F1 Race Winner & Strategy Prediction using Machine Learning

Khushal Yadav
IIIT Delhi
New Delhi, Delhi
khushal22249@iiitd.ac.in

Naman Birla
IIIT Delhi
New Delhi, Delhi
naman22310@iiitd.ac.in

Omansh Arora
IIIT Delhi
New Delhi, Delhi
omansh22342@iiitd.ac.in

Abstract

Formula 1 (F1) racing is a highly competitive sport that requires teams to make strategic decisions throughout the race to optimize their performance and increase their chances of winning the race. This paper aims to develop a machine learning (ML) model capable of predicting F1 race winners and providing optimal strategic decisions and recommendations, such as number of pit stops and tire selections, to improve their odds of winning. The MLP model is trained using multi-class cross-entropy loss and the Adam optimizer, enabling it to effectively differentiate strategies and improve prediction accuracy. In addition, we plan to fine-tune hyperparameters such as learning rate, batch size, and the architecture of the model, including the number of layers and neurons, to identify the optimal configuration. Finally, our exploration of evaluation metrics like accuracy, F1 score (macro and micro), precision, and recall deepened our understanding of their relevance in binary classification problems. [GitHub Link]

1. Introduction

Formula 1 (F1) racing is a highly competitive sport that requires teams to make strategic decisions throughout the race to optimize their performance and increase their chances of winning. Factors such as number of pit stops, tire selection, fuel management, race strategy and driver performance all play a crucial role in determining the winner. Recently in 2018, F1 partnered with Amazon Web Services (AWS) to revolutionize the sport of Formula 1 racing, by enabling the harnessing of Machine Learning techniques and High Performance Cloud Computing (HPCC) [8].

Developing a tool that can predict race winners and provide strategic recommendations could be invaluable for F1 teams [6] [5] [3].

This project report aims to use Machine Learning (ML) techniques to develop a model which predicts F1 race winners and provides optimal strategic decisions and recommendations, such as number of pit stops and tire selections, to improve there chances of winning the race. Also, determining the winner of the race also comes in as another application we can exploit with the use ML [5] [3].

2. Literature Survey

Recent studies have investigated the application of machine learning (ML) and statistical techniques in predicting Formula 1 (F1) race outcomes.

Van Kesteren and Bergkamp in 2022 employed a regression model, finding that 86% of race results are influenced by the constructor, with only 14% attributed to driver skill, underscoring the critical role of car performance [4]. Similarly, Bell et al. in 2016 used random-coefficient models to highlight that team performance is more decisive than driver skill, though driver influence becomes significant in challenging conditions such as wet races [1].

Thabtah and Bunker in 2019 demonstrated the efficacy of Artificial Neural Networks (ANN) in sports predictions, where they outperformed expert predictions. Their work emphasized the importance of feature selection and model validation for achieving high prediction accuracy [2]. Sicoie in 2022 extended this by using Support Vector Machines (SVM), Random Forest, and Gradient Boosting, leveraging High-Performance Cloud Computing (HPCC) for real-time data processing, which improved predictions of race winners and championship standings [8].

Addressing the complexities of sports predictions, including factors like pit stops, several studies have utilized ensemble learning and cross-validation techniques to ensure model robustness and prevent overfitting [8] [2]. These methods, combined with real-time telemetry data and historical race performance, form a strong foundation for accurate F1 race outcome predictions and strategic insights.

3. Dataset Details

The dataset is obtained from Kaggle [7] and consists of all F1 data from 1950-2024. The dataset includes files corresponding to drivers, constructors, qualifying, circuits, lap times, pit stops, and results of all F1 races over the years. The dataset consists of data compiled from Ergast [9], which is a compilation of F1 data.

The dataset contains details about **79** circuits where F1 has conducted its races, **215** constructors, and **860** drivers who participated in the **1144** races conducted since 1950. The data also consists of results of the races, having **26524** entries consisting of

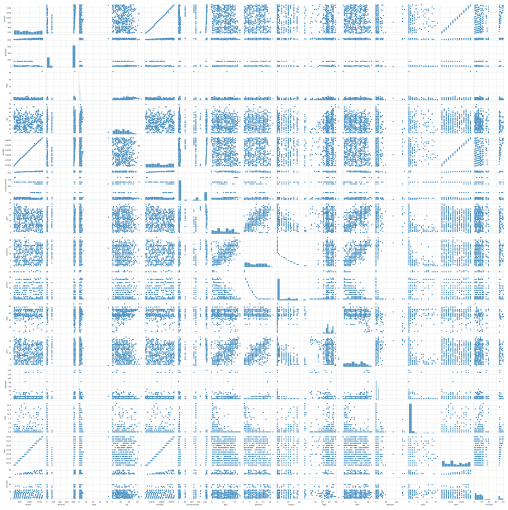

Figure 1. Pair Plot for the dataset of Pit strategy Model

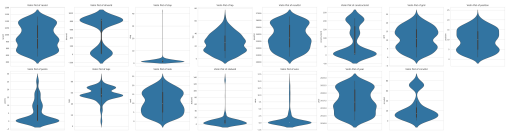

Figure 2. Violin Plot for the dataset of Pit strategy Model

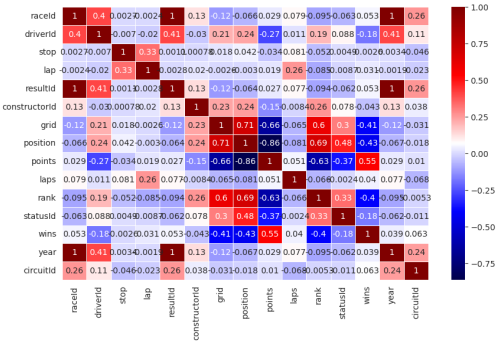

Figure 3. Correlation Plot for the dataset of Pit strategy Model

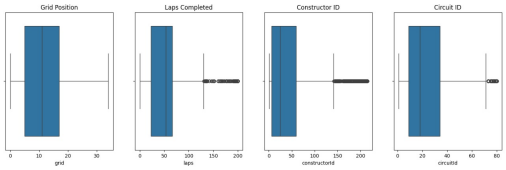

Figure 4. Box plot and outlier detection for winner-predictor model

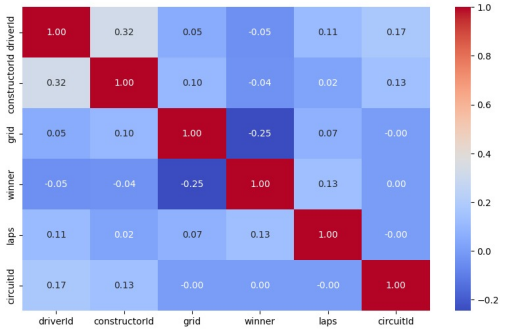

Figure 5. Correlation plot for dataset of winner prediction model

every driver of every race, giving details about their points and positions attained in each and every race they participated in since 1950, along with the number of pitstops taken by each driver and their duration in milliseconds and the number of laps completed by the drivers in the races with the time taken to complete each lap. We also have the details of the points attained by the constructors and drivers in **1132** of the total **1144** races conducted. The dataset will be used to analyze the performance details of various constructors and drivers over the years.

The data was pre-processed initially to make it fit for model training. The columns that were not required for model training, like the URL, car number, or string entries, were dropped from all the files. Several columns required conversion to appropriate data types. For instance, time-related fields (lap times, pit stop durations) stored as strings were converted into numerical formats like milliseconds for uniformity and ease of comparison. The positions were converted to a binary value indicating the winner or not. Since every race has only one winner and other non-winners, the dataset becomes quite imbalanced in terms of the output variables, with a ratio of around 25:1 for non-winners to winners.. To standardize the data for algorithms sensitive to feature scaling, the variables were normalized using standardization, ensuring that no variable solely determines the outcome of the race. We used a train-test split of 80:20. The data was visualized through various plots like Box plots, histograms, and violin plots to detect outliers. Outliers were detected using boxplots and the **IQR** method where values falling below the lower bound (Q1 - 1.5 x IQR) or above the upper bound (Q3 + 1.5 x IQR), where Q1 is the value below which 25% of the data distribution lies, whereas Q3 is the value below which 75% of the data distribution lies, were considered outliers. These extreme values, particularly in features such as lap times, pit stop durations, and race completion times, were removed.

The datasets were merged accordingly, and the main focus was to get the optimal number of pit stops and predict the race winners.

4. Methodology

4.1. Pipeline

Our approach employs a sophisticated two-stage pipeline aimed at accurately predicting race winners by identifying the optimal pit strategy. This comprehensive methodology integrates various influential factors, including track characteristics, race ID, and other relevant features, which will be elaborated upon in subsequent sections. In the first stage, we focus on the Pit Stop Strategy Predictor, which is crucial for determining the ideal timing and approach for pit stops during a race. This stage leverages a

Multi-Layer Perceptron (MLP) architecture, specifically designed to output scores ranging from 10 to 25, effectively ranking different pit strategies. Our tech stack is rooted in PyTorch and its sub-modules, allowing for a flexible and robust development environment

4.2. Pit Strategy Predictor Model

The updated Multi-Layer Perceptron (MLP) architecture improves classification performance through additional layers, batch normalization, and dropout. These changes promote better generalization, stabilization of gradients, and effective feature learning.

4.2.1 Architecture

The MLP now comprises six hidden layers instead of four, with the final layer outputting eight logits for multi-class classification. The architecture uses the ELU activation function to avoid vanishing gradients, and BatchNorm layers to stabilize training and speed up convergence.
The input layer processes one-hot-encoded Circuit ID and Race ID, concatenated with numerical features such as points, total race laps, and final position. The six hidden layers use the following configuration, with layer sizes as $[256 \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 8]$. Batch Normalization after each layer reduces covariate shifts. Dropout with a probability of 0.3 prevents overfitting by regularizing the model. ELU activation introduces non-linearity while mitigating gradient vanishing. The output layer produces eight logits, corresponding to multi-class pit stop strategies. The raw logits are fed into a softmax function during inference for probability-based classification.

4.2.2 Training Methodolgy

The training process leverages multi-class cross-entropy loss and the Adam optimizer with a learning rate of 0.01 and weight decay of 1e-4. Early stopping with a patience of 15 epochs is applied to prevent overfitting. Validation Loss, Macro and Micro Accuracy and R^2 and F1 Scores were tracked.

4.3. Winner-predictor model

The model is expected to give a winner prediction given the circuit on which the race is being held, the total laps of the race, the drivers participating in the race, and their grid positions. The optimal number of pit stops would be determined using the Pit Strategy Model. Our model answers, "Who is the predicted winner given the circuit ID, grid position, drivers in the race, assuming the optimal number of pit stops taken by the drivers".

4.3.1 Input Data transformation

In order to attain the configuration that can be used to train the model, the irrelevant columns from the dataset were dropped and the categorical data into **one-hot encoded label**. A one-hot encoded label represents each unique category as a binary vector, where only one element is "1" (indicating the presence of that category), and the rest are "0". This transformation ensures that categorical variables are converted into a numerical format that machine learning models can interpret without implying any ordinal relationship between the categories. We construct the one-hot encoded labels of constructors, drivers, and circuits in order to represent the presence of different drivers and/or circuits.

4.3.2 Models

We observed the correlation between data, and they displayed a weak correlation with respect to each other. The correlation with the dependent variable was quite decent; for example, the grid position showed a 0.30 negative correlation with winner-prediction. Since we needed a binary classification and other factors seemed to settle well, we chose to apply Logistic regression. Now, since the dataset is quite imbalanced with respect to the winner class and non-winner class, we used the weighted logistic regression, which assigns weights to the classes in inverse proportions to their frequencies. Another method employed to overcome the class imbalance is **the SMOTE - Synthetic Minority Oversampling Technique**. SMOTE uses the concept of **KNNs** to generate data on minority classes. SMOTE could only be employed on the training data. Decision trees and ensemble methods like random forests, MLP classifiers, and XG Boost could not be employed due to the lack of training samples. These methods showed great results on training sets, indicating **overfiting**. We also used GridSearch for finding the optimal hyper-parameters. Since the dataset was small after the irrelevant information was removed, the creation of a separate validation set was not optimal. To overcome this, we used the K-fold cross-validation to access the model's performance on different folds of unseen data. The model showed a consistent performance (see Results Section). The metrics were plotted accordingly. Confusion matrices were also visualized to understand the model better. Learning Curves were plotted to assess the model's predictions with varying training sample sizes and cross-validation sets.

5. Results and Analysis

The updated architecture for pit stop prediction demonstrates improved performance as shown below:

Epochs Trained	52 (early stopping triggered)
Validation Loss	0.3585
Macro Accuracy	84.43%
Micro Accuracy	84.43%
R^2 Score	0.7594
F1 Macro Score	0.5048
F1 Micro Score	0.8443

Table 1. Model Performance Metrics

Figure 7, Figure 8, and Figure 9 show the plot of metrics for Logistic Regression without minority oversampling. The K-fold cross-validation shows consistent performance of f1-score between 0.39 and 0.42 on the train set. The learning curve shows a little below-average performance considering the classes are imbalanced. However, Figure 10, Figure 11, and Figure 12 show the curves for Logistic Regression with SMOTE. The f1-score rises, but the precision and recall suffer. The learning curve here implies

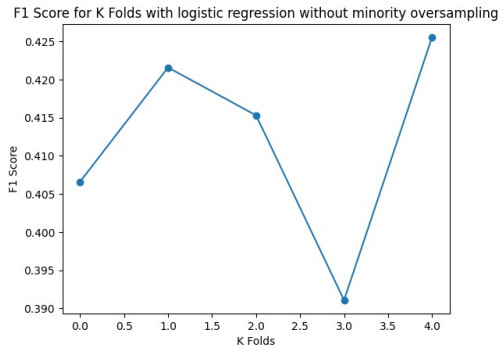

Figure 6. Fl Score for K Folds with logistic regression without minority oversampling

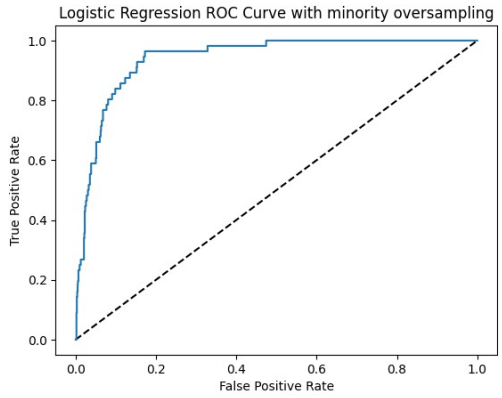

Figure 7. Logistic Regression ROC Curve with minority oversampling

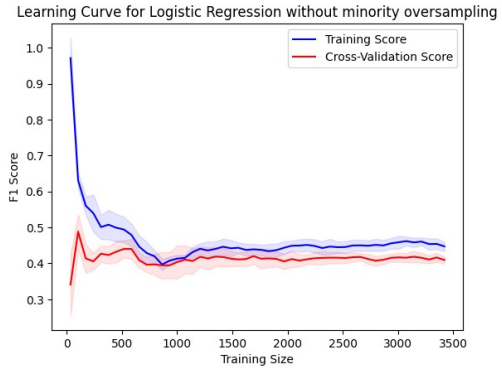

Figure 8. Learning Curve for Logistic Regression without minority oversampling

that the model generalizes better on a large training size than a smaller one.

The loss consistently declined during the early epochs, with validation loss stabilizing around epoch 20. Early stopping ensured no overfitting and the Micro Accuracy reflects strong overall classification ability.

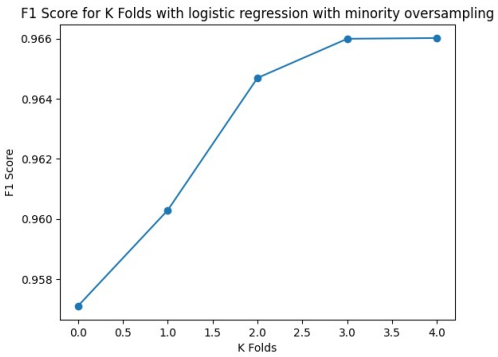

Figure 9. F1 Score for K folds with logistic regression with minority oversampling

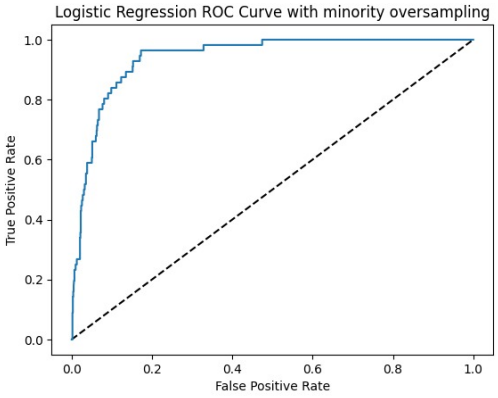

Figure 10. Logistic Regression ROC Curve with minority oversampling

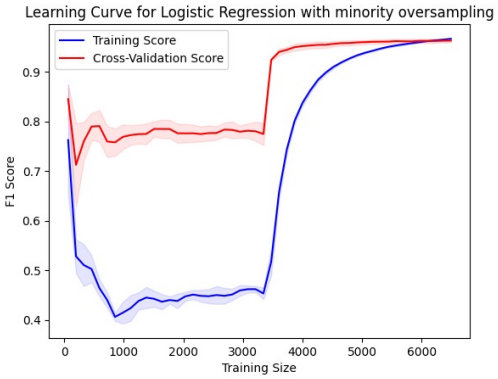

Figure 11. Learning Curve for Logistic Regression with minority oversampling

Metric for Winner-predictor model using logistic regression: Training accuracy: 0.88, precision: 0.22, recall: 0.96, F1-score: 0.36 Testing accuracy: 0.87, precision: 0.21, recall: 0.87, F1-Score: 0.34

	precision	recall	f1-score	support
0	1.00	0.86	0.92	1014
1	0.27	0.93	0.42	56
accuracy			0.87	1070
macro avg	0.63	0.90	0.67	1070
weighted avg	0.96	0.87	0.90	1070

Table 2. Results of Logistic Regression Model without Minority Sampling on Test Data

	Precision	Recall	F1-Score	Support
0	0.98	0.94	0.96	1014
1	0.38	0.64	0.48	56
Accuracy			0.93	1070
Macro Avg	0.68	0.79	0.72	1070
Weighted Avg	0.95	0.93	0.94	1070

Table 3. Results of Logistic Regression Model with SMOTE on Test Data

```
The optimal number of pitstop is: 3
The predicted winner is Driver ID: 8
```

Figure 12. Final output

6. Conclusion

Overall, the model is now capable of determining the winner on a circuit track provided the optimal number of pitstops.

6.1. Pit Stop Count Predictor Model

The updated MLP architecture, with added layers, batch normalization, and dropout, significantly improved the model's ability to generalize and classify pit stop strategies. Metrics such as validation loss and accuracy highlight robust performance, confirming the viability of our approach.

6.2. Winner-predictor model

Even though the logistic regression results in an accuracy of 87%, accuracy is not an appropriate measure because of class imbalance. It shows a high recall of 0.93 but a low precision of 0.27 leading to an overall f1-score of 0.42. It implies that the model correctly predicts 93% of the winners classifies non-winners as winners 73% of the time. Along with that, we see how introducing synthetic minority oversampling does not significantly affects the performance but insteads downgrades it a little. Employing other complex models lead to overfitting because of the small dataset and vanilla Logistic Regression with balanced class weights outperformed all other models.

6.3. Learnings

We learned about data piping, exploratory data analysis (EDA), data cleaning, and merging, which enhanced our skills with libraries like Matplotlib, PyTorch, and Scikit-learn, along with data processing using Pandas. During our exploration of handling imbalanced data, we faced challenges that were particularly evident in the F1 macro scores, emphasizing the need to assess performance across all classes rather than focusing solely on accuracy. Implementing the Exponential Linear Unit (ELU) activation function in the MLP model provided insights into how different activation functions can affect convergence and performance in deep learning. We also compared the performance of MLP and Support Vector Machine (SVM) models, gaining a clearer understanding of the strengths and weaknesses of deep learning compared to traditional machine learning techniques. Fine-tuning hyperparameters in the MLP demonstrated the importance of model architecture in assessing the loss and accuracy of the validation. Finally, our exploration of evaluation metrics such as accuracy, F1 score (macro and micro), precision, and recall deepened our understanding of their relevance in multiclass classification problems. For logistic regression, we learned how to deal with imbalanced datasets using the concept of assigning class weights.

6.4. Team Work

Our team collaborated equally and effectively, working together to compile the results for the final pipeline.

References

[1] Andrew Bell, Philip Smith, Clive Sabel, and Kelvyn Jones. Formula for success: Multilevel modelling of formula one driver and constructor performance, 1950-2014, 2016. Unpublished manuscript. 1

[2] Rory Bunker and Fadi Thabtah. A machine learning framework for sport result prediction. *Applied Computing and Informatics*, 15(1):27–33, 2017. 1

[3] Mohamed Hanafy and Ruixing Ming. Machine learning approaches for auto insurance big data. *Risks*, 9(2):42, 2021. Open Access Article. 1

[4] Erik-Jan van Kesteren and Tom Bergkamp. Bayesian analysis of formula one race results: disentangling driver skill and constructor advantage. *Journal of Quantitative Analysis in Sports*, 19(4):273–293, 2023. Open Access Published by De Gruyter July 25, 2023. 1

[5] Ł. Neumann, R. M. Nowak, R. Okuniewski, and P. Wawrzyński. Machine learning-based predictions of customers' decisions in car insurance. *Applied Artificial Intelligence*, 33(9):817–828, 2019. 1

[6] Musa Oytun, Cevdet Tinazci, Boran Sekeroglu, Caner Acikada, and Hasan Ulas Yavuz. Performance prediction and evaluation in female handball players using machine learning models. *IEEE Access*, 8:116321–116335, 2020. 1

[7] Rohan Rao. Formula 1 world championship (1950-2020). https://www.kaggle.com/datasets/rohanrao/formula-1-world-championship-1950-2020, 2020. Accessed: 2024-10-23. 1

[8] Sicoie. Machine learning framework for formula 1 race winner and championship standings predictor, 2022. 1

[9] Ergast Development Team. Ergast motor racing data api. http://ergast.com/mrd/, 2024. Accessed: 2024-10-23. 1