Formula 1 Prediction Challenge: 2024 Mexico Grand Prix

"We calmed the pace and turned down the engine to make sure we brought the car home so I was just cruising towards the end."

- Max Verstappen

2017, 2018, 2021, 2022, 2023 Mexico GP Winner

Yunus Gümüşsoy
yunusgumussoy@icloud.com
Ankara, Turkey

2024

Introduction

In this report, I present machine learning models designed to predict pit stop strategies for the 2024 Formula 1 Mexican Grand Prix. Using historical data from previous Mexican Grand Prix races and data from the entire 2024 F1 season, I aim to provide detailed insights into the models' approach, performance, and design across four critical areas.

1. Driver Performance and Variability

The models focus on predicting one of the most pivotal aspects of Formula 1 racing: pit stop strategies. With every second counting in a race, optimizing pit stops is essential to a driver's success. The models adapt to individual driver performance by factoring in numerous variables, such as '`Driver`' and '`GridPosition`'. Given the unique characteristics of the Mexico City Grand Prix—high altitude, long straights, and technical corners—understanding the impact of these elements on tire wear and pit stop decisions is critical.

The long straights and DRS zones (Drag Reduction System – '`DRS`') at this Grand Prix play a significant role in overtaking opportunities post-pit stop. The models consider these track-specific factors to predict how drivers can best time their pit stops for maximum advantage.

2. Adaptability to Race Conditions

Race conditions, both predictable and unpredictable, play a major role in strategy decisions. The models account for a variety of external variables such as weather (`AirTemp`, `Humidity`, `Pressure`, `TrackTemp`, `WindSpeed`, `Rainfall`) and race incidents (`TrackStatus`). Variables related to driver behavior (`Speed`, `Throttle`, `Brake`, `DRS`) and tire management (`TyreLife`, `FreshTyre`) are also considered. This ensures that the models can adapt to changing conditions and perform well in unexpected situations, such as weather changes or safety car interventions.

By training the models with these variables, they remain flexible and can adjust their predictions as race conditions evolve, making them highly resilient to outliers and dynamic race environments. Furthermore, by including these variables in model training, I allow them to be used as inputs during predictions, making the models adaptable to changing conditions throughout the race.

3. Innovative Methodology and Model Flexibility

I began by constructing a comprehensive and robust data processing pipeline that integrates historical data from previous Mexican Grand Prix races[1] with updated data from the 2024 F1 season.[2] This extensive dataset[3] allowed me to engineer key variables, such as the

[1] https://github.com/yunusgumussoy/Formula-1-Prediction-Challenge-2024-Mexico-Grand-Prix/blob/main/0-Formula1-2-data_merge-historical-race.ipynb
[2] https://github.com/yunusgumussoy/Formula-1-Prediction-Challenge-2024-Mexico-Grand-Prix/blob/main/1-Formula1-2-data_merge-2024-race.ipynb
[3] https://github.com/yunusgumussoy/Formula-1-Prediction-Challenge-2024-Mexico-Grand-Prix/blob/main/2-Formula1-2-data_merge-all.ipynb

Number of Stints, Tire Compounds Used, Laps per Stint, and Average Lap Time per Stint, and power the machine learning models.[4]

After thorough data preparation, I applied feature engineering based on domain knowledge to target critical aspects of pit stop strategies. Multiple machine learning models were trained and tested, using hyperparameter tuning and feature importance analysis to refine performance. Models were selected based on evaluation metrics like mean squared error and accuracy.[5]

The models are designed for flexibility, allowing individual driver inputs with various pre-race variables based on an individual driver's starting grid position and other pre-race information. These models can predict key race strategy elements such as the `Number of Stints`, `Used Tire Compounds`, `Laps per Stint`, and `Average Lap Time per Stint`. Detailed instructions for using the models are included to ensure they are practical and easy to implement.

4. Interpretability and Explanation of Predictions

To ensure both accuracy and usability, I selected XGBoost, LightGBM, and CatBoost for their superior performance in predicting key variables[6]. Each model, along with its training data and hyperparameters, is provided in this report. Models are saved in .pkl format with accompanying guidelines to ensure proper input structure and evaluation.[7]

By offering clear explanations of the model outputs and how they are influenced by various race factors, I aim to make these predictions both interpretable and actionable for race strategists and evaluators.

[4] https://github.com/yunusgumussoy/Formula-1-Prediction-Challenge-2024-Mexico-Grand-Prix/blob/main/3-Formula1-2-model-data.ipynb
[5] https://github.com/yunusgumussoy/Formula-1-Prediction-Challenge-2024-Mexico-Grand-Prix/blob/main/4-Formula1-2-feature-model-selection.ipynb
[6] https://github.com/yunusgumussoy/Formula-1-Prediction-Challenge-2024-Mexico-Grand-Prix/blob/main/5-Formula1-2-final-report-codes.ipynb
[7] https://github.com/yunusgumussoy/Formula-1-Prediction-Challenge-2024-Mexico-Grand-Prix/blob/main/6-Formula1-2-model-predictions.ipynb

Data Merge, Process and Feature Selection

Reliable data is the foundation of any statistical analysis, especially for predictive models. With that in mind, I focused on creating a detailed and accurate dataset by meticulously merging information from previous Mexican Grand Prix races with data from the entire 2024 F1 season. While the available data were quite comprehensive, offering information down to almost every second of the races, the primary challenge was ensuring a seamless merge, as the original data collection of each dataset followed different units of analysis.

To address this, I applied various merging strategies. First, I used the 2024 lap dataset as the base, as it provides the most granular information—down to event, driver, and second. Using this dataset, I employed a left-join merge to incorporate additional data at the event-driver-minute level. For historical datasets, I used a vertical merge, appending additional cases while keeping the variables constant.

The result was a unified dataset with observations at the event-driver-minute level, laying the groundwork for predictive analysis at both the event and driver levels. After merging the datasets, I also cleaned up certain columns to remove duplicates, fill in missing information, and eliminate unnecessary variables, ensuring a clearer dataset for the analysis ahead.

Code Snippet 1. Data Merge.

```
# lap_data - Extract days and hours
race_lap_data['Days'] = race_lap_data['Time'].apply(lambda x: int(x.split(' ')[0]))
race_lap_data['Hours'] = race_lap_data['Time'].apply(lambda x: int(x.split(' ')[2].split(':')[0]))
race_lap_data['Minutes'] = race_lap_data['Time'].apply(lambda x: int(x.split(' ')[2].split(':')[1]))

# weather_data - Extract days and hours
race_weather_data['Days'] = race_weather_data['Time'].apply(lambda x: int(x.split(' ')[0]))
race_weather_data['Hours'] = race_weather_data['Time'].apply(lambda x: int(x.split(' ')[2].split(':')[0]))
race_weather_data['Minutes'] = race_weather_data['Time'].apply(lambda x: int(x.split(' ')[2].split(':')[1]))

# Remove duplicates in weather_data based on 'eventname' and 'hours'
race_weather_data1 = race_weather_data.drop_duplicates(subset=['Year', 'Hours', 'Minutes'])

# Now perform the left join
race_lap_data1 = pd.merge(race_lap_data, race_weather_data1, on=['Year', 'Hours', 'Minutes'], how='left')
```

As seen in Code Snippet 2, this merged dataset enabled the creation of key variables required for the predictive analysis, including the number of stints, tire compounds, laps per stint, and average lap time per stint. These main variables of interest here became the target outputs for the machine learning models that followed, providing the foundation for predicting pit stop strategies and performance metrics.

Code Snippet 2. Creating the Target Variables.

```
# Calculate Laps per Stint, Tire Compounds, and Average Lap Time per Stint
# Group by Driver, Stint, EventName, and Year to calculate for each driver and stint in each event
stint_analysis = merged_data.groupby(['Driver', 'Stint', 'EventName', 'Year']).agg(
    Number_of_Laps=('LapNumber', 'count'),  # Laps per Stint
    Used_Tire_Compound=('Compound', 'first'),  # Tire compound used in this stint
    Avg_Lap_Time=('LapTime', 'mean')  # Average lap time per stint
).reset_index()

# Calculate the number of stints for each driver in each event (Grand Prix)
stint_count = merged_data.groupby(['Driver', 'EventName', 'Year'])['Stint'].nunique().reset_index()
stint_count.columns = ['Driver', 'EventName', 'Year', 'Number_of_Stints']

# Merge the stint analysis with the original DataFrame
merged_data1 = pd.merge(merged_data, stint_analysis, on=['Driver', 'Stint', 'EventName', 'Year'], how='left')

# Merge the stint count with the original DataFrame
merged_data2 = pd.merge(merged_data1, stint_count, on=['Driver', 'EventName', 'Year'], how='left')
```

As noted above, during the data merging process, I encountered several columns with similar names or contents, requiring careful inspection to ensure proper alignment. I also identified columns with a high percentage of missing values, which were subsequently cleaned or removed as necessary to maintain data integrity and improve the quality of the merged dataset (Code Snippet 3).

Code Snippet 3. Data Cleaning.

```
# Drop columns with very high missingness and repeated data due to merge
merged_data.drop(['Deleted', 'FastF1Generated', 'IsAccurate', 'Message', 'IsPersonalBest', 'Q1', 'Q2', 'Q3','PitOutTime',
'PitInTime', 'DeletedReason', 'HeadshotUrl', 'Time_right2', 'Status_left3', 'Days_left3', 'Time', 'Status_right3',
'Days_right3','Time_y', 'Date_left', 'Source_left', 'Time_left', 'SessionTime_left', 'Days_left', 'Date_right',
'Status_left', 'Source_right', 'Time_right', 'SessionTime_right', 'Days_right', 'DriverNumber_right', 'BroadcastName',
'DriverId', 'TeamName', 'TeamColor', 'TeamId', 'FirstName', 'LastName', 'FullName', 'CountryCode', 'Time_left2',
'Status_right', 'LapStartDate', 'Days_x', 'Days_y'], axis=1, inplace=True)
```

Machine Learning Models Selection and Training

I performed extensive model testing using various feature groups, complemented by feature importance and correlation analysis. This allowed me to identify the most effective models and key features that had the greatest impact on predictions, optimizing the overall performance and accuracy of the machine learning models.

Model 1 - Number of Stints

Code Snippet 4. Feature Selection.

```
feature_columns = ['EventName', 'Driver', 'GridPosition', 'Year', 'TyreLife', 'FreshTyre', 'TrackStatus', 'Rainfall',
                   'Speed', 'Throttle', 'Brake', 'DRS', 'ClassifiedPosition', 'AirTemp', 'Humidity', 'Pressure',
'TrackTemp', 'WindSpeed']
```

Table 1. Model Comparison.

Model 1 – Number of Stints	Mean Squared Error
XGBoost	**0.00003715100858079041**
LGBM	0.00006130215285626466
Catboost	0.0015488171162580046

Code Snippet 5. Model Design – XGBoost.

```
# Model 1: Predicting Number_of_Stints (Regression) - XGBoost
model_stints_xgb = XGBRegressor(n_estimators= 3500,learning_rate= 0.05,max_depth= 5, objective='reg:squarederror',
                                eval_metric= 'rmse', enable_categorical=True, random_state=42)
model_stints_xgb.fit(X_train_stints, y_train_stints)
y_pred_stints_xgb = model_stints_xgb.predict(X_test_stints)

stints_mse_xgb = mean_squared_error(y_test_stints, y_pred_stints_xgb)
stints_mse_xgb
```

Model 2 - Used Tire Compounds

Code Snippet 6. Feature Selection.

```
feature_columns = ['EventName', 'Driver', 'GridPosition', 'Year', 'TyreLife', 'FreshTyre', 'TrackStatus', 'Rainfall',
                   'Speed', 'Throttle', 'Brake', 'DRS', 'ClassifiedPosition', 'AirTemp', 'Humidity', 'Pressure',
'TrackTemp', 'WindSpeed']
```

Table 2. Model Comparison.

Model 2 – Used Tire Compound	Accuracy
LGBM	**0.9643478260869566**
Catboost	0.9414492753623188

Code Snippet 7. Model Design – LightGBM.

```
# Model 2: Predicting Used_Tire_Compound (Classification) - LightGBM

# Define and train the LightGBM classifier
model_tires_lgbm = LGBMClassifier(n_estimators= 3500,learning_rate= 0.08,max_depth= 5,
objective='multiclass',random_state=42,
                                  verbose=-1) # verbose=-1 disable warnings
model_tires_lgbm.fit(X_train_tires, y_train_tires)

# Make predictions
y_pred_tires_lgbm = model_tires_lgbm.predict(X_test_tires)

tires_accuracy_lgbm = accuracy_score(y_test_tires, y_pred_tires_lgbm)
tires_accuracy_lgbm
```

Model 3 - Laps per Stint

Code Snippet 8. Feature Selection.

```
feature_columns = ['EventName', 'Driver', 'GridPosition', 'Year', 'TyreLife', 'FreshTyre', 'TrackStatus', 'Rainfall',
                   'Speed', 'Throttle', 'Brake', 'DRS', 'ClassifiedPosition', 'AirTemp', 'Humidity', 'Pressure',
'TrackTemp', 'WindSpeed']
```

Table 3. Model Comparison.

Model 3 – Lap Number per Stint	Mean Squared Error
XGBoost	**6.987340431963285**
LGBM	7.385198664031037
Catboost	15.026561090649384

Code Snippet 9. Model Design – XGBoost.

```
# Model 3: Predicting Number_of_Laps (Regression) - XGBoost
model_laps_xgb = XGBRegressor(n_estimators= 3500,learning_rate= 0.05,max_depth= 5, objective='reg:squarederror',
                              eval_metric= 'rmse', enable_categorical=True, random_state=42)
# n_estimators= 3500,learning_rate= 0.05,max_depth= 5, objective='reg:squarederror', eval_metric= 'rmse',
enable_categorical=True, random_state=42
model_laps_xgb.fit(X_train_laps, y_train_laps)
y_pred_laps_xgb = model_laps_xgb.predict(X_test_laps)

laps_mse_xgb = mean_squared_error(y_test_laps, y_pred_laps_xgb)
laps_mse_xgb
```

Model 4 - Average Lap Time per Stint

Code Snippet 10. Feature Selection.

```
feature_columns = ['EventName', 'Driver', 'GridPosition', 'Year', 'TyreLife', 'FreshTyre', 'TrackStatus', 'Rainfall',
                   'Speed', 'Throttle', 'Brake', 'DRS', 'ClassifiedPosition', 'AirTemp', 'Humidity', 'Pressure',
'TrackTemp', 'WindSpeed']
```

Table 4. Model Comparison.

Model 4 – Average Lap Time	Mean Squared Error
XGBoost	**0.566850452081473**
LGBM	0.6029215315678983
Catboost	0.5919152330647612

Code Snippet 11. Model Design – LightGBM.

```
# Model 4: Predicting Avg_Lap_Time (Regression) - XGBoost
model_avg_lap_time_xgb = XGBRegressor(n_estimators= 3500,learning_rate= 0.05,max_depth= 5, objective='reg:squarederror',
                            eval_metric= 'rmse', enable_categorical=True, random_state=42)
model_avg_lap_time_xgb.fit(X_train_avg_lap_time, y_train_avg_lap_time)
y_pred_avg_lap_time_xgb = model_avg_lap_time_xgb.predict(X_test_avg_lap_time)

avg_lap_time_mse_xgb = mean_squared_error(y_test_avg_lap_time, y_pred_avg_lap_time_xgb)
avg_lap_time_mse_xgb
```

Throughout the rigorous process of model selection, feature engineering, and hyperparameter selection and tuning, I identified the best-performing combinations of models and parameters. Once optimized, I saved the final models to be used for future predictions, ensuring efficient and accurate results when applied to new data.

Code Snippet 12. Saving the Best Models.

```
# Saving the models
import joblib

joblib.dump(model_stints_xgb, 'model_stints.pkl')
joblib.dump(model_tires_lgbm, 'model_tires.pkl')
joblib.dump(model_laps_xgb, 'model_laps.pkl')
joblib.dump(model_avg_lap_time_xgb, 'model_avg_lap_time.pkl')
```

Model Predictions

Input Preparation

In the machine learning models, it is crucial that the input data for predictions maintains the same structure as the data used during training, including both the number of features and their exact names. This ensures consistency, as all features used to train the model must also be present when making predictions, guaranteeing accurate and reliable outputs.

Code Snippet 13. Example Input DataFrame.

```
test_input = pd.DataFrame({
    'EventName': ['Mexican Grand Prix'],     # Categorical - event name
    'Driver': ['VER'],                       # Categorical - driver name
    'GridPosition': [1],                     # Integer - starting position
    'Year': [2024],                          # Categorical - race year
    'TyreLife': [1],                         # Integer - tire age in laps
    'FreshTyre': ['True'],                   # Boolean - fresh tires (True means fresh)
    'TrackStatus': [1],                      # Categorical - track status (1: clear)
    'Rainfall': ['False'],                   # Boolean - rainfall (True means rainy)
    'Speed': [185.0],                        # Float - car speed in km/h
    'Throttle': [75],                        # Integer - throttle percentage
    'Brake': ['False'],                      # Boolean - brake applied (False: no)
    'DRS': [1],                              # Categorical - DRS active (1: yes)
    'ClassifiedPosition': [1],               # Categorical - current race position
    'AirTemp': [10],                        # Float - air temperature in Celsius
    'Humidity': [80.0],                     # Float - humidity percentage
    'Pressure': [1022.0],                   # Float - air pressure in hPa
    'TrackTemp': [12],                      # Float - track temperature in Celsius
    'WindSpeed': [5.5]                      # Float - wind speed in km/h
})
```

For weather conditions, I utilized forecasts specific to Mexico City's Autódromo Hermanos Rodríguez for October 27, 2024. The meteorological data was sourced from Ventusky,[8] a platform that visualizes weather data and relies on trusted providers such as the National Oceanic and Atmospheric Administration (NOAA), the U.S. Department of Commerce, and the Deutscher Wetterdienst (DWD). Additionally, I cross-referenced forecasts with Windy,[9] another reliable weather prediction source, to ensure accuracy and comprehensive coverage of expected conditions on race day. Weather forecasting can be seen in Figure 1.

[8] https://www.ventusky.com/?p=19.56;-97.07;7&l=clouds-total&t=20241027/1200
[9] https://www.windy.com/19.402/-99.093?19.402,-99.091,16

Figure 1. Weather Forecasting of Mexico City on October 27, 2024.

Weather Forecasting of Mexico City on October 27, 2024

Air Temperature: 10 °C

Track Temperature: 12 °C

Air Pressure: 1022 hPa

Wind Speed: 5.5 km/h

Precipitation: 0 mm

Humidity: %80

Clouds*: %50

* We can reasonably expect that track temperatures will be similar to air temperatures due to the forecasted 50% cloud cover, which will limit direct sunlight exposure.

While preparing the input dataframe, I convert the column types to the ‘category’ dtype to ensure they align with the model training data (Code Snippet 14). This step is crucial for maintaining consistency between training and prediction phases. After this type conversion, the input dataframe is ready for accurate model predictions.

Code Snippet 14. Input Dataframe Process.

```
# Convert categorical columns to 'category' dtype
test_input['Driver'] = test_input['Driver'].astype('category')
test_input['EventName'] = test_input['EventName'].astype('category')
test_input['ClassifiedPosition'] = test_input['ClassifiedPosition'].astype('category')
test_input['FreshTyre'] = test_input['FreshTyre'].astype('category')
test_input['Brake'] = test_input['Brake'].astype('category')
test_input['Rainfall'] = test_input['Rainfall'].astype('category')
test_input['Year'] = test_input['Year'].astype('category')
test_input['TrackStatus'] = test_input['TrackStatus'].astype('category')
test_input['DRS'] = test_input['DRS'].astype('category')
```

How to Run Models and Make Predictions

Saved models can be loaded from the .pkl files and executed using the `predict()` function on the prepared input dataframe. To assess the impact of unpredictable race conditions, the `unique()` function can be applied (e.g., `data_cleaned['TrackStatus'].unique()`) to review the distinct values of each feature. By examining these feature values, further predictions can be made based on specific rare variables, allowing for a more flexible analysis of varying conditions.

Code Snippet 15. Load the Models.

```
# load the models
loaded_model_stints = joblib.load('model_stints.pkl')
loaded_model_tires = joblib.load('model_tires.pkl')
loaded_model_laps = joblib.load('model_laps.pkl')
loaded_model_avg_lap_time = joblib.load('model_avg_lap_time.pkl')
```

Code Snippet 16. Predictions.

```
# Predictions from loaded models
# Model 1 - Number of Stints
pred_stints = loaded_model_stints.predict(test_input)
pred_stints = round(pred_stints[0])  # Rounding the prediction

print(f'Predicted Number of Stints: {pred_stints}')

# Model 2 - Used Tire Compound
pred_tires = loaded_model_tires.predict(test_input)
print(f'Predicted Tire Compounds Used: {pred_tires}')

# Model 3 - Lap Number per Stint
pred_laps = loaded_model_laps.predict(test_input)
pred_laps = round(pred_laps[0])  # Rounding the prediction

print(f'Predicted Number of Laps within Each Stint: {pred_laps}')

# Model 4 - Average Lap Time per Stint
pred_avg_lap_time = loaded_model_avg_lap_time.predict(test_input)
print(f'Predicted Average Lap Time per Stint: {pred_avg_lap_time}')
```

Example Predictions

I conducted machine learning predictions based on three different scenarios for the 2024 Mexican Grand Prix ('EventName': ['Mexican Grand Prix'], 'Year': [2024]) to analyze variations in the *Number of Stints, Used Tire Compound*, *Lap Number per Stint*, and *Average Lap Time*. These predictions were influenced by factors such as grid position, tire situation, weather conditions, driving style, and track status. For the *Drivers*, I selected the top five performers from the 2023 Mexican Grand Prix: Max Verstappen, Lewis Hamilton, Charles Leclerc, Carlos Sainz, and Lando Norris. The values of the variables not mentioned remain the same as in the sample input. The results of these predictions are summarized in Table 5.

Variable Set 1:

'TyreLife': [1], 'FreshTyre': ['True'], 'TrackStatus': [1], 'Rainfall': ['False'], 'Speed': [185.0], 'Throttle': [75], 'AirTemp': [11], 'Humidity': [60.0], 'Pressure': [1024.0], 'TrackTemp': [12], 'WindSpeed': [5.5]

Variable Set 2:

'TyreLife': [1], 'FreshTyre': ['True'], 'TrackStatus': [12], 'Rainfall': ['True'], 'Speed': [285.0], 'Throttle': [85], 'AirTemp': [30], 'Humidity': [80.0], 'Pressure': [1024.0], 'TrackTemp': [35], 'WindSpeed': [5.5]

Variable Set 3:

'TyreLife': [10], 'FreshTyre': ['False'], 'TrackStatus': [2671], 'Rainfall': ['True'], 'Speed': [285.0], 'Throttle': [85], 'AirTemp': [30], 'Humidity': [80.0], 'Pressure': [990.0], 'TrackTemp': [35], 'WindSpeed': [1]

Table 5. Model Predictions based on Different Scenarios.

EventName	Driver	GridPosition	Variable Set	Number of Stints	Used Tire Compound	Lap Number per Stint	Average Lap Time
Mexican Grand Prix	VER	1	1	3	MEDIUM	11	82.41780395
Mexican Grand Prix	HAM	1	1	3	MEDIUM	11	81.91855652
Mexican Grand Prix	LEC	1	1	3	MEDIUM	11	82.41854681
Mexican Grand Prix	SAI	1	1	3	MEDIUM	11	82.31513734
Mexican Grand Prix	NOR	1	1	3	MEDIUM	11	82.45009852
Mexican Grand Prix	VER	5	2	3	SOFT	16	84.31891547
Mexican Grand Prix	HAM	5	2	3	SOFT	16	83.90631182
Mexican Grand Prix	LEC	5	2	3	SOFT	16	84.85476551
Mexican Grand Prix	SAI	5	2	3	SOFT	16	83.66040676
Mexican Grand Prix	NOR	5	2	3	SOFT	16	83.79946591
Mexican Grand Prix	VER	10	3	3	HARD	18	85.8893259
Mexican Grand Prix	HAM	10	3	3	HARD	18	85.56683711
Mexican Grand Prix	LEC	10	3	3	HARD	18	85.59624491
Mexican Grand Prix	SAI	10	3	3	HARD	18	85.24106702
Mexican Grand Prix	NOR	10	3	3	HARD	18	85.09507728

For Scenario 1, the drivers start from pole position (1st on the grid). They all use Medium tires, likely because it is optimal for a balanced performance over 11 laps per stint. All drivers are predicted to have 3 stints. Each driver is expected to complete 11 laps per stint. The predictions indicate relatively close lap times, suggesting that the drivers' performances are quite similar under these conditions. Hamilton has the fastest predicted average lap time (81.9185 seconds), with the others trailing by small margins.

In the case of starting from 5th position on the grid, Scenario 2 shows that all drivers are predicted to switch to Soft tires. Still, 3 stints per driver, but each driver completes 16 laps per stint on the softer tires. Lap times are slightly longer compared to Scenario 1, with Leclerc having the slowest time (84.85 seconds) and Hamilton posting one of the faster times (83.91 seconds). So, Hamilton is still predicted to perform relatively well, while Leclerc seems to have a slight disadvantage in lap time. The use of soft tires suggests that drivers may prioritize early speed to make up positions. The lap times are generally slower than in Scenario 1, likely due to the increased stint length (16 laps), which could cause tire degradation and slower lap times toward the end of the stint.

In Scenario 3, where each driver starts further back on the grid at 10th position, all drivers switch to Hard tires. Three stints remain constant and this time, drivers are expected to complete 18 laps per stint. The use of hard tires suggests a more conservative strategy, focusing on longer stints (18 laps) and durability. The average lap times are longer, ranging from 85.09 seconds for Norris to 85.88 seconds for Verstappen. The slower lap times reflect the harder tire compound and longer stint length. Verstappen is predicted to have the slowest average lap time in this scenario, possibly indicating that a hard tire strategy may not be ideal for him compared to others like Norris and Hamilton, who perform slightly better.

Overall, *Number of Stints* remains constant across all scenarios (3 stints), implying that this strategy is likely robust to grid position changes for the Mexican Grand Prix. *Tire Compound* selection and *Lap Number per Stint* vary with grid position: Medium tires with shorter stints for front-runners, Soft tires with medium stints for mid-pack drivers, and Hard tires with longer stints for those starting further back. The drivers exhibit similar performances in each scenario, with Hamilton consistently showing faster lap times compared to the others, and Verstappen generally showing slower times, particularly in Scenario 3.

Conclusion

In this report, I successfully developed machine learning models designed to predict critical pit stop strategies for the 2024 Formula 1 Mexican Grand Prix. By leveraging historical data from previous races and integrating it with current season's data, I provided valuable insights into the dynamics that shape effective race strategies.

The models focus on key race elements, such as the *Number of Stints*, *Used Tire Compounds*, *Laps per Stint*, and *Average Lap Time per Stint*. By incorporating variables related to driver performance, grid position, and the unique characteristics of the Mexico City Grand Prix—such as its high altitude and specific track features—I enhanced the accuracy of the predictions. A particular emphasis was placed on the timing of pit stops, especially concerning the long straights and DRS zones, which play a crucial role in race outcomes.

Adaptability to changing race conditions was another critical aspect of the models. By accounting for various external factors, including weather conditions and race incidents, I ensured that the models could effectively respond to unpredictable circumstances, such as sudden shifts in weather or the deployment of safety cars. This flexibility allows for real-time strategy adjustments during the race, equipping teams with a robust tool for optimizing performance.

An innovative methodology involved constructing a comprehensive data processing pipeline and employing feature engineering techniques to enhance the predictive capabilities of the models. I utilized advanced machine learning algorithms—specifically XGBoost, LightGBM, and CatBoost—to fine-tune the predictions, ensuring they are both accurate and interpretable. By providing clear documentation and guidelines for model implementation, I aimed to make the findings accessible and actionable for race strategists.

Ultimately, this work contributes to a deeper understanding of the intricacies of pit stop strategies in Formula 1 racing. The models developed not only offer valuable predictions but also set the stage for further enhancements and applications in future races. As I approach the 2024 Mexican Grand Prix, these insights will be vital for teams striving to optimize their strategies and gain a competitive edge.