University of Toronto

Peace of Mind in Formula One

How do contracts and team stability affect driver performance?

Aamir Ali

STAD94

Professor Ken Butler

10 August 2022

Table of Contents

Abstract

Purpose: The purpose of this project is to explore how statistics can be analysed to draw conclusions about real-world phenomena. Specifically, this project will look at Formula One driver statistics to determine if there is a relationship between job congruence and driver performance.

Methodology: This study will use data collected from the FIA results archive, a database that contains information on every Formula One grand prix since the beginning of the championship in 1950 (FIA). This data will be processed to generate information about a driver's performance relative to their teammate.

Results: The results from this paper show that there is only a small relationship between driver performance and job congruence, and that not enough drivers have stayed at one team for a long time to conclusively determine whether it matters.

Limitations: the biggest limitation in this study is its highly quantitative nature, which disregards the psychological and social factors that impact an athlete's performance.

Introduction

Background

Performing well in the workplace is the culmination of many factors, some of them concrete and tangible, and others that are less so. One of factor that could potentially impact performance is job congruence and security. This is defined as having had one job for a long period of time without being afraid of losing it for reasons out of one's control.

Sport is a profession where performance is intrinsically linked to one's employability, with consistently strong performances leading to long careers. This study will specifically focus on the motorsport of Formula One, a sport where drivers compete in purpose-built cars to drive as fast as possible at numerous circuits around the world, culminating in the fastest driver being awarded the title of World Driver's Champion.

Formula One is widely considered the pinnacle of motorsport, and those who drive in the sport are considered the best drivers in the world. In addition to this, the sport of Formula One was chosen for this study due to my personal knowledge of the sport, having been an avid viewer since 2012.

This study will draw upon the knowledge gained during this decade of exposure to the sport and data collected from the FIA (Federation Internationale de l'Automobile) results archive. All information

presented originates from these sources unless otherwise specified. The conclusion of this project will also connect the results of the project to established psychological literature.

While the Formula One driver's job is to pilot a car as fast as possible, each team's car is an immensely complex and bespoke prototype built in complete secrecy, leading to numerous minute differences that add up. Furthermore, teams are comprised of hundreds (sometimes thousands) of employees who work differently, creating different social environments and professional cultures within each team.

Research Question

This study will explore how changing teams, or job congruence, affects driver performance. More specifically, the study will look at drivers' mean stint lengths during their career and compare it to their performance to determine the relationship, if any, between job congruence and performance within the sport of Formula One.

Hypothesis

This study's hypothesis is that higher job congruence improves driver performance. The assumption is that a longer stint at a team allows the driver to better adapt to the car, team culture, and have an influence on the way the car is developed, culminating in their ability to perform better.

Methodology

Variable selection

First, it's important to establish how driver performance will be measured. Since Formula One drivers on different teams compete in machinery with potentially large performance differences, it would be unfair to directly compare a driver in a car capable of winning races with a driver in a car that can barely hang on to the back of the grid. However, since each team produces identical cars, drivers' metrics will be measured as relative to their teammate. Returning to the example above, if the driver in the winning car struggles to beat their teammate but the one in the backmarker consistently finishes second-last, the latter will compare favourably to the former since they're fulfilling more of the car's potential.

The metrics chosen to measure performance are **race finishing position** as this is how points are earned, **qualifying time** as this represents a driver's peak performance over one lap, and **championship position** which reflects a driver's performance over an entire season.

Note that while there have been five different qualifying formats (eight if one counts variations as a new format) in the measured period, each is a variation on how long and how many attempts the drivers have for driving as fast as possible over one lap. An alternative would be a driver's fastest lap in the race, but there hasn't been an incentive to try and achieve the fastest lap until 2019 (when a point was introduced

for the fastest lap if the driver finishes in the top-10). As such, qualifying, which *requires* a driver to go as fast as possible, is still an accurate and better measure of a driver's peak performance.

Additionally, the code in the appendix will show that championship points were also measured for drivers. This was removed from the results analysis due to the numerous different points systems between 2000 and 2020 and the similarity to the championship position metric.

Driver selection

The official Formula One championship started in 1950, meaning that there is over 70 years of driver performance data to be potentially analysed. From this period, teams were mandated to have exactly two drivers in a given race starting from 1981, so the initial dataset contained all drivers from 1981 – 2020 (Since there is only one year of data on the sprint-qualifying format and it is such a large deviation from normal qualifying, 2021 was omitted). From there, the data had to be trimmed to results from 2000 – 2020 due to technical limitations.

While measuring driver statistics post 1981 means that there is just one teammate to compare against during races and qualifying sessions, there can be mid-season driver changes or substitutions for a variety of reasons. For example, Heinz-Harald Frentzen jumped ship form the collapsing Arrows team to Sauber in 2000 where he substituted for Felipe Massa at one race that year, and then replaced him as their full-time driver from 2003. Consequently, there were three drivers to compare at Sauber for the 2002 championship standings, Nick Heidfeld, Felipe Massa, and Heinz-Harald Frentzen (FIA). In such a situation, only the two drivers who started the season in the car will be measured for championship position and points. For example, Heinz-Harald Frentzen will be compared to Enrique Bernoldi, his Arrows teammate in 2002, and Nick Heidfeld will be compared to Felipe Massa (and vice versa) since they started the season as Sauber teammates (FIA).

If there is a temporary substitution at the beginning of the season, that driver is *not* counted. For example, even though Antonio Giovinazzi replaced Pascal Wehrlein at Sauber for the first two races of 2017 due to injuries, Wehrlein is counted as Marcus Ericsson's teammate that year (FIA).

Finally, drivers who did not compete in more than 22 races or had a mean stint length of less than two years were excluded from the data. The minimum race number was set at 22 because the longest seasons between 2000 and 2020 were 2018 and 2019, with 21 races each, meaning that eligible drivers would have had to compete in one full season and then remain in the sport for at least one more race. However, there are drivers such as Ricardo Zonta, who was the starting driver for only one season for BAR in 2000 but managed to rack up more than 22 races total as a substitute for Jordan and Toyota. This means that his

mean stint length is just one season, so him and drivers in a similar position were also removed by setting the minimum mean stint length to two seasons (FIA).

The intention of these restrictions is to focus the analysis on drivers who have had the time to acclimatise to the sport and are of a sufficient quality once up to speed; the assumption being that sub-par drivers will not be retained by teams after a poor season, forcing them to keep moving around the grid or retire. The only exception to this are pay drivers: sometimes, a team in financial trouble will hire a driver who may have sub-par performance but has financial backing that will alleviate the team's troubles and allow it to remain in the sport. These drivers do often improve and end up justifying their place on the grid, but some do not and end up taking away an opportunity to race from other potential drivers with less backing.

Data Sourcing

With the benefit of hindsight (and introspection), it should have been obvious that Formula One is a haven for stats nerds, and that driver history databases would be plentiful on the internet. Without this knowledge though, the method used to gather data was to take data from the FIA Results and Statistics classification database by building a web scraper. This was done in Python using the BeautifulSoup4 library.

Data clean-up and manipulation

Before formatting the collected data into tables, the first decision to be made was how to treat drivers classified who did not finish (DNF) the race, were disqualified (DSQ), or were not classified with a numerical position for some other reason. Since relative driver position is calculated by subtracting the driver's position from their teammate's, every driver needed a numerical finishing position. As such, drivers were given the position of where they were classified in the finishing order of the FIA database. Therefore, if 15 drivers finished a race, 4 drivers were classified as a DNF and one did not start the race (DNS), those last 4 would be given positions 16-20 in the order they were listed in the database. This order is informed by the distance completed so is not an overly large leap to make.

The next decision was regarding team-names. The large costs involved in running a Formula One team necessitate sponsors, and sponsorships are sold everywhere from team clothing and car liveries to the actual name of a team. These 'title-sponsors' can change the name of a team from McLaren Racing, for example, to Vodafone McLaren Mercedes (FIA). As such, it is important to distinguish between when a team has simply switched title-sponsorship and when they have changed ownership and become a new entity. As a rule, teams were renamed (and therefore drivers who stayed were given a second constructor in their career) only when there were meaningful management changes, regardless of the scale of the rebranding. For example, when Ford sold Jaguar and Giancarlo Minardi sold his eponymous team to

Dietrich Mateschitz who renamed them Red Bull and Toro Rosso respectively to promote his energy drinks company, this name change was reflected in this project's database (Tran). Similarly, When Vijay Mallya's Force India team was bought by Lawrence Stroll and renamed to Racing point, this change is also reflected in the database (Baldwin).

However, Stroll's renaming of the team to Aston Martin to promote the luxury automaker did not constitute a change in management, so both Racing Point and Aston Martin are the same team (and therefore called Aston Martin) in the database (FIA). Sauber's stint as BMW in the early 2000s and current guise as Alfa Romeo are also treated as one team and therefore called Sauber in the database (this decision can be argued as BMW was quite a significant influence in the management of the team, but Peter Sauber never fully gave up control over the team) (FIA).

There are also cases where new management comes in but chooses not to rebrand the team. Examples include the handing over of McLaren Racing from Ron Dennis to Zak Brown at the end of 2016, and the purchase of the Williams Team from Frank Williams' family by venture capital firm Dorilton Capital (Taylor; WilliamsF1). In both cases efforts were made to preserve continuity, so this is mirrored in the database where the teams are counted as the same entity.

After the data was modified to conform with the above decisions, the data was imported into R. Tools from the Tidyverse library were used to remove extraneous variables such as driver numbers, nationalities, and fastest race lap.

There were now two tables, one containing race and qualifying results and one containing overall championship statistics. The former contained one row for each driver's absolute results in every race of every year in the measured period and the latter contained championship position and points for each driver for each year.

While the race and qualifying results table was ready to be manipulated, the championship table had more than two drivers for teams that had done mid-season swaps or substitutions. These had to be manually changed using the driver selection criteria outlined above.

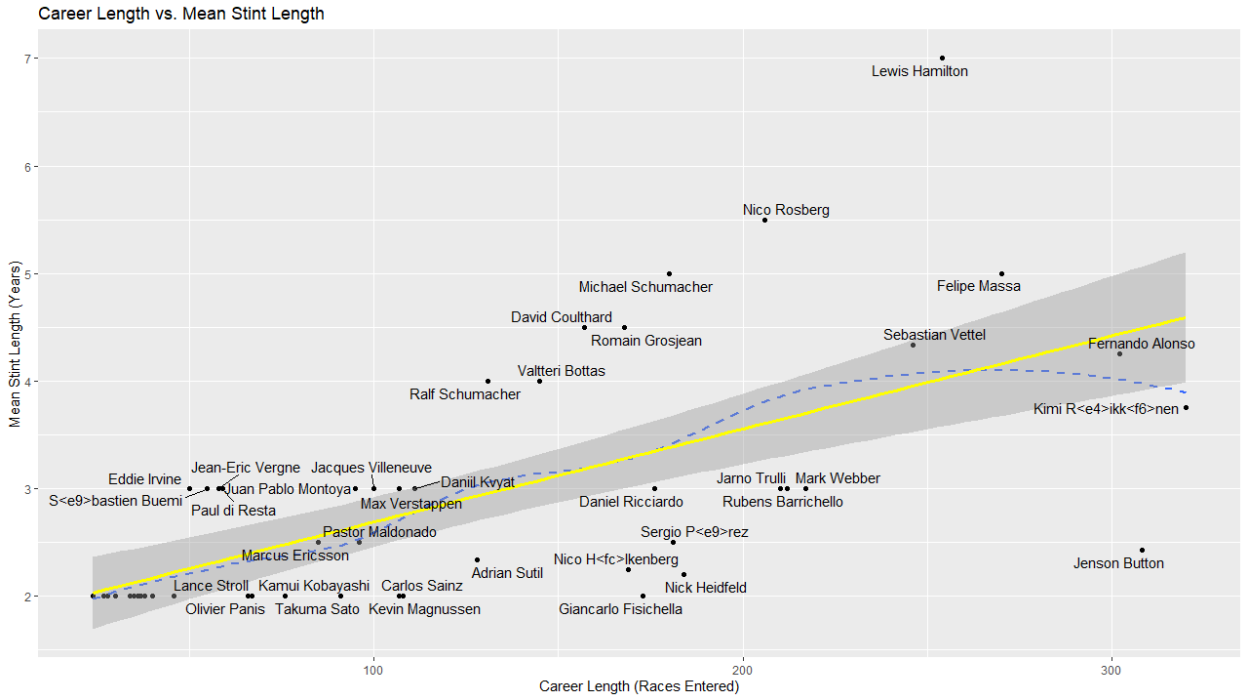
The tables were then processed in Python to perform the subtractions and get relative results for each driver. This new data was once again passed into R where each driver was assigned their mean result for each of the three response variables.

Finally, the assigned team for each driver was used to calculate stint lengths and averaged to get the **Mean Stint Length**, the desired explanatory variable.

Each of the response variables was then plotted against the Mean Stint Length for the population of 55 drivers, and the outputted graphs are discussed in the results section below.

Results

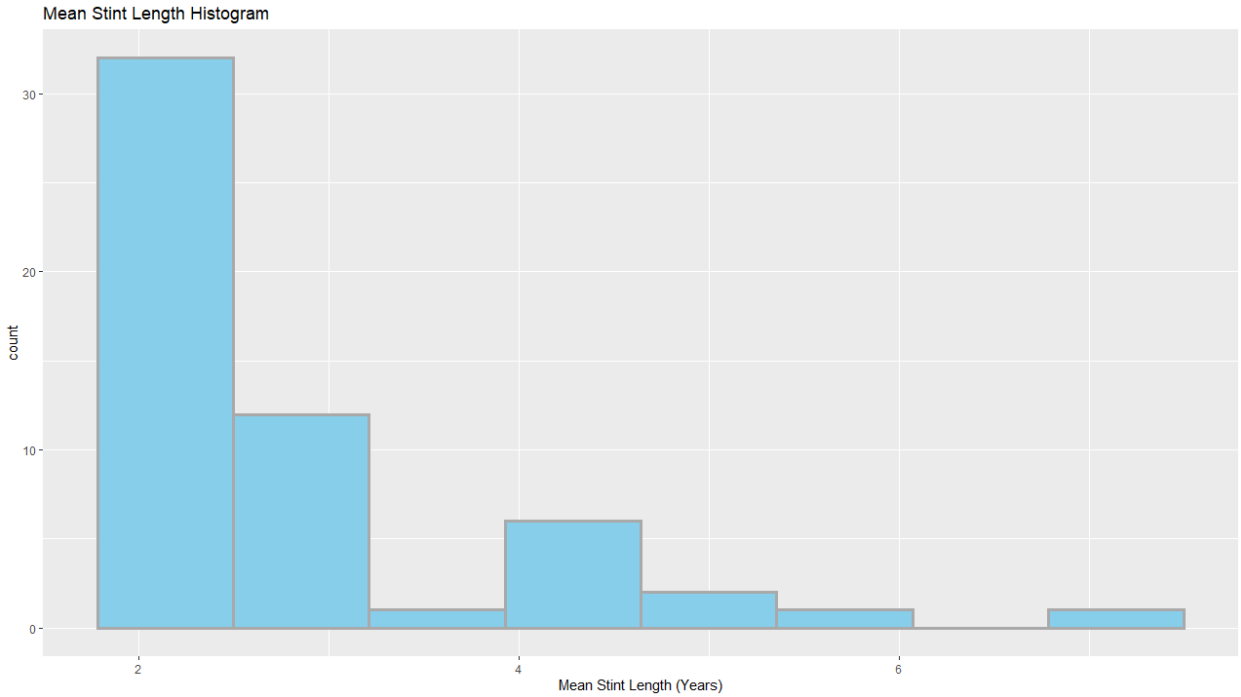
First, below is a graph showing the relationship between career length and mean stint length:



This graph shows that career length and mean stint length have a moderately positive correlation with a coefficient of 0.66. This means drivers with longer careers in Formula One tend to stay at one team rather than switching between many. Interestingly, the dotted blue trend line deviates from the yellow best-fit line after approximately 200 races, while this could indicate that drivers with careers longer than that switched teams more often, it is more a reflection of the lack of drivers with such long careers, as shown by the increasing error-bar shading. Only 10 out of the 55-driver population have over 200 races.

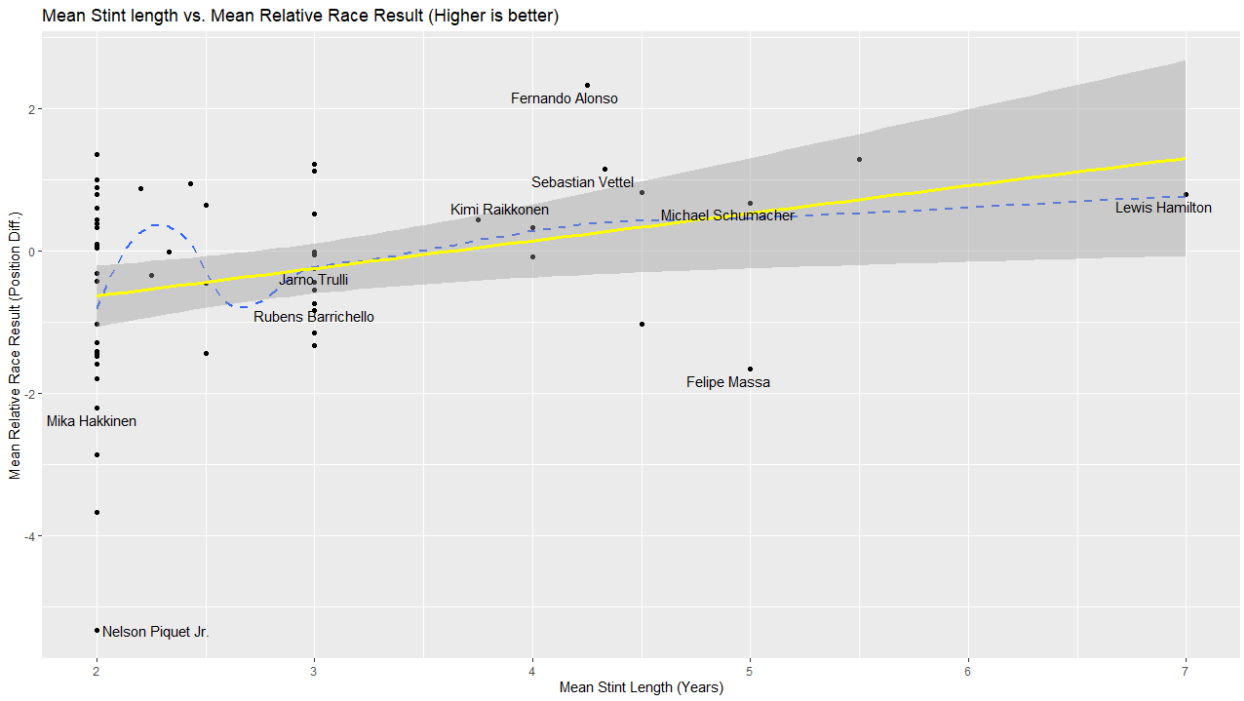
Looking at individual drivers, a notable outlier is Jenson Button. He had an impressively long career but a mean stint length of just 2.4 years despite driving for McLaren for seven years, from 2010-2016. Even though Button himself never moved teams, the team (BAR) he drove for was constantly changing ownership, resulting in the number being high on paper (FIA).

In contrast, Lewis Hamilton has only driven for two teams in his entire career: McLaren from 2007-2012 and then Mercedes from 2013-2020 (FIA).



This histogram visualises the distribution of mean stint lengths which is clustered around 2 and 3 years with a large right-tail. In numerical terms, the mean- mean stint length is 2.81 years, and the median is 2.33 years. This shows that most of the 55-driver population spent less than three years with any given team. Note that this is not reflective of *every* driver in Formula One as there are many with shorter careers. Introducing those to the study wouldn't have changed the results too much but may have changed this histogram to be more normally distributed (if still distinctly right-tailed).

While this is pure conjecture since driver contract details are highly confidential, this trend is likely also visible in driver contract lengths, with most only being signed on for two to three years at a time.



This graph shows that there is a slight positive relationship between stint length and relative race result with a correlation coefficient of 0.32. This means that higher job congruity leads to better race results which supports the project's hypothesis.

The wave in the dotted trend line (visible in the qualifying and championship graphs as well) is not a reflection of a dip and rise in performance but rather the fact that there are many more with short careers than long ones as shown in the histogram above.

The first labelled driver to look at is Mika Hakkinen. His lower performance is indicative of the fact that drivers who only had the tail-end of their career captured in the 2000-2020 timespan will be seen in a negative light since their accomplishments may be missed out (FIA). Hakkinen had an impressive career, winning two world-championships in 1998 and 1998, and drove quite well in the preceding years as well. These seasons were unfortunately not covered in this project (FIA). Like Mika Hakkinen is Rubens Barrichello. To date, he holds third place in the record for most grands prix entered (FIA). However, the majority of these were in the 1990s, meaning his metrics also suffer for this reason (FIA). Michael Schumacher also had approximately half his career in the 1990s when he won his first two world titles, but he proceeded to win five more from 2000-2004, so his results are superlative regardless (FIA).

The next driver to examine is Felipe Massa. While he had a fairly long career with high job congruity, his results are significantly lower than average (FIA). This is because his career is emblematic of the perennial number-two driver. He started the zenith of his career at Ferrari in 2007 when he had the world

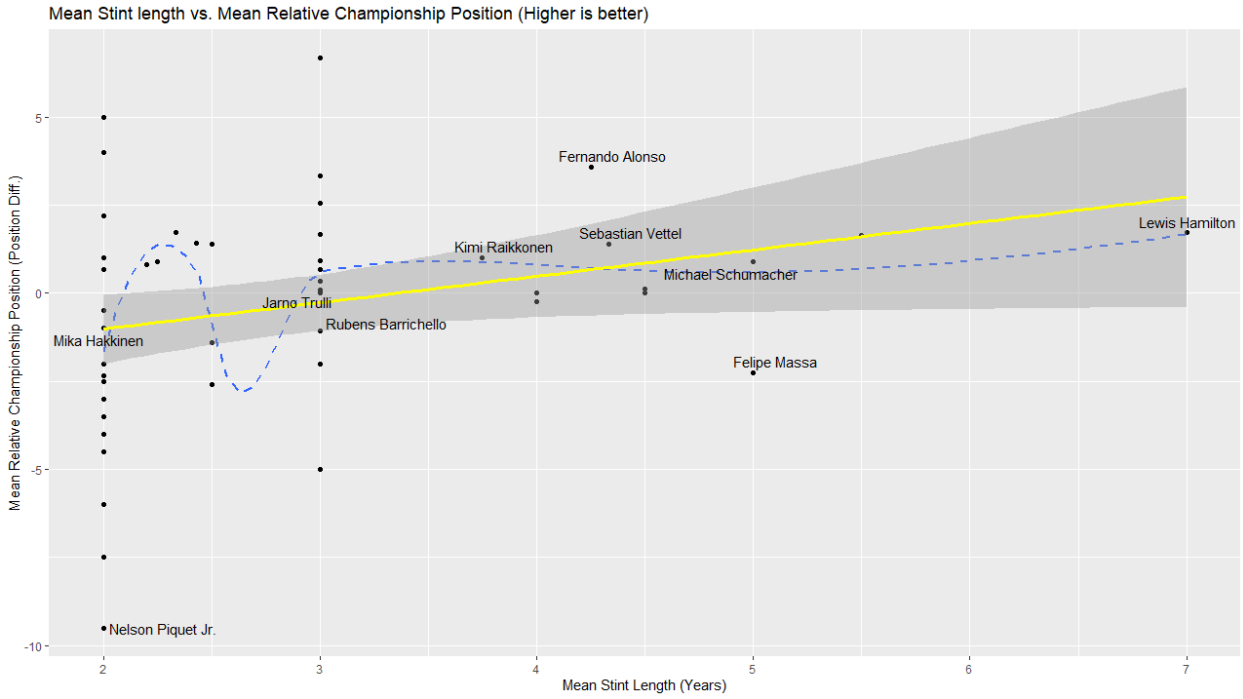
championship taken from under his nose by teammate Kimi Raikkonen that year and then Lewis Hamilton in 2008 (FIA). After that, he partnered a dominant Fernando Alonso, and was forced to play the support act by his team. There is an argument that his results tailed off due to his near-fatal accident in 2009, but this returns into the realm of conjecture as it is hard to quantify, especially with the data collected in this project (Hamilton).

In contrast, Nelson Piquet Jr. is a driver who was unable to shine in Formula One. He ranked lowest (or close to) in all three metrics in a career spanning only two years as a Renault driver and punctuated by the sport's largest race-fixing scandal. He did get some redemption once he moved to Formula E, where he won the title in the sport's first year (Boxall-Legge; FIA).

Sebastian Vettel and Fernando Alonso are two outliers who seem to have had much more success than their stint lengths would indicate. In the case of the former, Sebastian Vettel first dominated teammates in Toro Rosso, and then faced a perfect storm of having the fastest car and being at the peak of his talents when he graduated to Red Bull. This allowed him to win four world championships in a row from 2010-2013. He was sadly unable to reach similar peaks after moving to Ferrari, but easily dominated teammate Kimi Raikkonen who was in the twilight of his career (FIA).

Fernando Alonso's performance hides a more tragic career trajectory. After winning two consecutive titles for Renault in 2005 and 2006, he has been bouncing between teams chasing similar glory but only ever getting as close as second place. Had he had better luck in this regard, the record against his teammates shows how easily he could have been just as successful as Vettel, if not more (FIA).

Finally, there is Lewis Hamilton, a driver who has a uniquely congruent career in Formula One, averaging 7 years per team in the measured period. He epitomizes the driver who works to build a car and team around him to get the best results possible, a philosophy that culminated in him equalling Michael Schumacher's 7 world championship record in 2020 and leading many other statistics outright (FIA). While he has switched teams once, from McLaren to Mercedes, he has always driven with a Mercedes Engine in his car as they were McLaren's engine supplier during his stint at that team. While the relatively weak correlation value of 0.32 means that congruence likely only plays a small part in his extraordinary career, it would have been important in helping him get the most out of the skills, talent and work ethic that also contributed to propelling him to where he is.



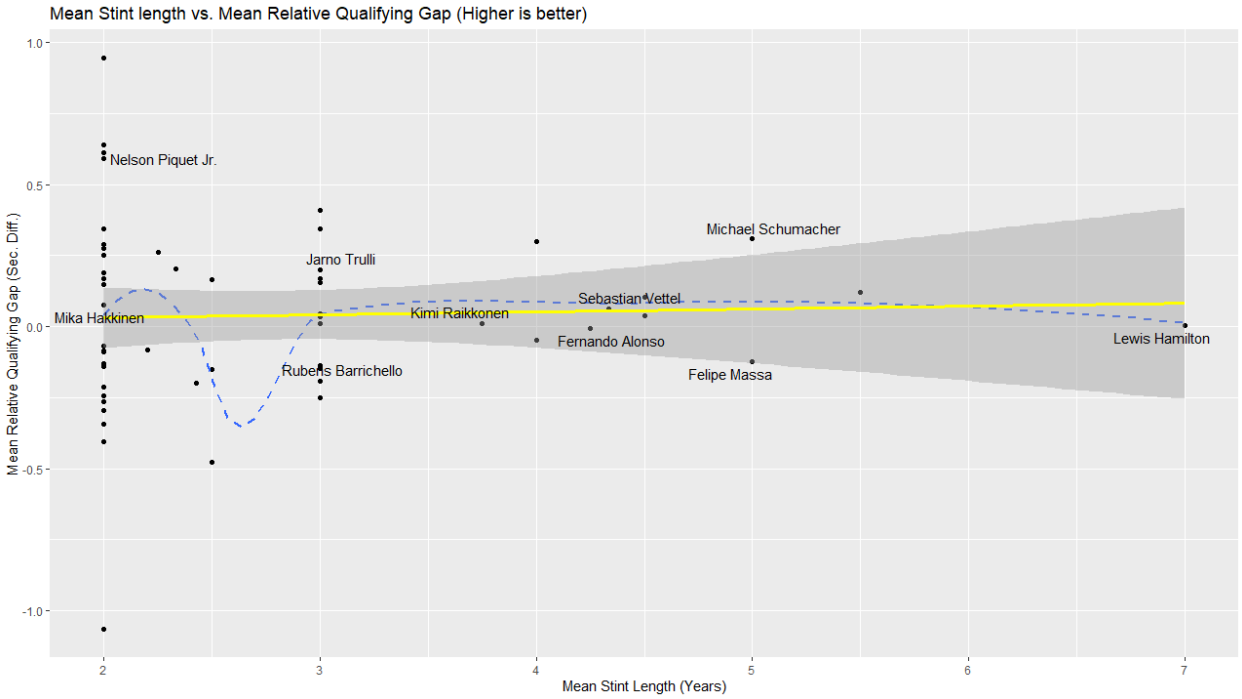
The next graph shows the relationship between mean stint length and mean relative championship position. Like the previous graph, these statistics have a slight positive relationship with a correlation coefficient of 0.28. The "wiggle" in the dotted trend line is also visible for drivers with a mean stint length of three years or less. This also supports the study's hypothesis that job congruence is positively related with driver performance.

Building onto the results of the prior graph, this plot shows that outliers are significantly more outlying than the race result graph. For example, Massa and Alonso have a mean relative finishing position of -1.66 and 2.32 but a mean relative championship position of -2.27 and -3.59 respectively. On an intuitive level, this means that consistently finishing ahead or behind one's teammate will be propagated and amplified into the championship standings.

Interestingly however, this is not the case for Michael Schumacher and Lewis Hamilton, who are the two most successful drivers the sport has seen but fall below the best-fit line. The key difference between them and a driver like Fernando Alonso is the car and teammate they have been paired with. After his stint at Renault where he won his titles, Alonso has only really had a car to win the championship in 2007, (when he was paired with Lewis Hamilton and finished equal on points). Instead, his career has been notable for finishing in high positions through sheer skill and canny race-craft in an arguably slow car (FIA). The best example of this is the 2012 season, when he came second, three points shy of the title but his teammate, Felipe Massa, finished in seventh with less than half of Alonso's points (FIA).

In contrast, both Schumacher and Hamilton had incredibly strong teammates and fast cars for most of their careers. When winning his five consecutive titles, Schumacher was paired with Rubens Barichello who never finished lower than fourth in the standings. However, his return with Mercedes in the 2010s saw him paired with Nico Rosberg, who managed to beat him over their time together, when they didn't have a car capable of winning races let alone titles (FIA). Had this study included the 1990s, Schumacher would have fared much better as he was constantly forcing his Benetton team to replace the second driver since none of them could hold a candle to him (FIA).

While Hamilton was easily the better driver when paired with Heikki Kovalainen for 2008 and 2009, those were the only two "easy" years in his career. His rookie year at McLaren pitted him as Fernando Alonso's teammate right after he won his two titles. Hamilton astoundingly managed to equal Alonso in both points and victories (but was placed ahead due to having more second place finishes) and lost out to Kimi Raikkonen in the championship by just one point (FIA). After that, his career was spent partnering Jenson Button and Nico Rosberg, two very strong drivers. In the case of the latter, Hamilton and Rosberg finished first or second every year in the undisputed fastest car from 2014 to 2016, Hamilton subsequently continued his winning streak with Valtteri Bottas as his teammate (FIA). In all these cases, it is hard to shine in the championship gap statistic, even if you're beating your teammate, if they finish right behind you.



The final plot shows the relationship between mean stint length and mean relative qualifying gap. This graph shows no relationship between the two factors with a correlation coefficient of 0.04, which means that it does not support this project's hypothesis.

Aside from being the one metric where Nelson Piquet Jr. is a standout performer, doing considerably better than the competition, the other highlighted drivers are all close to the mean. Whether they are stronger or weaker than their teammate is consistent with their performance in the last two metrics, and otherwise unremarkable.

One driver worth highlighting is Jarno Trulli. His qualifying performances are one of the best of the pack, with the only drivers who beat him having a considerably shorter career (and therefore less data). He was saddled with a Toyota that allowed him to qualify much higher than he was able to finish at the end of the race. While this normally meant that he would qualify in a flattering position and then regress to a more appropriate position during the race, tracks with few overtaking opportunities would create a "Trulli Train" as drivers in quicker cars would end up stuck behind him (Sportslumo Desk).

Returning to the graph, the main conclusion that can be drawn is that qualifying is a competition where external factors don't impact the drivers as much as races or the championship. While this project doesn't measure enough factors to pinpoint exactly why this might be the case, it can be argued that the short-term nature of qualifying means that it is much easier for the driver to ignore external factors and put in the absolute fastest lap that they can muster on the day.

In this sense, qualifying is distinct from the other measured variables. Over the course of a 300km long race, drivers must contend with much more than just maximizing the performance of the car. They can be beset with reliability woes, strategic mismanagement from the team, or just be forcibly held back from performing to their potential by team politics. Over the course of a whole season, this is further magnified, making it potentially impossible for a driver beat their teammate.

Sebastian Vettel, Lewis Hamilton, Michael Schumacher, and Fernando Alonso are all drivers who earned the first driver position at their respective teams. While this was rarely an overt admission by management, they were clearly favoured in strategy, in getting car upgrades and much better supported psychologically behind the scenes. Sebastian Vettel was at the centre of the "Multi 21" controversy where he disobeyed a team order to remain behind teammate Mark Webber while running second with Webber in first (Vinel). While being clearly in the wrong and receiving near-unanimous condemnation from fans and the world's press, his team ended up protecting him despite being the ones whose orders he disobeyed.

Lewis Hamilton and Fernando Alonso have both been helped on track by their teams asking their respective teammates to move aside for them in races where they were not as fast. This would help them finish higher and therefore score more points to bolster their championship campaign. There is absolutely no clearer sign that a team does not believe you can win them the championship than if you are asked to step aside to let your teammate pass, despite being ahead and presumably faster. It's hard to imagine this does not dampen a driver's confidence and impact their performance.

In contrast, qualifying is a much 'purer' form of competition. There is little strategy or politics, the only way to be advantaged is by having a better car than your competition, which is seldom the case between teammates. As such, it's perhaps unsurprising that qualifying performance is unaffected by a factor such as mean stint length, which can be reframed as a driver's average seniority at a team. There are certain exceptions to this, tracks with long straight-aways such as Monza demand cars with high top speeds, so drivers are usually asked to sacrifice their lap to give their teammate a slipstream. However, situations and tracks that demand such actions are rare, so drivers are generally given equal opportunity to drive the fastest lap that they can.

Conclusion

Overall results

To start with in the conclusion, it might be interesting to address the fact that this study looks at trends over an entire population rather than a sample. There were criteria identified by this study at the beginning, and the graphs above show the relationship between the explanatory and response variables for all 55 drivers that fit these criteria; changing the criteria would change the study.

Purely as a hypothetical, if we assume that Formula One never changes from its current state (which it already has from 2021 with sprint races but never mind that) then the current drivers can serve as a sample for who has and ever will drive in the sport. To see whether this data serves as a meaningful predictor of this sample, mean stint length can be modelled against the three predictor variables in a linear regression equation.

The model has an adjusted R-squared value of 0.05, which is quite small and shows that the predictor variables do not model the response very well. This is backed up by the p-values for each of the predictors themselves, which are all much higher than even an alpha value of 0.1. These data, in addition to the results presented by the graphs above, does not support the starting hypothesis in this study.

However, this does not mean that there is no relationship between job congruity and driver performance in Formula One. Instead, it shows that it alone is not a significant enough factor in determining driver

performance. Formula One is an intense and highly political sport where only twenty drivers can compete at any one time, only a handful of whom will ever get close to winning a race, let alone a world championship. Simply being able to sit on the grid at the start of a grand prix can be considered an accomplishment worth celebrating for a driver; getting to the chequered flag, and doing so in a decent position, is the culmination of so many factors that this study would have to expand significantly in scope to meaningfully cover this subject.

Overall, the results show that not enough drivers have had a career where they've spent more than two to three years at a single team. This means that compared to most drivers, those who stand out are also those who are retained by the teams where they were able to be successful in the first place. This positive feedback loop works to create the small group of hyper-successful drivers and teams that end up dominating the sport for years at a time.

Established Literature

Looking at established psychological literature, a study on the association between job insecurity and job performance across different employment groups showed that while there is a meaningful relationship for permanent employees, contract and part-time workers see only a weak relationship in this regard (van Vuuren et al. 241). This is reflected in this project as well, since Formula One drivers are classed as contractors who must have their employment renewed on a regular basis unlike the other employees at the teams such as mechanics and designers. Furthermore, both this project and the study being referred to also show that regardless of the strength of the association, low job security doesn't help job performance either (van Vuuren et al. 241).

A similar study measured the effects of job congruence on the mental health of sports officials. This study shows that mental health is strongly impacted by job congruence (Sundell 84). While this study looked at sports officials rather than the athletes themselves, if we extrapolate that the similar environment also impacts athletes who are competing in the sport, we can conclude that there must be some positive relationship (perhaps not one as strong) between job congruence and mental health. To support this, Formula One has had several drivers coming forward on the impact of mental health on their ability to compete (F1). Combining these factors allows us to make the leap that there is support for the hypothesis that job congruence does play some part in improving driver performance.

Potential Future Improvements

The first improvement would be to the qualifying gap variable. This variable fails to account for the different track lengths and relative lap-time differences this results in. The Red Bull Ring in Austria is the shortest track on the calendar with a length of 4.3km with a pole position time in 2020 of 64 seconds, and

the Circuit de Spa-Francorchamps is the longest at 7km long with a 2020 pole position time of 102 seconds. If the second placed driver was half a second slower at both grands prix, they would be 0.7% slower in Austria but only 0.4% slower in Belgium. As such, changing the times to be a percentage of pole position and then using that for the teammate comparison would normalise this statistic and be a better reflection of qualifying performance.

Focusing on the entire project, the first improvement would be to simply expand the dataset to all drivers from 1981 to the present, increasing the density of data available to all the years when teams could run two cars concurrently.

Second, since there is such a skew in mean stint lengths, it could be interesting to reframe the study as a comparative analysis. Driver performance could be compared between the drivers who have a mean stint length of less than three years versus those who average more than three years as a team (for example). This would dilute individual drivers' accomplishments but allow for a better comparison between low and high job congruity.

This project was quite a limited exercise when compared to the field of sport and professional psychology. As such, the primary way to improve this study would be to combine the quantitative approach with qualitative metrics that can provide more substantive information about the factors that impact performance.

Looking at the quantitative side, one way to expand this study would be to adapt the variables to include more different motorsports with varying social environments and cultures. While Formula One serves as the pinnacle of motorsport, beneath it are numerous championships with countless drivers also competing to be the best in their class. For example, an interesting study could look at the factors that impact driver performance in the championships that lead up to Formula One, such as Formula Two and Formula Three.

Works Cited

Baldwin, Alan. "Canadian Group Led by Stroll Paid $117 Million for Force India." *Reuters*, Thomson Reuters, 4 Oct. 2018, www.reuters.com/article/uk-motor-f1-forceindia-idUKKCN1ME1LV.

Boxall-Legge, Jake. "The Biggest Incidents of F1 Cheating: Spygate, Crashgate and More." *Motorsport.com: F1 News, MotoGP, NASCAR, Rallying and More*, 10 June 2021, www.motorsport.com/f1/news/f1-cheating-spygate-crashgate/6555686/.

F1. "McLaren's Lando Norris on Why He's 'Happy and Confident' to Be a Mental Health Pioneer in Formula 1: Formula 1®." *Formula 1*, Formula 1, 24 Jan. 2022,

www.formula1.com/en/latest/article.norris-on-why-hes-happy-and-confident-to-be-a-mental-health-pioneer-in.3z6ZRG9z35otsGUIn2ZHb.html.

FIA Championships Results & Statistics, motorsportstats.com, 2022, fiaresultsandstatistics.motorsportstats.com/results/.

Hamilton, Maurice. "Felipe Massa Suffers Skull Fracture in Freak Accident during Qualifying." *The Guardian*, Guardian News and Media, 25 July 2009, www.theguardian.com/sport/2009/jul/25/felipe-massa-fernando-alonso-hungarian.

Sportslumo Desk. "Jarno Trulli - One-Time Formula 1 Race Winner and Maestro of Qualifying." *Sportslumo*, 13 July 2021, sportslumo.com/motorsport/jarno-trulli-one-time-formula-1-race-winner-and-maestro-of-qualifying/.

Sundell, Paul J. "The Relationship of Job Congruence, Job Satisfaction, and Social Support on the Mental and Physical Health of Sport Officials." *West Virginia University* , ProQuest Dissertations Publishing, 1999, pp. 1–130.

Taylor, Michael. "Ron Dennis Exits Mclaren and Formula One Doesn't Mourn, but It Should." *Forbes*, Forbes Magazine, 3 July 2017, www.forbes.com/sites/michaeltaylor/2017/07/03/ron-dennis-exits-mclaren-and-formula-one-doesnt-mourn-but-it-should/?sh=558c0a276623.

Tran, Mark. "Red Bull Buys Jaguar F1 Team." *The Guardian*, Guardian News and Media, 15 Nov. 2004, www.theguardian.com/business/2004/nov/15/formulaone.money.

Van Vuuren, Tinka, et al. "The Association between Subjective Job Insecurity and Job Performance across Different Employment Groups." *Career Development International*, vol. 25, no. 3, 2019, pp. 229–246., doi:10.1108/cdi-05-2018-0155.

Vinel, Benjamin. "Multi 21: Flash Back to Red Bull's Most Controversial Race." *Motorsport.com: F1 News, MotoGP, NASCAR, Rallying and More*, Motorsport.com - Global, 24 Mar. 2022, www.motorsport.com/f1/news/multi-21-webber-vettel-red-bull/4772676/.

WilliamsF1. "Williams Racing Is Acquired by Dorilton Capital." *Williams Racing*, 20 Aug. 2020, www.williamsf1.com/posts/46e9fd54-c1b0-4634-a683-dfc95987c668/williams-racing-is-acquired-by-dorilton-capital.

Appendix

Race Result Web Scraper Code (RaceResult_Extractor.py)

```
from requests import *
from bs4 import BeautifulSoup
import tkinter as tk
from tkinter import *
import os
import time

def ExtractPage(ID, year, race, roundNo):
    # CreateURL and access webpage
    url = "https://fiaresultsandstatistics.motorsportstats.com/results/"
    raceURL = url + str(year) + "-" + race.replace(' ', '-').lower() +
"/classification"

    resp = get(raceURL)

    if resp.status_code == 200:
        soup = BeautifulSoup(resp.text, "html.parser")  # Success debug
message

        # Create data file
        data = soup.find_all("td", {"class": "_2sWDi"})
        filename = os.path.dirname(__file__) + "\\" + str(ID) + "-" +
str(roundNo) + "-" + str(year) + "-" + race.replace(' ', '-') + ".txt"
        newFile = open(filename, 'w')


newFile.write("ID\tRoundNo\tYear\tRace\tPos\tNo\tDriver\tNat\tTeam\tLaps\tTim
e\tGap\tInterval\tKph\tBest\tLap\n")

        n = 0


        try:  # Checks if the user entered a valid Grand Prix and then writes
its results onto a text file
            pos = 0
            for i in data:
                datum = i.text
                #print(i.string)
                if n == 0:
                    newFile.write(str(ID) + "\t" + str(roundNo) + "\t" +
str(year) + "\t" + race + "\t")
                    pos = pos + 1
                    datum = pos
                    newFile.write(str(datum) + "\t")
                    n = n + 1
                elif n < 11:
                    if(n == 4):
                        datum = SetConstructorName(datum)
                    newFile.write(str(datum) + "\t")
                    n = n + 1
                else:
                    newFile.write(str(datum) + "\n")
```

```
                    n = 0

            newFile.close()
            print("Done - File Ready")

        except Exception as e:
            newFile.close()
            if os.path.exists(filename):
                os.remove(filename)


            print(year, race, "does not exist or\n", e, )

    else:
        print("Code", resp.status_code, "Accessing FIA Website: FAILED")  #
Failure debug message

def BatchExtractor(start, end):
    startYear = start
    endYear = end

    n = 0

    for i in range(startYear, endYear + 1):
        baseURL =
"https://fiaresultsandstatistics.motorsportstats.com/results/" + str(i) + "-
british-grand-prix/classification/"

        resp = get(baseURL)

        if resp.status_code == 200:
            print("Code", resp.status_code, "Accessing FIA Website: SUCCESS")
            soup = BeautifulSoup(resp.text, "html.parser")  # Success debug
message

            # Create data file
            data = soup.find_all("a", {"class": "_3AdU8"})
            try:
                k = 1
                for j in data:
                    j=j.string
                    if "Season Test" not in j:
                        print("Accessing Data for the", i, j)
                        ExtractPage(n, i, j, k)
                        time.sleep(0)
                        n = n + 1
                        k = k + 1

            except:
                print("Error")

        print("Loading the", i+1, "season...")
        time.sleep(0)


def SetConstructorName(Name):
    if ("Midland" in Name):
```

```
        return "Nidland"
    if ("Jordan" in Name):
        return "Jordan"
    if ("Brawn" in Name):
        return "Brawn"
    if ("Super Aguri" in Name):
        return "Super Aguri"
    if ("BAR" in Name):
        return "BAR"
    if ("Virgin" in Name):
        return "Virgin"
    if ("Manor" in Name):
        return "Manor"
    if ("Marussia" in Name):
        return "Marussia"
    if ("Caterham" in Name):
        return "Caterham"
    if ("Spyker" in Name):
        return "Spyker"
    if ("HRT" in Name):
        return "HRT"
    if ("Prost" in Name):
        return "Prost"
    if ("Minardi" in Name):
        return "Minardi"
    if ("Benneton" in Name):
        return "Benneton"
    if("Ferrari" in Name):
        return "Ferrari"
    if("McLaren" in Name):
        return "McLaren"
    if("Williams" in Name):
        return "Williams"
    if("Toyota" in Name):
        return "Toyota"
    if("Haas" in Name):
        return "Haas"
    if("Force India" in Name):
        return "Force India"
    if("Alfa Romeo" in Name or "Sauber" in Name):
        return "Sauber"
    if ("AlphaTauri" in Name or "Toro Rosso" in Name):
        return "Toro Rosso"
    if ("Red Bull" in Name):
        return "Red Bull"
    if("Racing Point" in Name or "Aston Martin" in Name):
        return "Aston Martin"
    if("Tyrrel" in Name):
        return "Tyrrel"
    if("Ligier" in Name):
        return "Ligier"
    if("Footwork" in Name):
        return "Footwork"
    if("Honda" in Name):
        return "Honda"
    if("Mercedes" in Name):
        return "Mercedes"
```

```
    if("Benetton" in Name):
        return "Benneton"
    if("Alpine" in Name or "Renault" in Name):
        return "Renault"
    if ("Lotus" in Name):
        return "Lotus"
    if("Arrows" in Name):
        return "Arrows"
    if("Lola" in Name):
        return "Lola"
    if("March" in Name):
        return "March"
    if("Jaguar" in Name):
        return "Jaguar"
    if("Larousse" in Name):
        return "Larousse"
    else:
        return Name



#Creating the GUI
root = tk.Tk()
root.title("Race Result Generation Utility")
#root.geometry("430x150")
root.resizable(False, False)

frame = tk.Frame(root)
options = {'padx': 5, 'pady': 5}

yearTitle = tk.Label(frame, text = "Year")
yearTitle.grid(column = 1, row = 0, sticky = 'S', **options)
raceTitle = tk.Label(frame, text = "Race")
raceTitle.grid(column = 2, row = 0, sticky = 'S', **options)

singleRace = tk.Label(frame, text = "Single Race")
singleRace.grid(column = 0, row = 1, sticky = 'W', **options)
SRYear = tk.IntVar()
SRYEntry = tk.Entry(frame, textvariable = SRYear)
SRYEntry.grid(column = 1, row = 1, **options)
SRRace = tk.StringVar()
SRREntry = tk.Entry(frame, textvariable = SRRace)
SRREntry.grid(column = 2, row = 1, **options)
SRButton = tk.Button(frame, text="Parse")
SRButton.grid(column = 3, row = 1, **options)
SRButton.configure(command = lambda: ExtractPage(-1, SRYear.get(),
SRRace.get(), 0))

singleYear = tk.Label(frame, text = "Single Year")
singleYear.grid(column = 0, row = 2, sticky = 'W', **options)
SYYear = tk.IntVar()
SYYEntry = tk.Entry(frame, textvariable = SYYear)
SYYEntry.grid(column = 1, row = 2, **options)
SYButton = tk.Button(frame, text="Parse")
SYButton.grid(column = 3, row = 2, **options)
SYButton.configure(command = lambda: BatchExtractor(SYYear.get(),
SYYear.get()))
```

```
startTitle = tk.Label(frame, text = "First Year")
startTitle.grid(column = 1, row = 3, sticky = 'S', **options)
endTitle = tk.Label(frame, text = "Last Year")
endTitle.grid(column = 2, row = 3, sticky = 'S', **options)

multiYear = tk.Label(frame, text = "Multiple Years")
multiYear.grid(column = 0, row = 4, sticky = 'W', **options)
MYStart = tk.IntVar()
MYSEntry = tk.Entry(frame, textvariable = MYStart)
MYSEntry.grid(column = 1, row = 4, **options)
MYEnd = tk.IntVar()
MYEEntry = tk.Entry(frame, textvariable = MYEnd)
MYEEntry.grid(column = 2, row = 4, **options)
MYButton = tk.Button(frame, text="Parse")
MYButton.grid(column = 3, row = 4, **options)
MYButton.configure(command = lambda: BatchExtractor(MYStart.get(),
MYEnd.get()))

frame.grid(padx = 10, pady = 10)
root.mainloop()
```

Qualifying Result Web Scraper Code (QualiResult_Extractor.py)

```
from requests import *
from bs4 import BeautifulSoup
import tkinter as tk
from tkinter import *
import os
import time

def ExtractPage(ID, year, race, roundNo):
    # CreateURL and access webpage
    url = "https://fiaresultsandstatistics.motorsportstats.com"
    raceURL = url + "/results/" + str(year) + "-" + race.replace(' ', '-
').lower() + "/classification"
    print(raceURL)

    resp = get(raceURL)

    if resp.status_code == 200:
        soup = BeautifulSoup(resp.text, "html.parser")  # Success debug
message

        #Get to the qualifying results page
        data = soup("a", {"class": "uaJW4 _2j84j"})
        for i in data:
            if i.text == "Grid" or i.text == "Qualifying": #Some races only
have one or the other
                qualiURL = url + i.get('href')
                print("Using", i.text)
                break
        resp = get(qualiURL)
        soup = BeautifulSoup(resp.text, "html.parser")

        #Access qualifying/grid table and create its file
        data = soup("td", {"class": "_2sWDi"})
        filename = os.path.dirname(__file__) + "\\" + str(ID) + "-" +
```

```
str(roundNo) + "-" + str(year) + "-" + race.replace(' ', '-') + ".txt"
        newFile = open(filename, 'w')

newFile.write("ID\tRoundNo\tYear\tRace\tPos\tNo\tDriver\tNat\tTeam\tLaps\tTim
e\tGap\tInterval\tKph\tBest\tLap\n")

        #Write to file
        n = 0
        try:
            for i in data:
                if n == 0:
                    newFile.write(str(ID) + "\t" +
                                  str(roundNo) + "\t" +
                                  str(year) + "\t" +
                                  race + "\t")
                    newFile.write(i.text + "\t")
                    n = n + 1
                elif n < 11:
                    datum = i.text
                    if "+" in datum:
                        newFile.write( datum[2:] + "\t")
                    else:
                        newFile.write(datum + "\t")
                    n = n + 1
                else:
                    newFile.write(i.text + "\n")
                    n = 0

            newFile.close()
            print("Done - File Ready")

        except:
            newFile.close()
            if os.path.exists(filename):
                os.remove(filename)
            print("Race does not exist")

    else:
        print("Code", resp.status_code, "Accessing FIA Website: FAILED")  #
Failure debug message

def BatchExtractor(start, end):
    startYear = start
    endYear = end

    n = 0

    for i in range(startYear, endYear + 1):
        baseURL =
"https://fiaresultsandstatistics.motorsportstats.com/results/" + str(i) + "-
british-grand-prix/classification/"

        resp = get(baseURL)

        if resp.status_code == 200:
            print("Code", resp.status_code, "Accessing FIA Website: SUCCESS")
            soup = BeautifulSoup(resp.text, "html.parser")  # Success debug
```

```
message

            # Create data file
            data = soup.find_all("a", {"class": "_3AdU8"})
            try:
                k = 1
                for j in data:
                    j=j.string
                    if "Season Test" not in j:
                        print("Accessing Data for the", i, j)
                        ExtractPage(n, i, j, k)
                        time.sleep(0)
                        n = n + 1
                        k = k + 1

            except:
                print("Error in BatchExtractor")

        print("Loading the", i+1, "season...")
        time.sleep(0)
    return(n)

#Creating the GUI
root = tk.Tk()
root.title("Qualifying Result Generation Utility")
#root.geometry("430x150")
root.resizable(False, False)

frame = tk.Frame(root)
options = {'padx': 5, 'pady': 5}

yearTitle = tk.Label(frame, text = "Year")
yearTitle.grid(column = 1, row = 0, sticky = 'S', **options)
raceTitle = tk.Label(frame, text = "Race")
raceTitle.grid(column = 2, row = 0, sticky = 'S', **options)

singleRace = tk.Label(frame, text = "Single Session")
singleRace.grid(column = 0, row = 1, sticky = 'W', **options)
SRYear = tk.IntVar()
SRYEntry = tk.Entry(frame, textvariable = SRYear)
SRYEntry.grid(column = 1, row = 1, **options)
SRRace = tk.StringVar()
SRREntry = tk.Entry(frame, textvariable = SRRace)
SRREntry.grid(column = 2, row = 1, **options)
SRButton = tk.Button(frame, text="Parse")
SRButton.grid(column = 3, row = 1, **options)
SRButton.configure(command = lambda: ExtractPage(-1, SRYear.get(),
SRRace.get(), 0))

singleYear = tk.Label(frame, text = "Single Year")
singleYear.grid(column = 0, row = 2, sticky = 'W', **options)
SYYear = tk.IntVar()
SYYEntry = tk.Entry(frame, textvariable = SYYear)
SYYEntry.grid(column = 1, row = 2, **options)
SYButton = tk.Button(frame, text="Parse")
SYButton.grid(column = 3, row = 2, **options)
SYButton.configure(command = lambda: BatchExtractor(SYYear.get(),
```

```
SYYear.get()))

startTitle = tk.Label(frame, text = "First Year")
startTitle.grid(column = 1, row = 3, sticky = 'S', **options)
endTitle = tk.Label(frame, text = "Last Year")
endTitle.grid(column = 2, row = 3, sticky = 'S', **options)

multiYear = tk.Label(frame, text = "Multiple Years")
multiYear.grid(column = 0, row = 4, sticky = 'W', **options)
MYStart = tk.IntVar()
MYSEntry = tk.Entry(frame, textvariable = MYStart)
MYSEntry.grid(column = 1, row = 4, **options)
MYEnd = tk.IntVar()
MYEEntry = tk.Entry(frame, textvariable = MYEnd)
MYEEntry.grid(column = 2, row = 4, **options)
MYButton = tk.Button(frame, text="Parse")
MYButton.grid(column = 3, row = 4, **options)
MYButton.configure(command = lambda: BatchExtractor(MYStart.get(),
MYEnd.get()))

frame.grid(padx = 10, pady = 10)
root.mainloop()
```

Champion Ship Result Web Scraper Code (Championship_Extractor.py)

```
from requests import *
from bs4 import BeautifulSoup
from tkinter import *
import os
import time

def ExtractPage(year, race):
    # CreateURL and access webpage
    url = "https://fiaresultsandstatistics.motorsportstats.com/results/"
    raceURL = url + str(year) + "-" + race.replace(' ', '-').lower() +
"/standings"

    resp = get(raceURL)

    if resp.status_code == 200:
        soup = BeautifulSoup(resp.text, "html.parser")  # Success debug
message

        # Create data file
        data = soup.find("table", {"class": "_2Q90P"})
        filename = os.path.dirname(__file__) + "\\" + str(year) + "-" +
"Championship-Standings" + ".txt"
        newFile = open(filename, 'w')

        newFile.write("Year\tPos\tDriver\tPoints\n")

        n = 0
        try:  # Checks if the user entered a valid Grand Prix and then writes
its results onto a text file
            for i in data.findAll("td"):
                if n == 0:
                    newFile.write(str(year) + "\t" + i.text + "\t")
                    n = n + 1
```

```
                elif n < 2:
                    newFile.write(i.text + "\t")
                    n = n + 1
                else:
                    newFile.write(i.text + "\n")
                    n = 0

            newFile.close()
            print("Done - File Ready")

        except:
            newFile.close()
            if os.path.exists(filename):
                os.remove(filename)
            print(year, race, "does not exist")

    else:
        print("Code", resp.status_code, "Accessing FIA Website: FAILED")  #
Failure debug message

def BatchExtractor():
    startYear = 2020
    endYear = 2019

    n = 0

    for i in range(endYear, startYear + 1):
        baseURL =
"https://fiaresultsandstatistics.motorsportstats.com/results/" + str(i) + "-
british-grand-prix/classification/"

        resp = get(baseURL)

        if resp.status_code == 200:
            print("Code", resp.status_code, "Accessing FIA Website: SUCCESS")
            soup = BeautifulSoup(resp.text, "html.parser")  # Success debug
message

            # Create data file
            data = soup.find_all("a", {"class": "_3AdU8"})
            try:
                    ExtractPage(i, data[-2].text)
                    time.sleep(0)
                    n = n + 1

            except:
                print("Error")

        print("Loading the", i+1, "season...")
        time.sleep(0)
    return(n)

print(BatchExtractor(), "season standings successfully saved")
```

Race and Qualifying Result Relative-iser (PivotWiderButActuallyUseful.py)

```
import os
```

```
class TeamResult: #Result object
    ID = ''
    Driver1 = ''
    Driver2 = ''
    PosD1 = 0
    PosD2 = 0
    GapD1 = 0
    GapD2 = 0

resultList = []

def TranslateToResult(result): #Take table line and turn it into  result
object
    curSnippet = result[result.find('\t')+1:] #Trimming row number

    id = curSnippet[:curSnippet.find('\t')]
    curSnippet = curSnippet[curSnippet.find('\t')+1:]

    driver = curSnippet[:curSnippet.find('\t')]
    curSnippet = curSnippet[curSnippet.find('\t')+1:]

    pos = curSnippet[:curSnippet.find('\t')]
    curSnippet = curSnippet[curSnippet.find('\t')+1:]

    gap = curSnippet

    elementExists = False
    existingElementIndex = 0


    if len(resultList) == 0: #Create first element if list is empty
        #print("List is empty, making first element")
        resultList.append(TeamResult())
        resultList[0].ID = id
        resultList[0].Driver1 = driver
        resultList[0].PosD1 = pos
        resultList[0].GapD1 = gap
    else: #Find index if team exists for current result
        #print("Searching for existing result")
        for i in range(0, len(resultList)):
            if resultList[i].ID == id:
                elementExists = True
                existingElementIndex = i
                break

    if elementExists: #Adding second driver details to result
        #print("Existing result found, entering data for 2nd driver")
        resultList[existingElementIndex].Driver2 = driver
        resultList[existingElementIndex].PosD2 = pos
        resultList[existingElementIndex].GapD2 = gap
    else: #Creating new result entry with first driver details
        #print("Creating new result")
        resultList.append(TeamResult())
        lastIndex = len(resultList)-1
        resultList[lastIndex].ID = id
        resultList[lastIndex].Driver1 = driver
        resultList[lastIndex].PosD1 = pos
```

```
        resultList[lastIndex].GapD1 = gap


data = open("RQ_Combined.txt", "r").read() #reading data file


for i in range(0, data.count('\n')): #traversing data file
    print(i)
    TranslateToResult(data[:data.find('\n')])
    data = data[data.find('\n')+1:]

filename = os.path.dirname(__file__) + "\\" + "ActuallyWideResults" + ".txt"
newFile = open(filename, 'w')
newFile.write("ID\tDriver1\tDriver2\tPosDriver1\tPosDriver2\tQualiDriver1\tQu
aliDriver2\n")

for i in resultList: #Rewriting widened results as relative gaps in the same
row
    newFile.write(i.ID + "\t")
    newFile.write(i.Driver1 + "\t")
    newFile.write(i.Driver2 + "\t")
    newFile.write(str(int(i.PosD2) - int(i.PosD1)) + "\t")
    newFile.write(str(int(i.PosD1) - int(i.PosD2)) + "\t")
    newFile.write(str(round(float(i.GapD2) - float(i.GapD1), 3)) + "\t")
    newFile.write(str(round(float(i.GapD1) - float(i.GapD2), 3)) + "\n")

filename = os.path.dirname(__file__) + "\\" + "SingleDriverRelative" + ".txt"
newFile = open(filename, 'w')
newFile.write("ID\tDriver\tRelativePos\tRelativeQuali\n")

for i in resultList: #Rewriting widened results as relative gaps with each
driver in their own row
    newFile.write(i.ID + "\t")
    newFile.write(i.Driver1 + "\t")
    newFile.write(str(int(i.PosD2) - int(i.PosD1)) + "\t")
    newFile.write(str(round(float(i.GapD2) - float(i.GapD1), 3)) + "\n")

    newFile.write(i.ID + "\t")
    newFile.write(i.Driver2 + "\t")
    newFile.write(str(int(i.PosD1) - int(i.PosD2)) + "\t")
    newFile.write(str(round(float(i.GapD1) - float(i.GapD2), 3)) + "\n")
```

Championship Result Relative-iser (PwbauStandings.py)

```
import os

class TeamResult:
    ID = ''
    Driver1 = ''
    Driver2 = ''
    PosD1 = 0
    PosD2 = 0
    CPD1 = 0
    CPD2 = 0

resultList = []

def TranslateToResult(result): #Take table line and turn it into  result
```

```
object
    curSnippet = result #Trimming row number

    id = curSnippet[:curSnippet.find('\t')]
    curSnippet = curSnippet[curSnippet.find('\t')+1:]

    driver = curSnippet[:curSnippet.find('\t')]
    curSnippet = curSnippet[curSnippet.find('\t')+1:]

    pos = curSnippet[:curSnippet.find('\t')]
    curSnippet = curSnippet[curSnippet.find('\t')+1:]

    cp = curSnippet

    elementExists = False
    existingElementIndex = 0

    if len(resultList) == 0: #Create first element if list is empty
        #print("List is empty, making first element")
        resultList.append(TeamResult())
        resultList[0].ID = id
        resultList[0].Driver1 = driver
        resultList[0].PosD1 = pos
        resultList[0].CPD1 = cp
    else: #Find index if team exists for current result
        #print("Searching for existing result")
        for i in range(0, len(resultList)):
            if resultList[i].ID == id:
                elementExists = True
                existingElementIndex = i
                break

    if elementExists: #Adding second driver details to result
        #print("Existing result found, entering data for 2nd driver")
        resultList[existingElementIndex].Driver2 = driver
        resultList[existingElementIndex].PosD2 = pos
        resultList[existingElementIndex].CPD2 = cp
    else: #Creating new result entry with first driver details
        #print("Creating new result")
        resultList.append(TeamResult())
        lastIndex = len(resultList)-1
        resultList[lastIndex].ID = id
        resultList[lastIndex].Driver1 = driver
        resultList[lastIndex].PosD1 = pos
        resultList[lastIndex].CPD1 = cp

data = open("cut_standings.txt", "r").read() #reading data file


for i in range(0, data.count('\n')): #traversing data file
    TranslateToResult(data[:data.find('\n')])
    data = data[data.find('\n')+1:]

filename = os.path.dirname(__file__) + "\\" + "RelativeStandings" + ".txt"
newFile = open(filename, 'w')
newFile.write("ID\tDriver\tRelativePos\tRelative_CP\n")
```

```
for i in resultList: #Rewriting widened results as relative gaps with each
driver in their own row
    newFile.write(i.ID + "\t")
    newFile.write(i.Driver1 + "\t")
    newFile.write(str(int(i.PosD2) - int(i.PosD1)) + "\t")
    newFile.write(str(round(float(i.CPD1) - float(i.CPD2), 3)) + "\n")

    newFile.write(i.ID + "\t")
    newFile.write(i.Driver2 + "\t")
    newFile.write(str(int(i.PosD1) - int(i.PosD2)) + "\t")
    newFile.write(str(round(float(i.CPD2) - float(i.CPD1), 3)) + "\n")
```

R Code (Import_Result_Data.R)

```
library(tidyverse)
library(ggrepel)

#Find the files in the project directory
standing_files <- list.files(path = "./Championship Standings", pattern =
".txt", full.names = T)
race_files <- list.files(path = "./Race Results", pattern = ".txt",
full.names = T)
quali_files <- list.files(path = "./Quali Results", pattern = ".txt",
full.names = T)

#Read them into the software
standings_results <- lapply(standing_files, read_tsv)
race_results <- lapply(race_files, read_tsv)
quali_results <- lapply(quali_files, read_tsv)

#Binding data into a single table
standings_results %>% bind_rows() -> standings_results
race_results <- do.call("rbind", race_results)
quali_results <- do.call("rbind", quali_results)

#Remove unnecessary variables
race_results %>% select(ID:Team) %>% select(!c(No, Nat)) -> race_results
quali_results %>% select(ID:Gap) %>% select(!c(Nat, Pos, No, Laps, Time,
Team)) -> quali_results

#modify standings table to extract data from race table
standings_results$Points %>% replace_na(0) -> standings_results$Points
standings_results %>% rename(C_Points = Points) -> standings_results

#Combine data
race_results %>% left_join(quali_results, by = c("Year", "Race", "Driver")) -
> RQ_Combined
race_results %>% left_join(standings_results, by = c("Year", "Driver")) ->
standings_results

#Manipulate combined race table
RQ_Combined %>% rename(ID = ID.x, RoundNo = RoundNo.x, QualiGap = Gap) ->
RQ_Combined #Rename the cols
RQ_Combined$QualiGap %>% replace_na(0) -> RQ_Combined$QualiGap #Set gap to 0
for polesitters
RQ_Combined %>% drop_na(ID.y, RoundNo.y) -> RQ_Combined #Remove rows with
missing data
```

```
RQ_Combined %>% select(!c(ID.y, RoundNo.y)) -> RQ_Combined #Remove quali ID
cols
RQ_Combined %>% mutate(across(QualiGap, as.double)) -> RQ_Combined #Turn
quali positions to doubles

#Manipulate combined standings table
standings_results %>% select(c(Year, Driver, Team, Pos.y, C_Points)) %>%
rename(Pos = Pos.y) -> standings_results
standings_results %>% drop_na(Pos, C_Points) -> standings_results
standings_results %>% distinct(Year, Driver, Team, Pos, C_Points) ->
standings_results
standings_results %>% relocate(Team, .before = Driver) -> standings_results
standings_results %>% arrange(Team) %>% arrange(Year) -> standings_results

#Create unified ID
RQ_Combined %>% mutate(RaceID = str_c(Year, "-", Race, "-", Team)) %>%
  select(!c(ID, RoundNo, Year, Race, Team))-> RQ_Combined

RQ_Combined  %>% relocate(RaceID, .before = Pos) %>%
  relocate(Driver, .before = Pos) %>%
  arrange(RaceID) -> RQ_Combined

#Export table to python
write.table(RQ_Combined, file = 'RQ_Combined.txt', quote=FALSE, sep='\t',
col.names = F)

#Import properly widened table back in
widened_results <- read_tsv("ActuallyWideResults.txt")
relative_results <- read_tsv("SingleDriverRelative.txt")

relative_results %>% group_by(Driver) %>%
  summarise(Number_of_Races = n(),
            Mean_Relative_Finishing_Position = mean(RelativePos),
            Mean_Relative_Qualifying_Gap = mean(RelativeQuali)) -> RQ_summary

#Drivers with fewer than 22 races in their career are cut off
RQ_summary %>% arrange(Number_of_Races) %>% filter(Number_of_Races >= 22) ->
RQ_summary

#Removing reserve drivers and establishing which team a driver started the
season in
standings_results %>% arrange(Pos) %>% arrange(Team) %>% arrange(Year) ->
standings_results
standings_results %>% mutate(ID = row_number()) %>% relocate(ID, .before =
Year) -> standings_results

standings_results %>% slice(-c(11, 34, 37, 38, 43, 44, 46, 47, 71, 78, 83,
86, 91, 100, 109, 116, 119,
                               122, 125, 126, 135, 136, 139, 142, 151, 158,
163, 168, 173, 174,
                               193, 195, 199, 206, 233, 234, 237, 244, 249,
252, 261, 262, 275,
                               288, 291, 300, 303, 320, 343, 360, 389, 409,
413, 417, 425, 440, 445, 448,
                               449, 484, 489, 496, 501, 506, 517)) ->
cut_standings
```

```
#Exporting standings table to generate relative results
cut_standings %>% mutate(ID = str_c(Year, "-", Team)) %>% select(c(ID,
Driver, Pos, C_Points)) -> cut_standings

write.table(cut_standings, file = 'cut_standings.txt', quote = F, sep = '\t',
row.names = F, col.names = F)

#Reimporting relative standings table
relative_standings <- read_tsv("RelativeStandings.txt")

#Generating means for driver standing stats
relative_standings %>% group_by(Driver) %>%
  summarise(Number_of_Seasons = n(),
            Mean_Relative_Championship_Position = mean(RelativePos),
            Mean_Relative_Points = mean(Relative_CP)) ->
Relative_standings_summary

#Combining race & quali summaries with standings summary
RQ_summary %>% left_join(Relative_standings_summary, by = c("Driver")) ->
RQS_summary
RQS_summary %>% relocate(Number_of_Seasons, .after = Number_of_Races) ->
RQS_summary

#Get the number of years a driver has been at a time
cut_standings %>%
  separate(ID, c("Year", "Team"), sep = '-') %>%
  mutate(Partnership = str_c(Team, "-", Driver)) %>%
  group_by(Partnership) %>% summarise(n = n()) -> stint_summary

#Will need some manual editing
stint_summary %>%
  separate(Partnership, c("Team", "Driver"), sep = "-") -> stint_summary
write.table(stint_summary, file = "stintsummary.txt", quote = F, sep = '\t',
row.names = F, col.names = T)

#bring back into R for more manual editing
stint_fixed_summary <- read_tsv("stintsummary.txt")
stint_fixed_summary %>% mutate(ID = row_number()) %>% relocate(ID, .before =
Team) -> stint_fixed_summary

stint_fixed_summary %>% rows_update(tibble(ID = 6, Team = "Aston Martin",
Driver = "Sergio P<e9>rez", n = 2)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 7, Team = "BAR", Driver =
"Jacques Villeneuve", n = 4)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 26, Team = "Ferrari", Driver
= "Kimi R<e4>ikk<f6>nen", n = 8)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 33, Team = "Force India",
Driver = "Nico H<fc>lkenberg", n = 4)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 35, Team = "Force India",
Driver = "Sergio P<e9>rez", n = 5)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 37, Team = "Haas", Driver =
"Esteban Guti<e9>rrez", n = 1)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 64, Team = "Lotus", Driver =
"Kimi R<e4>ikk<f6>nen", n = 2)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 82, Team = "McLaren", Driver
= "Kimi R<e4>ikk<f6>nen", n = 2)) -> stint_fixed_summary
```

```
stint_fixed_summary %>% rows_update(tibble(ID = 86, Team = "McLaren", Driver
= "Sergio P<e9>rez", n = 1)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 130, Team = "Renault", Driver
= "Nico H<fc>lkenberg", n = 3)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 136, Team = "Sauber", Driver
= "Esteban Guti<e9>rrez", n = 2)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 143, Team = "Sauber", Driver
= "Kimi R<e4>ikk<f6>nen", n = 3)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 147, Team = "Sauber", Driver
= "Nico H<fc>lkenberg", n = 1)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 152, Team = "Sauber", Driver
= "Sergio P<e9>rez", n = 2)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 168, Team = "Toro Rosso",
Driver = "S<e9>bastien Buemi", n = 3)) -> stint_fixed_summary
stint_fixed_summary %>% rows_update(tibble(ID = 194, Team = "Williams",
Driver = "Nico H<fc>lkenberg", n = 1)) -> stint_fixed_summary

#Get drivers' average stint lengths
stint_fixed_summary %>% group_by(Driver) %>% summarise(Mean_Stint_Length =
mean(n)) -> stint_fixed_summary
RQS_summary %>% left_join(stint_fixed_summary, by = "Driver") -> RQS_summary

#Remove drivers who have an average stint length of less than 2 years
RQS_summary %>% filter(Mean_Stint_Length >= 2) -> RQS_summary
RQS_summary %>% select(-c(Mean_Relative_Points)) -> RQS_summary

#create labels for notable drivers
RQS_summary %>% add_column(labels = NA) %>% relocate(labels, .before =
Driver) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Lewis Hamilton", labels = "Lewis
Hamilton")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Michael Schumacher", labels =
"Michael Schumacher")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Kimi R<e4>ikk<f6>nen", labels =
"Kimi Raikkonen")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Rubens Barrichello", labels =
"Rubens Barrichello")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Sebastian Vettel", labels =
"Sebastian Vettel")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Jarno Trulli", labels = "Jarno
Trulli")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Fernando Alonso", labels =
"Fernando Alonso")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Felipe Massa", labels = "Felipe
Massa")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Mika Hakkinen", labels = "Mika
Hakkinen")) -> RQS_summary
RQS_summary %>% rows_update(tibble(Driver = "Nelson Piquet Jr.", labels =
"Nelson Piquet Jr.")) -> RQS_summary

#Creating graphs
RQS_summary %>% ggplot(aes(x = Mean_Stint_Length,y =
Mean_Relative_Finishing_Position)) +
  geom_point() + geom_smooth(se=F, linetype = "dashed") +
  geom_smooth(method = "lm", colour = "yellow", size = 1.1) +
geom_text_repel(aes(label = labels)) +
```

```
  ggtitle("Mean Stint length vs. Mean Relative Race Result (Higher is
better)") +
  xlab("Mean Stint Length (Years)") + ylab("Mean Relative Race Result
(Position Diff.)")
RQS_summary %>% ggplot(aes(x = Mean_Stint_Length, y =
Mean_Relative_Qualifying_Gap)) +
  geom_point() + geom_smooth(se=F, linetype = "dashed") +
  geom_smooth(method = "lm", colour = "yellow", size = 1.1) +
geom_text_repel(aes(label = labels)) +
  ggtitle("Mean Stint length vs. Mean Relative Qualifying Gap (Higher is
better)") +
  xlab("Mean Stint Length (Years)") + ylab("Mean Relative Qualifying Gap
(Sec. Diff.)")
RQS_summary %>% ggplot(aes(x = Mean_Stint_Length, y =
Mean_Relative_Championship_Position)) +
  geom_point() + geom_smooth(se=F, linetype = "dashed") +
  geom_smooth(method = "lm", colour = "yellow", size = 1.1) +
geom_text_repel(aes(label = Driver)) +
  ggtitle("Mean Stint length vs. Mean Relative Championship Position (Higher
is better)") +
  xlab("Mean Stint Length (Years)") + ylab("Mean Relative Championship
Position (Position Diff.)")
RQS_summary %>% ggplot(aes(x = Number_of_Races, y = Mean_Stint_Length)) +
  geom_point() + geom_smooth(se=F, linetype = "dashed") +
  geom_smooth(method = "lm", colour = "yellow", size = 1.1) +
geom_text_repel(aes(label = Driver)) +
  ggtitle("Career Length vs. Mean Stint Length") +
  ylab("Mean Stint Length (Years)") + xlab("Career Length (Races Entered)")
RQS_summary %>% ggplot(aes(x=Mean_Stint_Length)) +
  geom_histogram(bins = 8, fill = "sky blue", colour = "dark grey", size =
1.2) +
  ggtitle("Mean Stint Length Histogram") + xlab("Mean Stint Length (Years)")

#getting correlation values for each graph
cor(RQS_summary$Mean_Stint_Length,
RQS_summary$Mean_Relative_Finishing_Position)
cor(RQS_summary$Mean_Stint_Length, RQS_summary$Mean_Relative_Qualifying_Gap)
cor(RQS_summary$Mean_Stint_Length,
RQS_summary$Mean_Relative_Championship_Position)
cor(RQS_summary$Number_of_Races, RQS_summary$Mean_Stint_Length)

#creating a LM with the data
summary(lm(Mean_Stint_Length ~ Mean_Relative_Finishing_Position +
Mean_Relative_Qualifying_Gap + Mean_Relative_Championship_Position, data =
RQS_summary))
```