COMP 479 Final Project Write-up   Elias Crum + Adriana Ene

Introduction

Formula 1 (F1) is a racing circuit that shows off some of the fastest, most expensive, and most technically advanced motor vehicles in the world. The circuit began in the 1950's and has blossomed into one of the largest sporting leagues in the world. While data generation and analysis has become an invaluable asset to F1 engineers and car developers, access to such data is heavily guarded. In the project described here, we aim to use the results of F1 races from 1950-2017 to build and train multiple Logistic Regression and SVM models to predict the probability that a particular driver will win a given F1 race.

Dataset Description

We used data relating to Formula 1 (F1) racing outcomes over the last 70 years. A brief description of the information offered in the dataset we analyzed: in F1 there are teams that develop cars that race on different tracks across the world. Each team has two drivers and, essentially, gives each driver identical cars. These different teams are called constructors and at the end of each season there is an award for the team that accumulates the most points (termed the constructors championship). Additionally, each individual driver is competing to win the individual drivers' championship by accumulating the most points of any one driver by finishing near the front in each race. The dataset we used to train our models had a total of 18 features and we combined from three individual datasets. The features included information we thought could be predictive of the placing in which a driver could finish. The features include information about the starting position of the racer, the racer's constructor, the placing of the constructor and driver at the time of the race, the fastest lap the driver had, the number of wins the driver has, number of points the driver has at the time of the race, etc.

Overall, the dataset includes data about racing outcomes for 20 drivers in each of 988 races along with the constructors' championship and the drivers' championship rankings of that driver at the time of the race. In total, the dataset has 18 features and 22,981 rows.

Data Cleanup

A large part of this project was spent combining and cleaning the dataset. We looked at the data file called constructor_standings.csv, driver_standings.csv and results.csv. We combined the consttuctor_standings, results, and driver_standings dataframes over conserved 'constructorId','raceId', and 'driverId' features. Thus, the different dataframes could be combined using the `pd.merge()` method within pandas. The final dataframe was then modified to remove some features we thought were too highly indicative of the winning a race, such as points won and total time. We decided to remove the highly predictive columns columns as well as some columns that we deemed to be totally useless to classification such as 'resultId', 'constructorStandingsId', 'positionText_x', 'positionText_y','positionOrder', 'points_x','time','milliseconds','fastestLapSpeed', and 'fastestLapTime'. The 'position_x' feature is the final position of the driver so this feature was removed and assigned to be our vector of labels. An issue we ran into was missing data, particularily in places were drivers did not finish a race. In these cases, we replaced the empty data with the value ‘21’ to indicate that this driver placed in last (21 because there are 20 drivers that compete in a race). We also kept the fastestLap as a feature and replaced any empty rows with the value 0 to indicate there was no fastest lap. There were 6 datapoints that had the car number missing, a fairly unpredictive feature but something we chose to keep in to supplement the amount of predictive power. Due to added complexity, we chose to drop the 6 rows that did not have a number value out of the dataframe. To compare the performance of the trained models, we created a sister dataframe that has the finishing position lumped into 3 categories. The first category, ‘0’, are the podium positions (1-3), category ‘1’ are positions that win points, positions 4 –10, and category ‘2’ are the rest of the participants in positions 11+. We split both dataframes into 80% training and 20% testing. The Training set was further divided into 80% training and 20% development for optimizing out model hyperparameters.

Baseline Approach

We wanted to test the difference between SVM and Logistic Regression models on both the uncategorized and categorized race data. We made two separate methods that perform a grid search for the most predictive hyper-parameters of both the SVM and Logistic Regression models separately. Importantly, because we were not doing binary classification, we used the ‘multi-class’ hyperparameter in both the categorized and uncategorized cases. Unfortunately, this disallowed for use some regularization techniques in the logistic regression model. In the SVM grid search, the C value and kernel selection were altered to find an optimal model. For the Logistic Regression model, the C value and penalty (regularization term) was altered. The algorithm we created determined the most accurate model hyperparameters using both the F1 score and accuracy value because we were not sure of how sparse our dataset was. These optimal hyperparameters were attained using a development dataset.

Method Description

For dataframe merging, cleaning, and troubleshooting, we utilized multiple Pandas methods. For a more detailed description please see our code and the associated comments. We also standardized all features using the sklearn `StandarScalar()` method.

The first method we created `svm_select()` runs a gridsearch for SVM hyperparameters. The gridsearch searches over C values and kernel types of models trained on the training set and assessed on the development set of both the categorized and uncategorized places separately. The methods print the accuracy and F1-scores for each combination of hyperparameters on the development set. The method returns the hyperparameters found to be most accurate / with the highest F1 score.

The other method we created is `lg_select()` which uns a gridsearch for Logistic Regression hyperparameters. The gridsearch searches over C values and penalty types. The Logistic regression method runs in the same way as the SVM method and returns the hyperparameters found to be most accurate / with the highest F1 score when assessed on the development set.

The `test()` method evaluates the performance of the optimized models trained by the previous methods on the testing set. The accuracy and F1 scores produced were then compared to assess the efficacy of the different models/data organization on the prediction of F1 race outcomes via historical data.

Evaluation

When optimizing the hyperparameters of the uncategorized SVM model, we found that a C-value of 100 and a linear kernel gave the highest accuracy/F1-score whereas for the categorized SVM model a C-value of 1 and a linear kernel gave the highest accuracy/F1-score. Comparatively, the optimal hyperparameters of the uncategorized Logistic Regression model were a C-value of .001 and no penalty and the categorized had a C-value of 10 and an L2 penalty. These optimal hyperparameters were then used to train their respective models on the training data. The trained models were then presented the testing data and the following results were obtained. Uncategorized SVM model accuracy: 0.475; categorized SVM model accuracy: 0.741; uncategorized Logistic Regression model accuracy: 0.742; categorized Logistic Regression model accuracy: 0.447. For a baseline, we used the `dummy_classifier()` method with strategy = “uniform”. For the unclassified data, the accuracy was 0.04, and for the classified the accuracy was 0.32.

Discussion

Going into this project, we assumed it would be difficult for an ML model to predict the winners of the F1 races using the data we provided. We also hypothesized that the models would perform better on the categorized data than on the uncategorized data because there are fewer categories to choose from.

Based on these hypotheses, it was unsurprising that the uncategorized data proved more challenging for the models to predict than the categorized data. What is surprising is that both the logistic regression and SVM models were both very challenged to accurately classify the Testing set labels, each with only 75%. This suggests that when categorized, the data is much more separable within the classified dataset. When assessing the uncategorized data, the SVM and logistic regression models both performed with ~50% accuracy whereas the control dummy classifier had only 4% accuracy. On the categorized data, the two models both performed much better than the dummy classifier, like on the uncategorized data, suggesting that our models are working but that the dataset is a challenge for the models chosen. When determining the hyperparameters of the logistic regression model using the `gridsearch()` method, there was a recurring error stating that the logistic regression model was unable to converge over the max number of iterations. This suggests that the uncategorized data just simply does not work well with the classifiers chosen, likely because of the 21 different potential classifications.

Conclusion

We found that when categorized, both logistic regression and SVM models could somewhat accurately predict the position category of each driver’s finish in a given F1 race. Conversely, when the data was not categorized, we found that both classifiers performed substantially worse and only exhibited about 50% prediction accuracy. These findings supported our hypothesis that the classified data would offer a more accurate model due to less predictive complexity. Based on these findings, we conclude that either model could be used and provide almost equal accuracy in the different classified and unclassified scenarios and that it is possible to predict the winner of a given F1 race with a small degree of confidence.