Data Prep and Analysis:

Data Prep:

The data extracted is extracted from the The Ergast API and include information on all races and drivers who participated in the 2009 through 2022 seasons. (Table A in the appendix offers a detailed description of the dataset variables after data cleaning.)

Preprocessing has been done on the acquired data including removing values that are not pertinent to this scenario and determining what to do with null values:

- driverDateOfBirth and raceDate are converted to datetime. This is to facilitate operations that involve calculating drivers ages, durations between dates, and sorting the data by dates.
- driverNumber, driverFastestLapSpeed, driverFinalGridPos, and driverFatestLapNumber are converted to numeric types, from strings. This ensures that numerical computations, like mathematical operations or statistical analysis, can be performed on these columns.
- Combining Columns: The full name of the driver is created by concatenating driverForename and driverSurname, and storing it in a new column driverName. This is helpful for any analysis where a full name format is more useful than separate forename and surname.

Feature Engineering

- Calculating Age: The age of the drivers is calculated by subtracting their date of birth (driverDateOfBirth) from today's date. Calculating the age can be critical for analyses that might look into factors like experience or age-related performance.
- Handling Missing Data for driverFastestLapSpeed: The missing values in the driverFastestLapSpeed column are replaced with the average of its median and mean. This approach helps to mitigate the impact of outliers on the replacement value.
- Filling Numeric Columns with Zero: Several columns (driverFinalRank, driverFatestLapNumber, constructorRacePoints, driverNumber, driverFinalGridPos, constructorChampionshipStandingWins, constructorChampionshipStandingPoints, constructorChampionshipStandingPosition, driverChampionshipStandingPoints, driverChampionshipStandingPosition) are filled with zeros. In racing data, missing points and data indicate that a driver did not finish a race, did not earn points, or did not achieve a certain position. Using zeros preserves the meaning of having no contribution or no occurrence, which might be more informative than averaging, particularly for rankings and scores.
- The names of racing constructors are being updated in the dataset to reflect changes over the years.
- The driverDnf column is created to indicate whether a driver did not finish the race due to specific reasons. The lambda function checks if the driverRaceResultStatus is one of several status codes (like 3, 4, 20, etc.), which represent various DNF reasons (like crashes, mechanical failures, etc.). If the status code matches any of these, the column is set to 1; otherwise, it's set to 0.

- Similarly, the constructorDnf column is created, but it checks for a different set of status codes. If the status code is not in the provided list (and is also not 1, which typically means 'finished the race'), then it indicates a DNF for reasons not common to those tracked in driverDnf. If it matches these conditions, it is set to 1; otherwise, 0

Data Analysis

Upon reviewing Chart a[1], we observe the following trends and insights regarding team performance in terms of points per race:

- Performance Dominance: Mercedes stands out as the dominant team in terms of points per race, suggesting they have had a successful strategy, excellent drivers, and reliable cars.
- Competitive Cluster: The closeness of Ferrari and Red Bull suggests a competitive rivalry, where both teams are fairly matched in their ability to earn points in races.
- Midfield Competition: The grouping of teams like McLaren, Renault, and Alpha Tauri/Williams implies tight competition in the midfield, with these teams occasionally breaking into higher positions.
- Emerging or Struggling Teams: Haas, Toyota, BMW Sauber, and Honda represent teams that either have not yet established a foothold in the sport or have historically struggled for points.

Examining Chart b and Chart c offers insights into both drivers' and constructors' championship points across multiple seasons and the factors shaping their performance:

- The driver's graph can be influenced by the individual skill of the driver, their career decisions, and the performance of their team. High points in a season usually correlate with success in races, but a sharp fall could suggest a bad season, accidents, or moving to a less competitive team.
- The constructor's graph highlights overall team performance, impacted by budget, leadership, and technical innovations. Consistently high or improving average points suggest a strong team possibly benefiting from successful car development and good driver line-ups.

Exploring the Correlation Matrix illustrated in Chart d, we uncover significant associations among key variables:

- Driver start grid position has a strong positive correlation with driver final grid position (0.59). This means that drivers who start in higher positions are more likely to finish in higher positions.
- Driver final grid position has a strong positive correlation with driver race points (0.44). This means that drivers who finish in higher positions are more likely to score more points.

[1] **For all charts please refer back to the appendix**

- Driver age has a weak negative correlation with driver championship standing points (-0.61). This means that younger drivers tend to score more points in the championship.

Modeling:

Upon completion of the data preparation phase, the subsequent step involves the application of machine learning algorithms. To tackle this matter, three distinct algorithms, Logistic Regression, Random Forest, Decision Tree and a dummy variable were employed. Each algorithm underwent experimentation with different configurations and hyperparameters. This section presents a detailed account of the specific configurations utilized for these algorithms.

From our EDA analysis, we concluded that the most important features with a high correlation between the driver final position were ‘season','driverId','constructorId', 'driverStartGridPos', 'driverFinalGridPos','driverChampionshipStandingPosition', 'constructorChampionshipStandingPosition'. We used these as our prediction features with the target variable being driverFinalRank.

In our model development process, 80% of our total dataset was allocated for training, while the remaining 20% was set aside for testing. Within the training subset, we employed a StratifiedKFold cross-validation technique with 10 splits, where each fold represented 10% of the training data. For each cross-validation iteration, 9 folds (90% of the training data) were used for training the model, and 1 fold (10% of the training data) was used for validation. This setup was consistently applied across all models (Logistic Regression, Decision Tree, Random Forest), ensuring uniform training and validation conditions to fairly assess and compare the performance of each model on data it has seen and data it has not.

Models Selected:

Logistic Regression

The Logistic Regression model performance is influenced by hyperparameters such as the regularization strength (C) and the type of penalty applied (l1 or l2). The model's coefficients, which were optimized during GridSearchCV, provide insights into the importance of each feature. For instance, a high positive coefficient for driverStartGridPos suggests starting positions significantly influence final rankings.

Decision Tree

The Decision Tree model uses a flowchart-like structure to make predictions, where the most influential features are closer to the root. The hyperparameters such as max_depth, min_samples_split, and min_samples_leaf were tuned using GridSearchCV. The tree's splits are based on feature importances, indicating the impact of features like driverFinalGridPos on the prediction of driver final ranks. The model's simplicity provides transparent decision rules, but it can be prone to overfitting.

Random Forest
The Random Forest model is a collection of decision trees where each tree contributes a vote to the final prediction. Key hyperparameters like the number of trees (n_estimators), maximum tree depth (max_depth), and minimum samples for a split (min_samples_split) were optimized through GridSearchCV. The feature importances extracted from the model reveals that variables like driverChampionshipStandingPosition carry considerable predictive power. Unlike single decision trees, the Random Forest model mitigates overfitting through ensemble learning, which leads to a more robust and generalized performance.

Dummy Model
In the context of predicting F1 rankings, a dummy classifier does not provide meaningful insights or performance metrics. This is because the F1 ranking is determined by complex factors such as driver skill, car performance, race strategies, and external conditions, which a dummy classifier cannot capture.
For example, we used a dummy classifier that predicts our target variable based on the most frequent class (e.g., always predicts the most common ranking). The accuracy obtained as shown in Table 1 reflects the model's inability to correctly predict F1 rankings based on relevant features.
After conducting a grid search, the table showcasing the best hyperparameter configurations for the three models shows:

Model	Best Parameters	Best Accuracy Score
Logistic Regression	C: 10, penalty: 'l1', solver: 'liblinear'	0.2043
Decision Tree	max_depth: None, min_samples_leaf: 1, min_samples_split: 2	0.8481
Random Forest	max_depth: 30, min_samples_split: 10, n_estimators: 200	0.8125

Table 1: Best Model Parameters following hyperparameter tuning

The Logistic Regression model shows significantly lower performance with an accuracy of just over 20%. This indicates that the model struggles with the complexity or non-linearity of the data. The l1 penalty might help in feature selection by driving some coefficients to zero, which is useful in high-dimensional datasets or datasets with irrelevant features. However, the model's low accuracy suggests it might be overly simplistic or not adequately capturing the relationships in the data needed to accurately predict F1 rankings.

The Decision Tree model performs much better, achieving an accuracy score of approximately 85%. The lack of a maximum depth (max_depth: None) indicates that the tree was allowed to grow until all leaves are pure or contain less than min_samples_split samples. This configuration might risk overfitting, but it seems to work well in this scenario, possibly due to the complexity and specific nuances of the data, like different tracks, weather conditions, and team strategies.

The Random Forest model, an ensemble of Decision Trees, also shows strong performance, with an accuracy of over 81%. The use of multiple trees helps reduce the risk of overfitting compared to a single decision tree, and the substantial number of trees (n_estimators: 200) likely contributes to a robust model that averages numerous trees' predictions. The max_depth of 30 allows the individual trees to grow deep enough to capture complex patterns but is limited to prevent fitting excessively to noise.

Models Performance

Metrics in Context:
In Formula One ranking prediction, F1 score and a balance between precision and recall are more valuable than just accuracy for a few key reasons:

- **Uneven Distribution of Rankings:** Not all finishing positions in F1 hold equal weight. Podium finishes (top 3) are far more important than midfield positions. Accuracy can be misleading if a model simply predicts the most common finishing positions (e.g., midfield) and achieves a high score without identifying true top performers.
- **Focus on Both Top and Underdog Performers:** F1 races can have surprise results and underdogs occasionally outperform expectations. A good model should not only identify consistent top drivers but also catch these breakout performances.

Here's how F1 score and the balance of precision and recall address these issues:

- **F1 Score:** This metric considers both precision (correctly predicting high finishes) and recall (correctly predicting low finishes). A high F1 score suggests the model can identify both the best and the underdogs effectively.
- **Precision and Recall Balance:** A model with high recall but low precision means it might predict many top finishes, but most of them turn out to be wrong. Conversely, high precision with low recall suggests the model rarely predicts top finishes, even when they occur. A balanced approach, reflected in a good F1 score, is ideal.

BEFORE HYPERPARAMETER TUNING METRICS

Model	Accuracy	F1 Score	Recall	Precision	Balanced Accuracy

Logistic Regression	0.20515	0.16770	0.18740	0.16697	0.18740
Decision Tree	0.85061	0.77160	0.77319	0.77209	0.77319
Random Forest	0.81773	0.74121	0.74185	0.74395	0.74185

Table 2

Logistic Regression: This model has the lowest accuracy (0.205) and F1 score (0.168) across all metrics. This indicates poor performance in predicting driver rankings. It suggests the model struggles to capture the complexities of F1 races, which depend on various factors beyond basic driver skill.

Decision Tree: This model shows significantly better performance than Logistic Regression. It boasts a high accuracy (0.851) and F1 score (0.772). This suggests the model can learn complex relationships between driver performance and finishing positions. However, the high accuracy might be misleading.

Random Forest: This model performs well with an accuracy (0.818) and F1 score (0.742) that falls between the other two models. Random Forests often provide a good balance between accuracy and robustness compared to single Decision Trees.

- Accuracy: While Decision Tree has the highest accuracy, it might not be the most reliable measure. Formula One races can have many retirements and unexpected events. A model might simply predict the most common finishing positions (e.g., midfield drivers) and achieve high accuracy without truly understanding driver skill.
- F1 Score: This metric considers both precision (correctly predicting high finishes) and recall (correctly predicting low finishes). A good F1 score suggests the model can identify both top performers and underdogs effectively.
- Precision & Recall: These metrics provide further details. Here, the models have slightly lower precision than recall, indicating they might struggle to identify true top performers more often than missing strong finishes.
- Balanced Accuracy: This metric gives equal weight to all positions, which might be less relevant in F1 where podium finishes are more important than midfield battles.

Overall the Decision Tree appears to be the best model based on the data, but accuracy alone might be misleading. For F1 ranking prediction, F1 score and a balance of precision and recall are more valuable.

AFTER HYPERPARAMETER TUNING METRICS

Model	Accuracy	F1 Score	Recall	Precision	Balanced Accuracy
Logistic Regression	0.211580	0.189905	0.211580	0.195955	0.193819
Decision Tree	0.822731	0.822774	0.822731	0.830046	0.750323
Random Forest	0.854182	0.851192	0.854182	0.849782	0.776649

Table 3

Hyperparameter tuning seems to have yielded the following improvements:

- Logistic Regression: Still the weakest performer, but a slight increase in all metrics (around 0.005) suggests a marginal improvement.
- Decision Tree: Accuracy and F1 score remain high (around 0.82), indicating a robust model for capturing complex relationships. However, the lower balanced accuracy suggests it might favor predicting the middle of the pack more often.
- Random Forest: This model shows the most significant improvement, with accuracy and F1 score exceeding 0.85. It maintains good precision and recall balance and surpasses Decision Tree in balanced accuracy. This suggests it offers the best prediction power across the field.

Overall Hyperparameter tuning has improved all models, but Random Forest emerges as the clear winner for F1 ranking prediction. It provides high accuracy, good balance between precision and recall, and performs well across the finishing positions.

Implemented Model and Performance

The performance of the Decision Tree and Random Forest models suggests that the ranking of F1 drivers is influenced by complex interactions of features that these tree-based models can capture more effectively than a linear model like Logistic Regression. Given the strategic and varied nature of Formula 1 racing, where factors such as driver skill, team strategies, car performance, and race conditions play intertwined roles, models that can handle non-linearity and feature interactions seem more suitable.

The logistic regression's poor performance might be improved by feature engineering, considering interaction terms, or using polynomial features. However, given the high accuracy of the tree-based models, focusing on refining the Random Forest model or further tuning the Decision Tree (possibly by adjusting parameters to control overfitting) might be more fruitful approaches for predicting F1 rankings.

Appendix

Table A: Data Set Variables

Variable	Type	Source	Description
raceId	Numerical	Extracted	Unique identifier for each race.
season	Numerical	Extracted	Year of the racing season.
raceNumber	Numerical	Extracted	The sequence number of the race in the season.
prixName	Categorical	Extracted	Name of the Grand Prix.
raceDate	Date	Extracted	The date on which the race took place.
driverId	Numerical	Extracted	Unique identifier for each driver.
constructorId	Numerical	Extracted	Unique identifier for each constructor.
driverStartGridPos	Numerical	Extracted	The starting grid position of the driver in the race.
driverFinalGridPos	Numerical	Extracted	The final grid position of the driver in the race.
driverFinalRank	Numerical	Extracted	The final rank of the driver in the race.
driverRacePoints	Numerical	Extracted	The number of points the driver earned in the race.
driverLapCount	Numerical	Extracted	The number of laps completed by the driver.
driverFatestLapNumber	Numerical	Extracted	The lap number on which the driver had the fastest lap.
driverFastestLapTime	Time	Extracted	The time duration of the driver's fastest lap.
driverFastestLapSpeed	Numerical	Extracted	The average speed during the driver's fastest lap.
constructorRacePoints	Numerical	Extracted	The number of points earned by the constructor in the race.
driverNumber	Numerical	Extracted	The official racing number of the driver.
driverDateOfBirth	Date	Extracted	The date of birth of the driver.
driverNationality	Categorical	Extracted	Nationality of the driver.
constructorName	Categorical	Extracted	The name of the racing team. Updated for historical consistency.
constructorNationality	Categorical	Extracted	Nationality of the constructor.
constructorChampionshipStandingPoints	Numerical	Extracted	The number of points the constructor has in the championship standings.

constructorChampionshipStandingPosition	Numerical	Extracted	The position of the constructor in the championship standings.
constructorChampionshipStandingWins	Numerical	Extracted	The number of wins the constructor has in the championship standings.
driverChampionshipStandingPoints	Numerical	Extracted	The number of points the driver has in the championship standings.
driverChampionshipStandingPosition	Numerical	Extracted	The position of the driver in the championship standings.
driverChampionshipStandingWins	Numerical	Extracted	The number of wins the driver has in the championship standings.
circuitName	Categorical	Extracted	The name of the circuit where the race took place.
circuitLocation	Categorical	Extracted	The location of the circuit.
circuitCountry	Categorical	Extracted	The country where the circuit is located.
driverRaceResultStatus	Categorical	Extracted	The race result status for the driver.
driverName	Categorical	Extracted	The name of the driver.
driverAge	Numerical	Calculated	Calculated by subtracting the driver's date of birth from the current date and converting to years.
driverDnf	Binary	Calculated	Indicates if a driver did not finish the race, with 1 for various DNF reasons and 0 for finishing.
constructorDnf	Binary	Calculated	Indicates if a constructor's driver did not finish the race, with 1 for various DNF reasons and 0 for finishing.

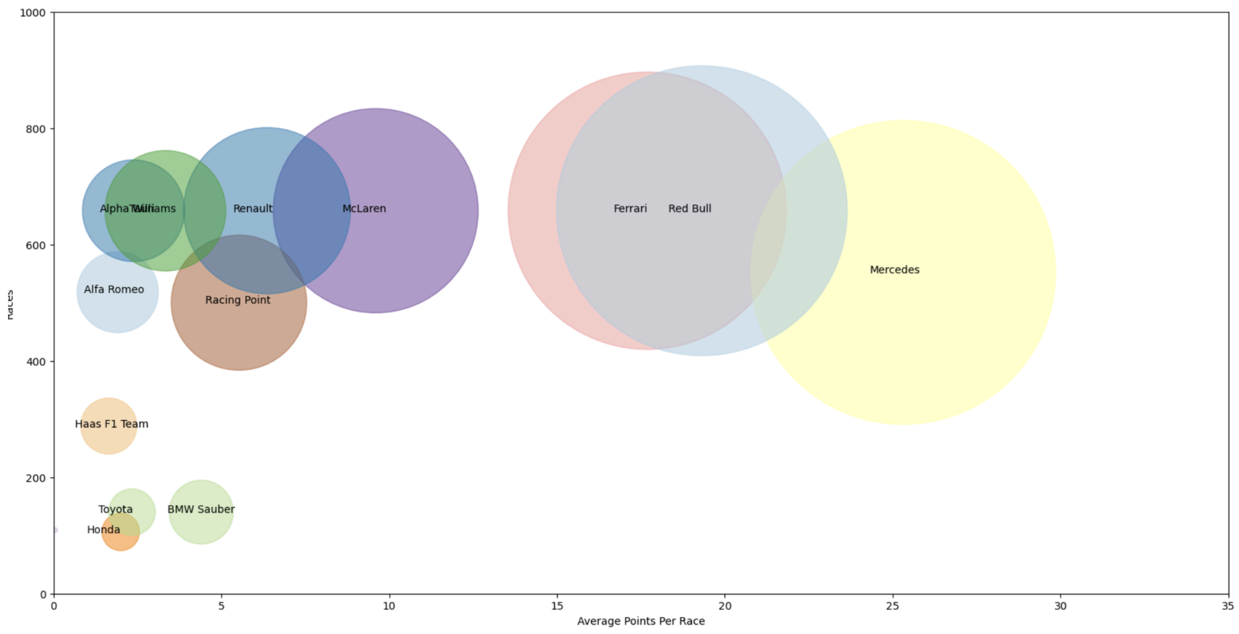

Chart a: Team performance in terms of points per race

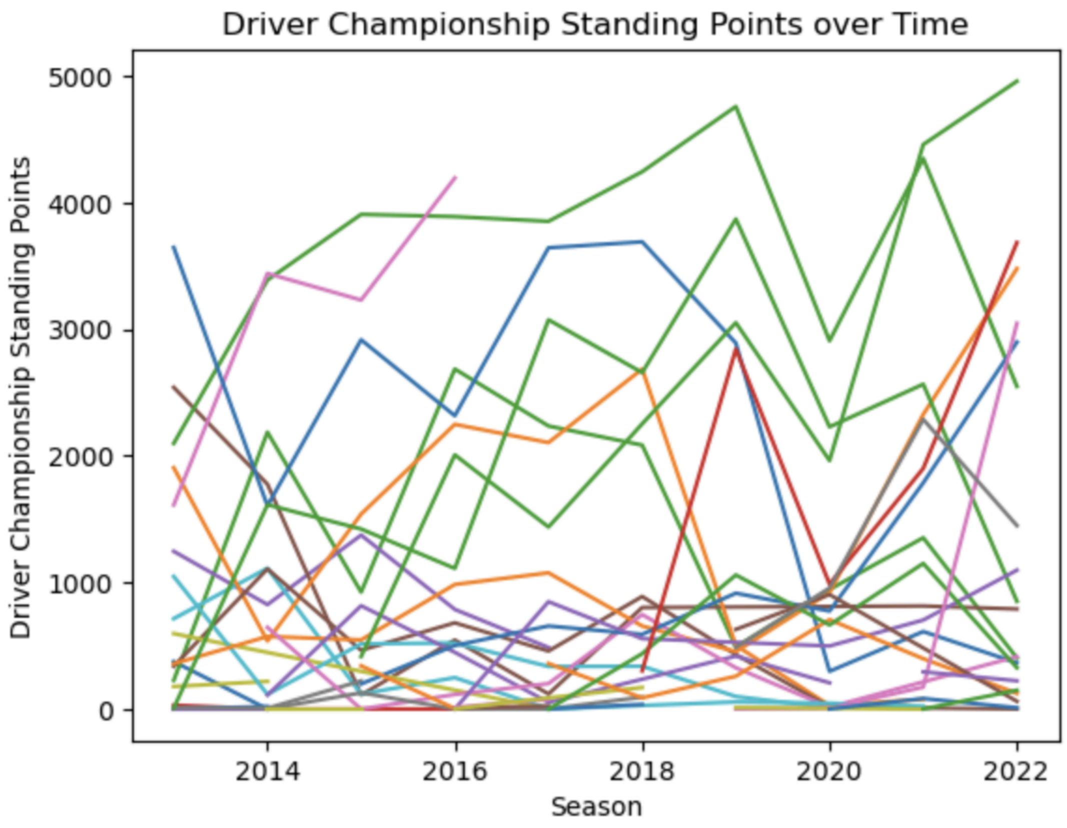

Chart b: Driver championship standings points over time

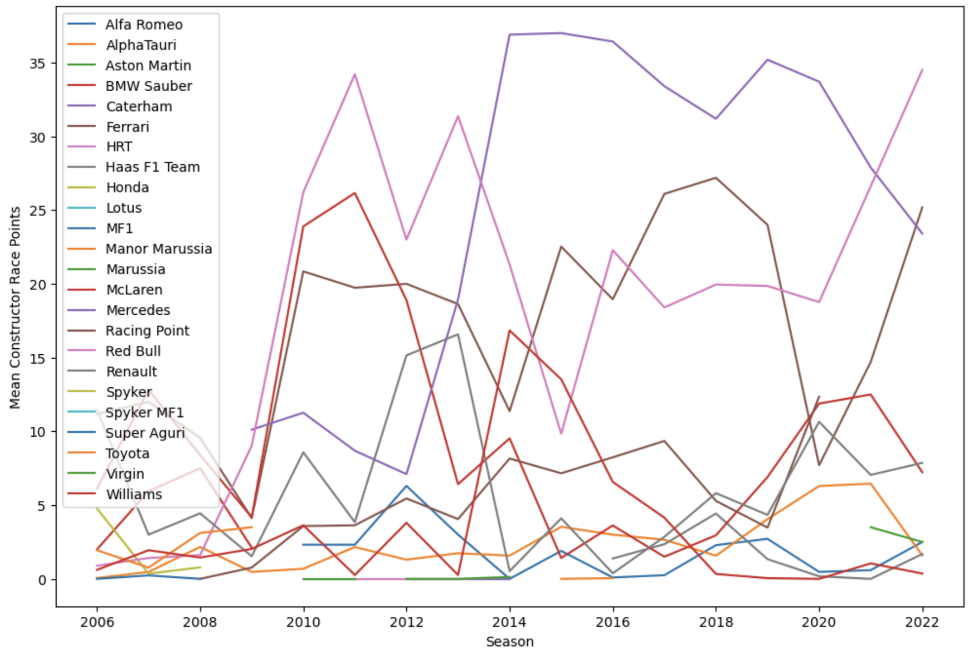

Chart c: Performance of constructors over the years

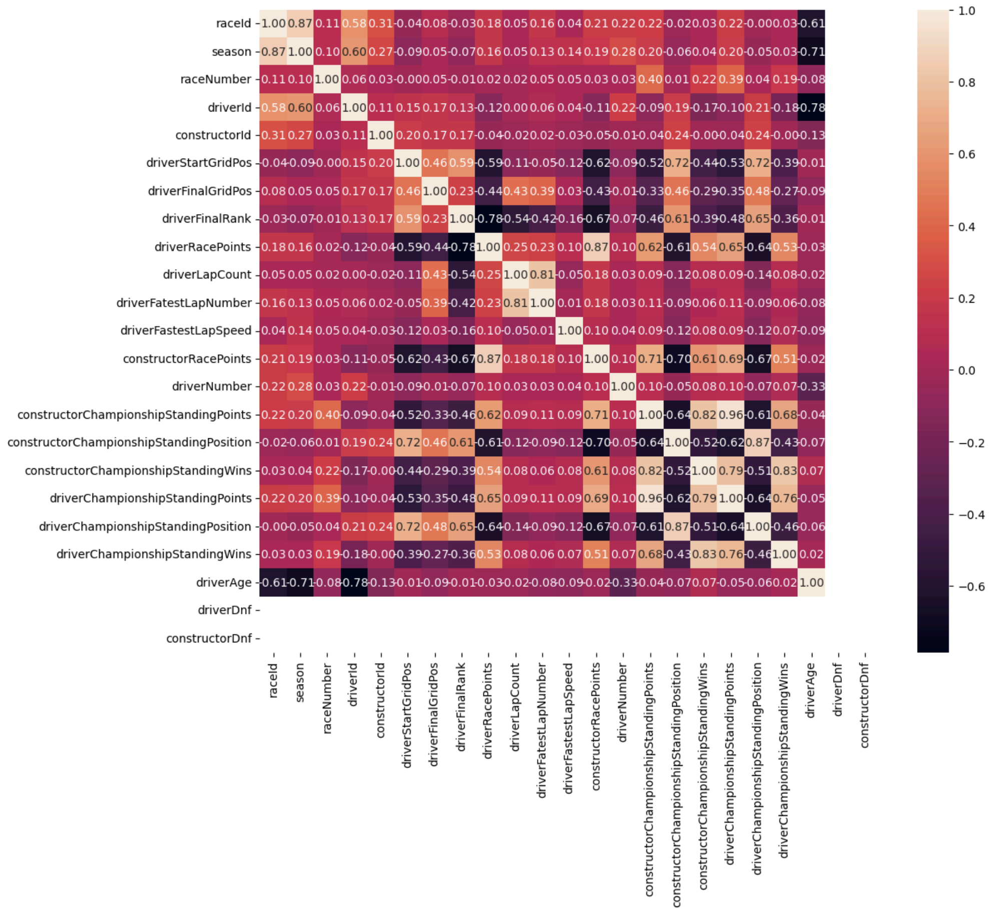

Chart d: Correlation Matrix