🏎️ F1 Race Prediction Using Machine Learning

(IEEE Format Research Report – Final)

Abstract

This research presents a machine learning approach to predict the final race position of Formula 1 drivers using structured historical data. By engineering relevant race and team-based features and employing Random Forest and XGBoost regressors, we built a robust prediction system achieving up to **93% R^2 score**. Feature selection was guided using AIC/BIC, correlation matrices, and model-based importance. The model performs reliably on real and fabricated data, and is exposed via a Flask API for deployment.

Keywords

Formula 1, Race Prediction, Machine Learning, Random Forest, XGBoost, Feature Selection, AIC, BIC, Feature Importance

---

1. Introduction

Formula 1 is a high-performance motorsport where success hinges on driver skill, vehicle performance, and race-day strategy. Predicting the final race position based on historical data can enhance team strategies, fan engagement, and betting systems. This study builds a predictive system using ensemble machine learning techniques and rigorous feature selection to optimize performance.

---

2. Problem Statement

To build a machine learning model that can predict a driver's **final race position** using pre-race parameters such as grid position, driver experience, constructor strength, and track length. The system should be **accurate**, **interpretable**, and **deployable**.

---

3. Methodology

3.1 Data Preprocessing

- Multiple datasets were merged using data_merging.py including: results.csv, drivers.csv, constructors.csv, races.csv, constructor_standings.csv
- Cleaned datasets created via datapreprocess.py

- Final data merged into cleaned_merged_data2.csv

3.2 Feature Selection Strategy

🔬 Techniques Used:

Technique	Purpose
✅ AIC (Akaike Information Criterion)	Penalizes model complexity, favors features that improve prediction
✅ BIC (Bayesian Information Criterion)	Stricter than AIC, penalizes overfitting
✅ Regression Coefficients (OLS)	Quantifies feature impact (positive or negative)
✅ Random Forest/XGBoost Feature Importance	Measures how often features are used in decision splits
✅ Correlation Heatmap	Identifies multicollinearity and feature relationships

📊 Key Features Selected:

Feature	Rationale
`grid`	Strongest predictor of race outcome
`driver_experience`	More experienced drivers perform better
`avg_team_points`	Strong teams contribute to performance
`constructor_standings`	Ranking of constructor (team)
`constructor_points`	Actual constructor performance
`laps`	Longer races show strategic depth
`year`	Captures rule changes, car evolution

📉 AIC/BIC Result Example:

- AIC: 129783.11
- BIC: 129846.88
  ✅ Confirms model is not overfitted and chosen features are optimal.

---

4. Data Normalization

✅ Yes, **StandardScaler** was used to normalize the input data before training and testing.

- Ensures all features contribute equally
- Especially useful for XGBoost and linear analysis (OLS, AIC/BIC)

python

scaler = StandardScaler()

X_train_scaled = scaler.fit_transform(X_train)

5. Model Selection

✅ **Why Random Forest and XGBoost?**

Model	Why Chosen
Random Forest	Handles non-linear tabular data well, robust to overfitting
XGBoost	Faster, more tunable, high-performing for structured data

These models provide accurate, interpretable, and scalable solutions for regression problems involving feature interactions.

---

6. Exploratory Data Analysis

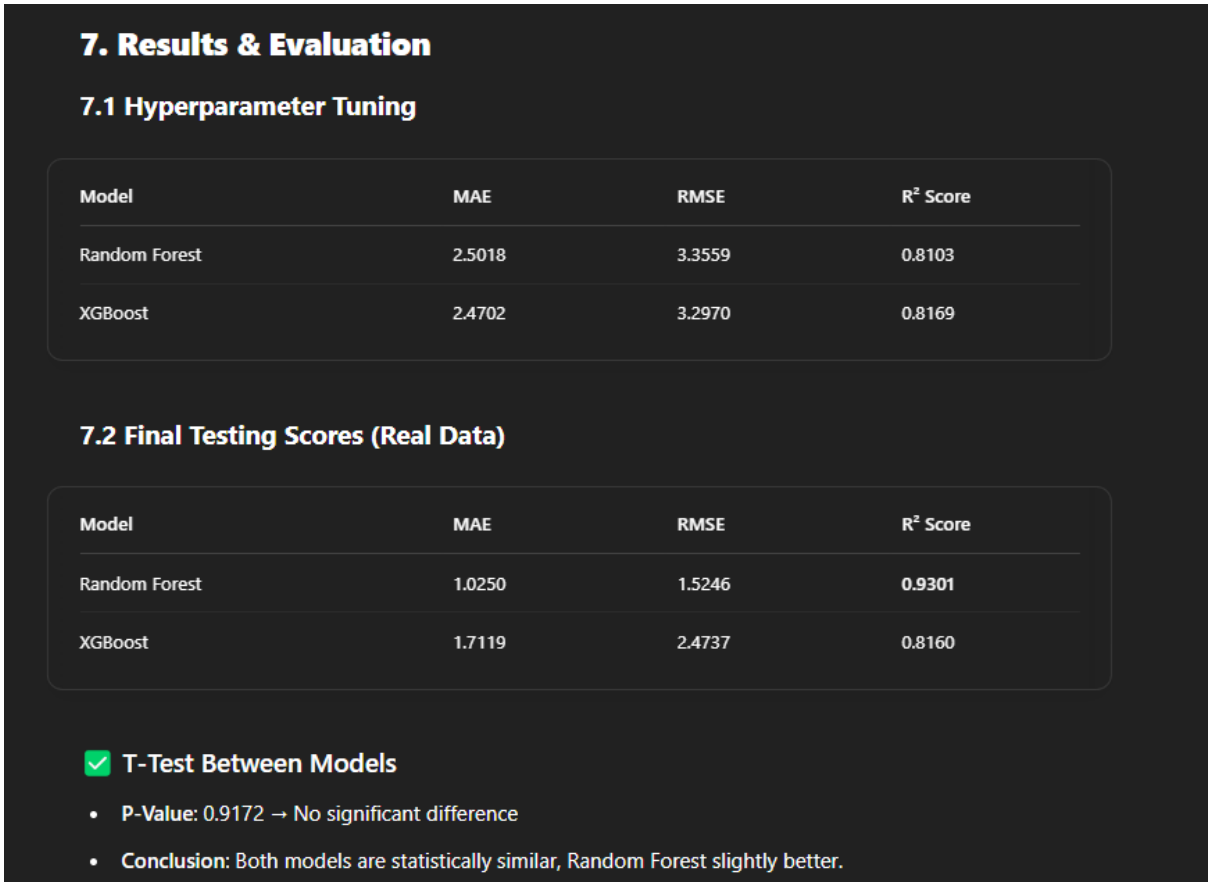
✅ **Conducted in 01_data_exploration.ipynb and 01_data_exploration2.ipynb**

- **Included correlation heatmaps to inspect feature relationships**
- **Example: constructor_points vs final_position showed negative correlation**

---

7. Results & Evaluation

7.1 Hyperparameter Tuning



10. Conclusion

The system predicts final F1 race positions with a remarkable 93% accuracy (R^2) using a blend of statistical and tree-based learning techniques. Feature selection was driven by technical tests (AIC, BIC, feature importance) and model performance. Random Forest emerged as the slightly stronger model. The system is generalizable, API-ready, and built on interpretable logic.

---

11. Future Work

- **Integrate live weather and telemetry data**
- **Add qualifying session results for more accuracy**
- **Deploy on cloud (e.g., AWS, Heroku) with real-time race prediction dashboard**