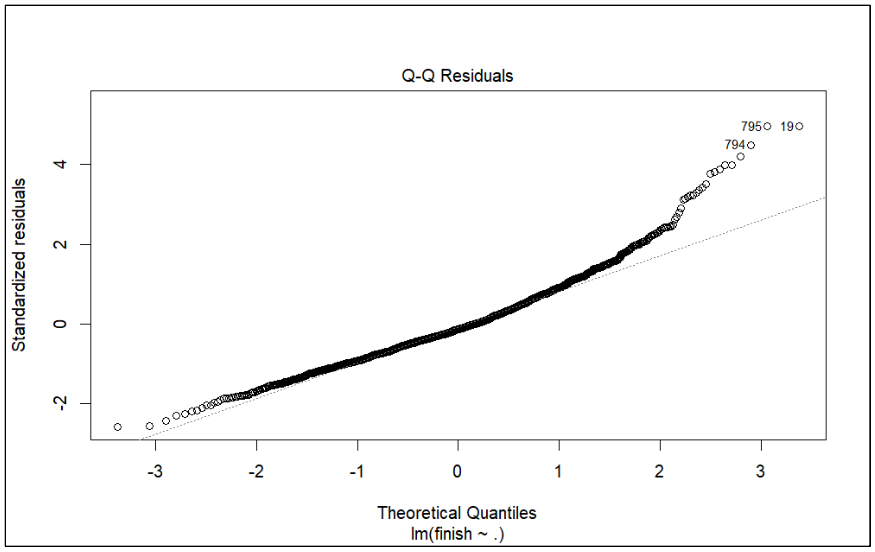
Once the dataset was successfully prepared, initial modelling commenced. Each of the models discussed here are multiple linear regression models, with slightly differing feature selection. Modelling solutions in Python were considered and primitively explored, however due to time constraints as well as greater familiarity already existing for modelling in R and RStudio as a result of previous and concurrent statistics courses, the decision was ultimately made to model in R. There is existent precedent for this transition in the literature review, as Emma O'Hanlon also states that her initial modelling approaches in Python ultimately led her to the decision to transition to R for the modelling portion of her work, with an additional source form 2018 stating that both Python and R "can work collectively when trying to achieve a common goal" [3].

The initial approach adopted within this project contained 10 predictors and finishing position response variable. The predictors were as follows: year, round, Grand Prix name, circuit length, proportion of laps completed, race length, weather, driver, team, staring position, and finish. The model was trained on all 1,350 observations obtained as part of the data preprocessing phase. The initial results of this model without the introduction of any more complex terms included the following: an adjusted R^2 of 0.6667, with a residual standard error (RSE) of 3.311 on 1314 degrees of freedom. This model wasn't terribly inspiring, but did offer the opportunity to explore the first bit of data refinement as a reference marker.

That first refinement was the dimensionality reduction of the weather attribute. Initially, this attribute was simply pulled from the Wikipedia article of each Grand Prix and placed into the model. Of course, the descriptions page-to-page would vary, even if they effectively documented the same conditions. As such, the attribute was simplified down to two classifications: dry and wet. Wet races were classified as any Grand Prix to have experienced rain during its duration. The threshold of what makes a race a "wet" race is typically when race directors disable the use of DRS[3]. Upon implementing this change, the model did improve, but only very slightly. It achieved an adjusted R^2 of 0.6675, and an RSE of 3.307 with 1284 degrees of freedom.

[3] DRS refers to Formula One's Drag Reduction System. A system in which, when a driver is within one second of a car in front, they are permitted to flatten the rear wing of their car, reducing drag and increasing top speed on straightaways.

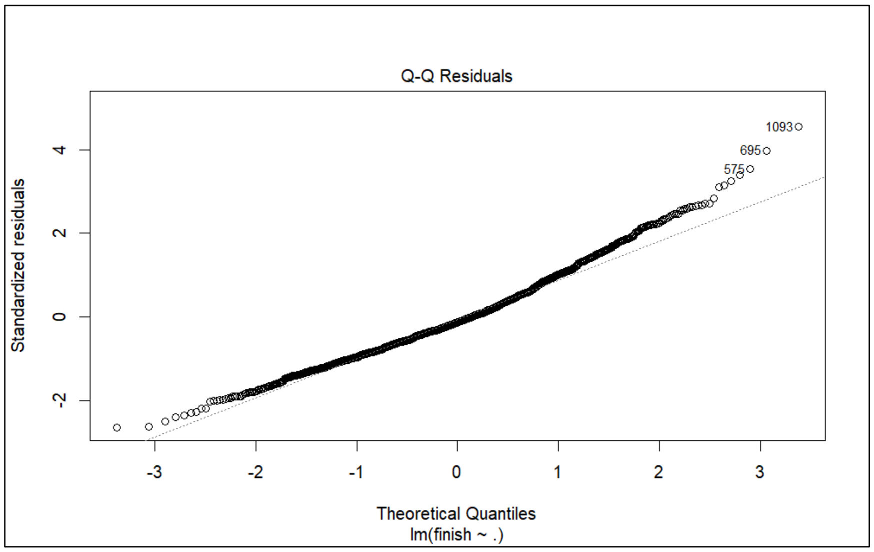
Figure 2: Initial Model QQ Plot



Upon inspecting this initial model and its diagnostic plots, it was discovered that there was some nonnormality in the right tail of the QQ plot, pictured right. This indicated that there was a significant discrepancy in instances where drivers were predicted to finish highly but ultimately finished very poorly. Of course, a logical explanation for instances such as these is that the driver failed to finish the Grand Prix, and as such, was classified poorly. In response to this, the *fetchWikiRaceTable* function adapted to classify a driver as having DNF[4] using the way in which their Wikipedia entry was recorded in the table, as an integer, or as ”Ret.” DNFs were appended to the database as a Boolean following these changes.

[4] Did not finish.

Figure 3: DNF Model QQ Plot



Once DNFs were introduced to the model, the performance improved substantially. It achieved an adjusted R^2 0.7229 and an RSE of 3.019, an increase and decrease of 0.0554 and 0.288 from the previous model, respectively. This was further reinforced when an ANOVA analysis was run to compare the two models, revealing a significantly small p-value, indicating a statistically significant improvement in the DNF model. The DNF model's QQ plot is pictured above (Figure 3), which visualizes the flattening of the discrepancy for drivers predicted to finish high, but not due to their inability to finish.

The final component worth noting with regards to the modelling phase in R is the *predictRace* function, and its potential applications in a more fleshed-out form. At present this function serves to generate an ordered list of finishing results, provided an entry list race vector, as stipulated by the minimum and expected goals of this project. The finishing order is produced by sorting the predictions for each driver provided in the race vector, with the lowest value (best finishing position) sorted to the first entry in the DataFrame, down to the largest value (worst finishing position). The index of the newly sorted DataFrame is what is intended to be considered the predicted finishing results produced by the model.

CHAPTER 4: Conclusion

4.1 Conclusion

This project served to explore the applications of regression modelling in the context of Formula One Grand Prix prediction. It provided valuable insight into the planning of projects of this scale, as well as the time management necessary to execute them to their fullest potential. There is plenty of growth potential both with regards to the produced solutions, but also to take into broader contexts beyond the scope of this project as well.

4.2 Future Scope

There were plenty of additional avenues down which potential model improvements were achievable. Unfortunately, due to significant time constraints, several of these were unable to be realized. That said, the following will attempt to cover a series of additions and adjustments possible using the accompanying solution that could enhance its predictive power.

First and foremost, given the best achieved model achieved a peak R^2 of 72.29%, that is, 72.29% of the variance present in the response was explainable using the provided predictors, further feature engineering was required to account for a greater degree of the variance present in the finishing results. While acknowledging the inherent variability present in auto racing, be it from mechanical failures, driver mistakes, or otherwise, perfect predictions are unlikely to be achieved. However, if we compare the model produced for this report with those covered in the works of Stoppels and O'Hanlon, we'll observe that a far greater R^2 is achievable. Stoppels' incorporation of a form metric, both for the opposition of each driver individually, but also for their respective competition, would likely prove immensely insightful [2]. Similarly, obtaining comparative performance in both qualifying and the race using each driver's best lap time relative to the best set in that session would likely provide the model with additional insights to enhance their predictive power.

The further feature that would likely improve the model's sensitivity to new and more recent data would be to insert a time series variable into the model. This would take the form of an attribute indicating the number of Grand Prix to have taken place since the race from which the observation was recorded occurred. Alternatively, this could be introduced in the form of days since the given Grand Prix occurred, this would have the additional effect of potentially penalizing races from previous seasons at a progressively greater rate given the off-season Winter spells, arbitrarily inserting large dead zones into the potential range of the variable.

Another avenue that would be considered in a far greater capacity with the introduction of a large swath of new predictors would be to explore the omission of statistically insignificant features. Upon experimenting with this in its purest form when referring to the *summary* function available for regression models in R, it was observed that there was a slight decrease in the R^2 coefficient for the reduced model. In addition to this, several factors were important in a contextual capacity rather than a statistical sense, creating a dilemma surrounding what information can be gleamed from the predictions.

While a primitive train and test split was utilized to explore model generalizability, there were plans to make use of cross-validation and, if time had permitted greater feature engineering, the potential exploration of the lasso technique. K-fold cross-validation is a valuable tool for obtaining a more accurate estimate of the true performance of a model and would be useful in this context if the folds were appropriately distributed. In a similar vein, with the introduction of a large volume of new features to the model and then looking to simplify, the lasso technique would prove useful for informing the selection of the most important predictors.

Exploring different modelling techniques more broadly, originally, it was planned that there would be multiple techniques explored and tested for comparison as part of this project. One of those techniques was intended to be artificial neural networks, as discussed in the work of Stoppels. Considering the performance achieved by the model referred to in his work, it would appear to be the next logical step following the perceived optimization of the regression modelling technique. Further to this point, given the potential nonlinear influences present in some predictors, the exploration of generalized additive models is another conceivable avenue down which modelling could go. This avenue wasn't explored in as great a depth due, like artificial neural networks, to time constraints, but also due to the author's uncovering of them through a course taken simultaneously, and the potential performance benefits being revealed soon after.

It was not until late in the developmental stage that the OpenF1 API was discovered [10]. This API, open-source and easily accessible, possesses real-time data concerning all things at the track. Obtaining lap time data on a per lap basis, both with regards to time, as well as pit stop information, would have proved beneficial for model performance when considering how drivers performed in previous Grand Prix. The only discernible drawback of this resource is its lack of data for the 2022 season, meaning the data could become skewed if not handled properly.

4.3 Challenges and Reflections

Chief among the challenges encountered during this project was the balancing of time allocations. Although an initial timeline was included in the submission of the proposal at the start of the term, this was not stuck to terribly strictly. As such, progress was achieved in bursts spaced throughout the term. Upon reviewing the timeline and how progress was ultimately made during the term, it’s been considered that less time ought to be spent on the literature review, beyond the initial informing of oneself in a primitive capacity. Furthermore, given the extent to which predictive models can be refined, spending more time on the implementation will be critical moving forward. Data preprocessing took an extensive duration, and ensuring data quality consumed significant portions of development time, however this is often known to be the case.

A particular oversight when approaching data preparation was the failure to include more observation-specific attributes. Several of the predictors in the dataset served to contextualize in what Grand Prix the observation was being recorded, as opposed to more performance-based metrics.

Additional challenges encountered included those experienced in the early stages of modelling. In particular, the initial approach was to keep all programmatic implementation in Python, however, as time continued to dwindle, and the relative inexperience with modelling in this fashion in Python, moving to R was deemed necessary in the interests of completing the project to any reasonable degree on time. That said, the sklearn and statsmodels.api packages were initially trialed, but between their handling of categorical variables, as well as what appeared to be extensive manual computation of key metrics, the transition seemed necessary [11], [12].

During the modelling phase in the R environment, most predictors, including categorical indicators, were provided with a coefficient and p-value, amongst other metrics. However, a collection of four teams did not possess any values in the summarized R model output. This was acknowledged by the system as being due to the presence of singularities in each of these cases. Singularities in this context are present due to these predictors being highly, if not perfectly, correlated with other predictor variables. As such, these predictors could not accurately be assigned the aforementioned values. Of course, given the nature of the problem, omitting only a subset of the teams competing would not make contextual sense for predicting Grand Prix, which rules out the possibility of dropping them from the dataset. Alternatively, a correlation matrix could be constructed to see with what other predictors these indicators are highly correlated and handle those interactions accordingly.

Finally, refinements to the output format and interpretation of the *predictRace* function during the R preprocessing steps would also be desirable. The presentation of these results in the console would ideally be more polished, with a summary of the overall performance of the race prediction at the bottom of the output, particularly including the adjusted RSE to consider the finishing position assigned after sorting the predicted response in ascending order. A further enhancement to this function was to ensure that the provided race vector contains data from the same Grand Prix of the same year. For larger datasets, this would be particularly useful if using an index slicing approach to pass a subset of a larger DataFrame into the function for predictions.

The implementation for this project can be found at the following GitHub repository link: https://github.com/TmaiL03/COMP4983CapstoneProject

References

1. N. Hotz, “What is CRISP DM?,” Data Science PM, https://www.datascience-pm.com/crisp-dm-2/ (accessed Apr. 4, 2025).
2. E. Stoppels, Predicting Race Results using Artificial Neural Networks, https://essay.utwente.nl/74765/1/FinalThesisEloyStoppelsNoCompany.pdf (accessed Jan. 30, 2025).
3. E. O’Hanlon, Using Supervised Machine Learning to Predict the Final Rankings of the 2021 Formula One Championship, https://norma.ncirl.ie/6628/1/emmaohanlon.pdf (accessed Feb. 18, 2025).
4. “F1 - the official home of Formula 1® racing,” Formula 1® - The Official F1® Website, https://www.formula1.com/en/racing/2025 (accessed Apr. 4, 2025).
5. “API documentation,” Ergast Developer API, https://ergast.com/mrd/ (accessed Jan. 26, 2025).
6. J. Perry, “Jolpica/Jolpica-F1,” Jolpica F1, https://github.com/jolpica/jolpica-f1 (accessed Jan. 26, 2025).
7. H. Wickham, R. François, L. Henry, K. Müller, D. Vaughan, dplyr: A Grammar of Data Manipulation, https://CRAN.R-project.org/package=dplyr (accessed Mar. 22, 2025).
8. J. Fox, S. Weisberg, An R Companion to Applied Regression, Third edition. https://www.john-fox.ca/Companion/ (accessed Mar. 22, 2025).
9. B. Hamner, M. Frasco, Metrics: Evaluation Metrics for Machine Learning, https://CRAN.R-project.org/package=Metrics (accessed Mar. 22, 2025).
10. B. Godefroy et al., “OpenF1,” OpenF1 API | Real-time and historical Formula 1 data, https://openf1.org/ (accessed Mar. 20, 2025).

11. F. Pedragosa et al., Scikit-learn: Machine Learning in Python, https://jmlr.csail.mit.edu/papers/volume12/pedregosa11a/pedregosa11a.pdf (accessed Mar. 22, 2025).

12. S.Seabold, J. Perktold, statsmodels: Econometric and statistical modeling with python, In 9th Python in Science Conference (accessed Mar. 22, 2025).

Appendix A: Installation and User Manual

1. Run the following command in a terminal of your choice:

   ```
   git clone https://github.com/TmaiL03/COMP4983CapstoneProject.git
   ```

2. The cloned repository should house a `database.csv` file. This can be utilized directly with the `COMP4983Modelling.R` file as outlined in Step 5.
3. In the event that the `database.csv` file is not present, open the `COMP4983WikiDataWebScrape.ipynb` file.
4. Run each chunk of the `COMP4983WikiDataWebScrape.ipynb` file in sequential order, specifying the desired path of the output `database.csv's` location. By default, this will be in the same directory as the web scrape file.
   a. NOTE: If `database.csv` file already exists in your directory and the web scrape file is run, it will NOT overwrite the existing CSV file. To refresh the database, or to test the web scraping file's capabilities, please delete your existing `database.csv` file, and rerun your `COMP4983WikiDataScrp.ipynb`.
5. Once the `database.csv` file has successfully been produced, open the `COMP4983Modelling.R` file in your desired location.
6. Once open, run the file in its entirety to read in and process the database instances, generate plots, and produce the models.
   a. If in RStudio, using Ctrl+Shift+Enter will run every line in the script in sequential order.
7. To inspect the dataset from within RStudio, in the Environment pane, double click on the database instance you wish to inspect.
8. To generate predictions for a particular race, refer to line 104, where the indexes between which a vector can be specified for a given race. To run this line in isolation to refresh the vector's value, select the line (or place your cursor on the line) and press Ctrl+Enter.
   a. Remember that R uses 1-based indexing.

9. Using line 107, press Ctrl+Enter from within RStudio to run this solitary line to generate predictions using the DNFs model. The output of the function will be returned in the console.