Formula 1 Driver Analyser

A visual analysis tool for inspecting F1 drivers' performances

Luca Sannino 1542194
Sapienza University of Rome

Abstract

In this relation I describe a web application that uses the D3.js library to create a visual environment where a user can choose a Formula 1 driver (from 1994 to present day), so that they can learn more about their performance and place in the sport's history. This is achieved by interacting with visualizations that focus on the aspects of a driver's career that are of particular interest, like points achieved, results against their teammates, DNFs and consistency of their placements. Such visualizations, that are all synchronized between each other, include bar graphs, lollipop charts, box plots and scatter plots on multidimensional data achieved through PCA.

1. Introduction

At any given time, about only two dozens of people in the entire world can say to be Formula 1 drivers. Given the lack of seats, not only they need to drive as best as they can to climb up the standings, but they also need to prove that they are not wasting the place of far more skilled drivers.

As such, their performance has always been a topic of study and discussion, both internally in the teams themselves who are always looking for the next best hire, and externally across passionate fans, including people looking to make some quick money through bets. And since their performance is based on stats, it means it is a topic of interest for Visual Analytics as well.

This environment is then meant to help users (like those described above) gain a better knowledge of the drivers they are interested in, by presenting interactive visualizations that try to answer questions such as:

- Is the driver on an upward trajectory, or are his best days clearly behind him?
- How does his performance compare against the ones of his teammates across his career?
- Does he get DNF'd often? If so, is it because he is a hothead who constantly gets involved in crashes and collisions, or is it due to unlucky reliability issues?
- How constant he is in his placements? In which constructor did he gain the best results? In which circuits?
- How does he fit in the history of the sport itself, in terms of achievements reached?

This relation will go in detail in the related works, the dataset that made all of this possible, the visualizations themselves, how the user can interact with them, and will conclude with some final remarks, including my thoughts about the limitations.

2. Related Work

This environment's goal is partially inspired by "Visual Analysis of Formula 1 Races" by Lampprecht et al. [1]. Even though both works have the intention to enhance the user's understanding of a F1 driver, their visualization is "race-focused", meaning that a user can learn about the drivers' performances only in relation to a given race that is selected through a calendar chart, so that they have later the possibility to inspect their times down to single laps in a timeline graph and see which one was faster or slower during the various phases of a race.

This work, instead, is meant to offer a "driver-focused" prospective, meaning that users get to choose drivers in order to gain a better knowledge about their performances in relation to the entire stretch of their careers, and not for a race in particular.

The work proposed here is then meant as companion piece, since both tools together can now allow analyses that can be both focused on more narrower details (performances in races), or that can look at the bigger picture (performances in general).

3. The Dataset

The dataset at the basis of this project was found on Kaggle, courtesy of Rohan Rao [2]: it consists of a series of .csv files that cover a lot of interesting stats since the beginnings of Formula 1 until present day. In particular, the ones that were important for this environment were the files that dealt with the results of races and qualifying sessions, which also contained information about points obtained and the reason of a DNF, if there was any.

However, the dataset was fractured and had to be integrated before it could be effectively used: in order to do so, I used Talend Open Studio [3], which is a tool that allows the user to implement GAV mappings [Figure 1] in a intuitive way. Implementing GAV mappings means creating a new global schema such that every element inside it is a view over the source schemas (the structure of the original .csv files).

Source Schemas

results(..., raceId, driverId, constructorId, ..., position, ..., points, ..., statusId)
races(raceId, year, round, circuitId, name, ..)
circuits(circuitId, ..., name, ...)
driver(driverId, ..., forename, surname, ...)
constructors(constructordId, ..., name, ...)
status(statusId, status)

Global Schema

results(year, round, grandprix, circuit, surname, name, constructor, position, points, status)

GAV Mapping

```
{ (a, b, c, d, e, f, g, h, i, l) |
        results(..., x1, x2, x3, ..., h, .., l, ..., x4),
        races(x1, a, b, x5, c, ...),
        circuits(x5, ..., d),
        driver(x2, ..., e, f, ...),
        constructors(x3, .., g, ...),
        status(x4, l)
        } => results(a, b, c, d, e, f, g, h, i, l)
```

Figure 1: GAV mapping for the file results.csv

At the end of the process [Figure 2] I had obtained the following files: results.csv (that deals with race results), qualifying.csv (that deals with qualifying results) and standings.cvs (that deals

with the drivers' standings after each race). I'll now describe just results.csv, since the other two files have only small differences in their structure:

- **year**: Season in which the race was set;
- **round**: Round of the race in the season;
- **grandprix**: Official name of the race;
- **circuit**: Name of the circuit were the race was set in;
- **surname**: Surname of the driver;
- **name**: Name of the driver;
- **constructor**: Constructor the driver was driving for in that race;
- **position**: Position achieved at the finish line;
- **points**: Points achieved with the position;
- **status**: A comment that explain the reason of DNF, if there was any.

These files are then loaded in the application, and further manipulated by D3.js functions in order to extract the meaningful info that is needed by each chart.

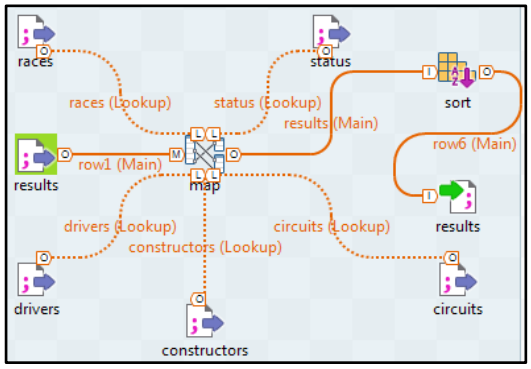

Figure 2: Implementing GAV mapping in Talend in order to create results.csv

4. Application Structure

This is a simple Javascript application that uses D3.js [3] to load data, to perform transformations on it, and to visualize it. In order to perform the PCA, however, the application sends a POST request to a Python script (that uses Flask), which performs the PCA with sklearn, and then sends the results back.

5. Visualizations

Once the environment has loaded in, the user is given the opportunity to choose a driver in a drop-down list, where the drivers are listed in alphabetical order. In order to restrict the range of choices, the user can also use a time slider to narrow the time period, so that only drivers that have driven in that particular range of time can appear in the list [Figure 3].

Once selected the driver, the user is presented with five visualizations, all displayed on the same page without need of scrolling: Points Per Season, Comparisons Against Teammates, DNF's Causes, Placement's Consistency On Finishes and Place In The Sport's History.

Each visualization has the task to answer one the questions that were asked in the Introduction section.

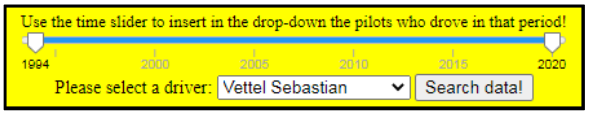

Figure 3: Driver selection screen

A. Points Per Season

This visualization is a bar graph [Figure 4], where every bar represents the points achieved by the driver across each year of his career. The bars are also colored, so that the user knows the constructors they are associated with thanks to a legend that is near the graph. If a pilot has driven for more constructors in the same year, then the bars are stacked on top of each other.

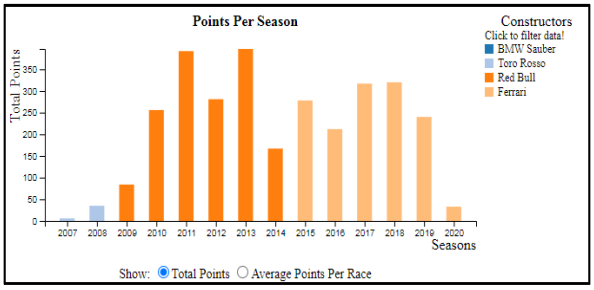

Figure 4: Points Per Season chart

The user can interact with the chart by switching between two units of measures (total points achieved or average per race), by clicking on the bars themselves in order to create new means on the fly and by clicking on the entries of the legend in order to filter in/out entire periods of time under a constructor from all the charts.

I think this visualization is useful because it gives a quick overview of where the driver was at the peak of his career, and where at the bottom. In this way, by interacting with the legend, the user can entirely focus only on those stretches of time under particular constructors that caught their eye, filtering out from this chart and all of the others information that may be considered useless or superfluous at the task at hand.

B. Comparisons Against Teammates

This is a diverging bar chart [Figure 5] where all the bars above represent the number of wins of the selected driver against his teammates across all the years of his career, while the bottom bars represent the losses. The bottom bars are colored as well, so that the user can check which teammates are being represented on the legend. Winning against a teammate means placing above him in an event, no matter the position, while losing means the opposite.

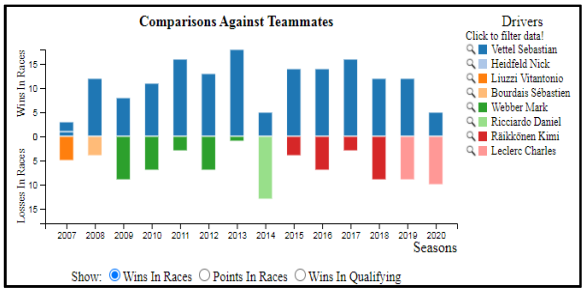

Figure 5: Comparisons Against Teammates chart

The user can interact with the chart by switching from wins/losses in races and qualifying, by visualizing the differences in points achieved in the races, by interacting with the legend to filter in/out specific teammates from this chart and the "Place in History" one (note that some teammates may already be disabled if a particular constructor was previously filtered out), and by selecting a new driver by clicking the magnifying glasses near the drivers on the legend.

Since I've always felt that you can't just compare two pilots who are driving cars belonging to different constructors to gain some insight of their skill (in Formula 1 there are far too many differences between all the cars), I thought that a comparison between teammates is the closest you can get to a comparison that feels 'fair', or at least reasonable, in order to extract how good a driver is.

Also, by switching between races and qualifying, it is possible to guess if a driver can drive faster than a teammate (more wins in qualifying) at the cost of a worse race pace (more losses in races), and vice versa, and by switching to the differences in points achieved it is possible to assign a more meaningful value to the differences in wins/losses, since it's one thing to have more wins when we are dealing with races that grant no points, another is to win that important race that instead scored a lot of points.

C. DNFs' Causes

This lollipop chart [Figure 6] shows the number of DNFs of the selected driver, divided in three macro categories: Collision Dmg, Reliability and Retirement. Collision Dmg includes cases of external damage to the car (tyre punctures, broken wings etc.) that are the consequence of collisions with the track or other drivers, while Reliability includes all cases of internal issues (power loss, overheated engines, brakes failures etc.) that are beyond the driver's responsibility. Lastly, Retirement includes the cases where the driver voluntarily drove to the pit line in order to quit the race.

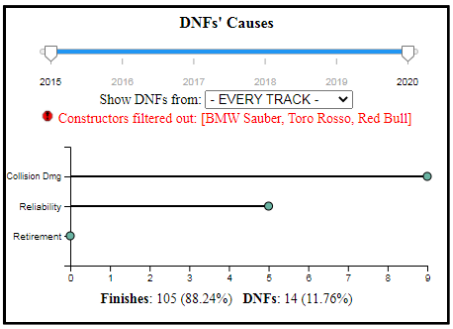

Figure 6: DNFs' Causes chart

The reason I decided to focus on these three macro categories was because the status.csv file contained hundreds of different reasons for a DNF, far too many to be reasonably visualized. So I decided to incorporate them in macro categories, so the user could instantly see if a driver

gets DNFed because he always manages to get himself in the worst position at the worst time (high number of Damage DNFs), or if he's just unlucky and it's all the car's fault (high number of Reliability DNFs).

Users can also use a time slider, and the previously mentioned ability to filter races under certain constructors, to focus on the exact range of races they are interested in. If they wish, they can also focus on particular track. Also, since the scale of the x-axis never changes, they can see how the number of DNFs in a particular range of time fares in relation to the whole.

D. Placements' Consistency

If the previous chart was focused on the retirements, this box plot [Figure 7] is instead focused on showing the distribution of the placements when the driver actually reaches the finish line.

Once again, user can focus on a set of races by using a time slider and by picking a select circuit, and once again the set of races accounts for the filtered constructors.

The reason I decided to use a box plot was because I felt there wasn't anything better that could instantly showcase how the performance of a driver increases or degrades when certain time periods and constructors are selected, because all the user needs to do is focus on where the quartiles of the distribution are placed.

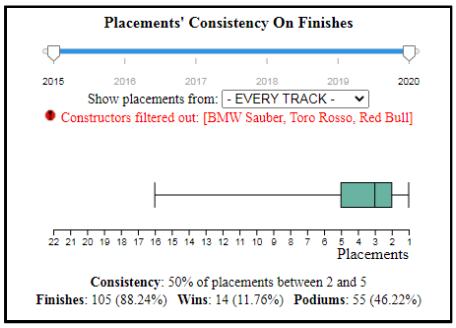

Figure 7: Placements' Consistency chart

E. Place In Sport's History

This is a scatter plot (Figure 8) that shows where a driver, identified by a circle, stands against all the other drivers, also identified by circles, in terms of achievements. The closer two drivers are, the more similar their achievements are.

In order to represent this, a 7-dimensional tuple was first created for each driver that contained some of their most important achievements (number of points, starts, finishes, poles, podiums, wins and championships). Then after executing the PCA on this set of tuples, the results are shown in the 2D scatter plot, where the two axis are the two principal components that better express the variance of the dataset. Of course there's always the possibility of the introduction of false positives.

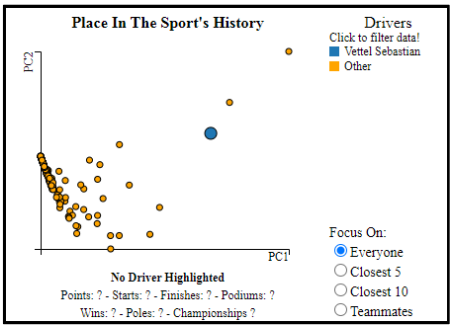

Figure 8: Place In Sport's History chart

When the scatter plot is displayed, the dot belonging to the selected driver is bigger and differently colored from the rest, so that the user can instantly recognize him, and see where he fits in terms of achievements.

Since the view is cluttered by a lot of points, if a user wants to see which are the drivers that most resemble the selected one, then they can choose an option that displays only the closest five (or ten) drivers. Alternatively, they can also choose an option that only focuses on the selected driver and his teammates. Since hovering over the names of the teammates in the Comparisons Against Teammates chart highlights the corresponding circles in the PCA (and vice versa), a user can also see how strong a teammate is, so that they can attribute the right "value" to some of the wins/losses (a rookie losing against a far more skilled and awarded teammate is way less severe than the opposite situation).

Also, once again, the user can select a new driver by clicking the magnifying glasses on the legend.

6. Limitations

The biggest limitation is surely the lack of qualifying info, that forced the list of selectable drivers to start from 1994. Even though the dataset promised complete stats from the fifties to today, not only qualifying data starts only from 1994, but there are also some short gaps even after, which were noticed only during the more advanced phases of this project. This forced me to use an asterisk near every value that expresses the number of pole positions, which notifies the user that such stats may contain missing information and that should be taken with a grain of salt.

Regarding the scope of this work, being a solo project, some cuts had to be made: for once, I wish I could have implemented more filters for the 'Comparisons Against Teammates' chart, for example visualizing the wins/losses in races where only both drivers achieved a certain threshold of points, or filtering out races where one of them suffered from a reliability issue that prevented him from reaching the better place against the teammate. Such filters can only improve the user's understandings of a dynamic between the selected driver and his teammates.

Also, even though I stand by my decision to show only three macro categories, I wish I could have implemented a toggle that lets the DNF chart display more categories in order to satisfy the more demanding users that want to know the exact reason behind a DNF.

I also wish I could have implemented the initial selection of a driver through visualization, and not through the unusual time slider/dropdown combo: however I could not think of anything that did not feel "gimmicky", meaning that it used visualization just for the sake of it, so I settled for something that I felt to be more intuitive.

7. Conclusion

All in all, while it's surely no match for a more professional tool that is built on a larger quantity of data not publicly available, I feel that "Driver Analyser" offers a quick and intuitive way to gain a greater understanding of a driver's career, where a lot of meaningful insights can be made just with a short series of 'clicks'.

8. References

[1] Lampprecht, Tobias & Salb, David & Mauser, Marek & Wetering, Huub & Burch, Michael & Kloos, Uwe. (2019). Visual Analysis of Formula One Races. 10.1109/IV.2019.00025.

[2] https://www.kaggle.com/rohanrao/formula-1-world-championship-1950-2020

[3] www.talend.com

[4] www.d3js.org