Jason Merchan, jm2693
Yousef Naam, yn164

F1 Mini-Sector Analysis and Predictions

GitHub: https://github.com/jm2693/F1-ML

Objective

Predict F1 position outcomes based on practice sessions and qualifier rounds, in combination from historical data.

Problem and Strategic Aspects

- This project addresses the challenge of predicting Formula 1 championship outcomes using historical race data. The strategic complexity comes from F1's multifaceted nature. Success depends on driver skill, team performance, technical capabilities, and race conditions. The project connects to core data management concepts we discussed, particularly data integration, cleaning, and the ETL (Extract, Transform, Load) pipeline process. We're essentially building a sports analytics system that demonstrates how proper data management can drive predictive insights.

Novelty and Importance

The project's importance lies in its approach to F1 data management. While existing F1 prediction systems often focus solely on race results, our system integrates multiple data dimensions, such as race results, qualifying performance, driver standings, constructor standings, and even weather conditions (though not yet used in predictions). This approach addresses current limitations in F1 data analysis, where data often exists in silos.

As F1 fans, analyzing race data at this detailed level presents an opportunity to make meaningful contributions to motorsport data analytics. The excitement comes from creating a complete pipeline that transforms raw racing data into meaningful predictions. This reflects real-world challenges in sports analytics, where raw data must be carefully processed and integrated to derive insights. Prior works have typically focused on either pure race prediction or historical analysis, while our system combines both aspects.

Data Collection and Processing:

- We utilized the Ergast F1 API as our primary data source, collecting data from 2015-2020 including:
    - Basic race information (dates, locations, circuits)
    - Race results and driver performances
    - Driver and constructor standings
    - Qualifying results

Note: Most Kaggle datasets receive their F1 data from either the Ergast F1 or FastF1 API. We wanted to use the API directly for a primary source.

- The data collection process involved careful error handling and rate limiting to respect API constraints. We stored this data in a SQLite database using SQLAlchemy ORM, creating proper relationships between different data types.

Models and Techniques

- We implemented a Random Forest Classifier with specific configurations:
    - 200 estimators for robust predictions
    - Maximum depth of 10 to prevent overfitting
    - Class weight balancing to handle the imbalanced nature of F1 championships
    - Feature scaling to normalize different measurement scales
- The model uses 13 key features including total points, wins, podiums, and various performance metrics. Our feature engineering focused on creating meaningful aggregations of raw race data.

Experimental Design and Results

- Our experiments followed a temporal split, using pre-2019 data for training and 2019-2020 for testing. The model achieved impressive results:
    - 93% accuracy in predicting championship outcomes
    - Season-based precision (1.00) for championship predictions
        - In other words, our model selected one ‘champion’ as the sole champion for predictions, as designed.
    - High recall (0.93) for identifying champions
- The model's feature importance analysis revealed interesting insights:
    - Total points (24%) and wins (22%) are the strongest predictors
    - Points per race (16%) provides insight into consistency
    - Average final position (11%) captures overall performance

Advantages and Limitations:

- Advantages:
    - Comprehensive data integration from primary source
    - Robust database design for efficient data management
    - High prediction accuracy for championship outcomes
    - Interpretable results through feature importance analysis
- Limitations:
    - By using the Ergast API, we were subject to rate limiters which meant it took a very long time to collect racing data. Thus for most tests we use a smaller date range from 2015 - 2020, though this can be simply adjusted in the main.py file.

    - By using a shorter date range, the data is actually slightly overfitted. Again, this can be adjusted to provide a result with higher accuracy.
    - Model doesn't account for mid-season regulation changes or team developments.
    - F1 is a constantly-changing sport. Its rules and regulations have been in constant flux since the 1950s. This made cleaning the data especially important and made us take error handling with great care.

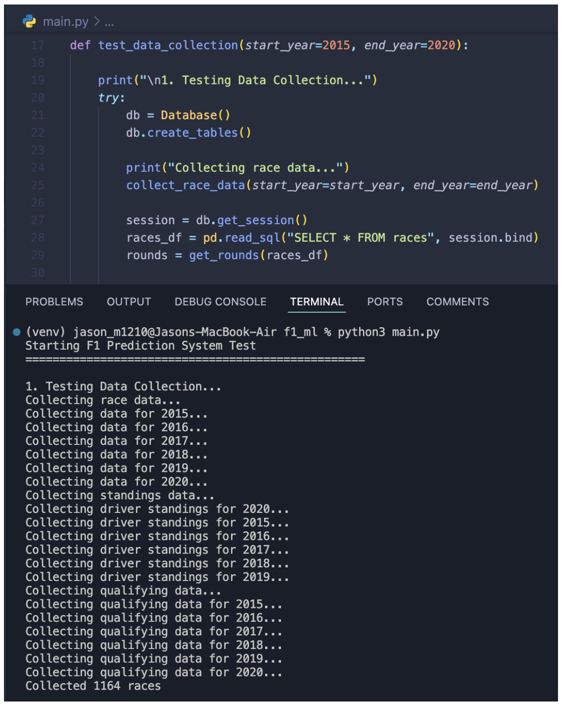
These findings generally supported our hypothesis that championship outcomes can be predicted from comprehensive race data. Future work could focus on incorporating weather data and developing more sophisticated feature engineering.

Generally our program was segmented into 4 aspects:

1. Data Collection
2. Data Cleaning
3. Model Training
4. Model Testing

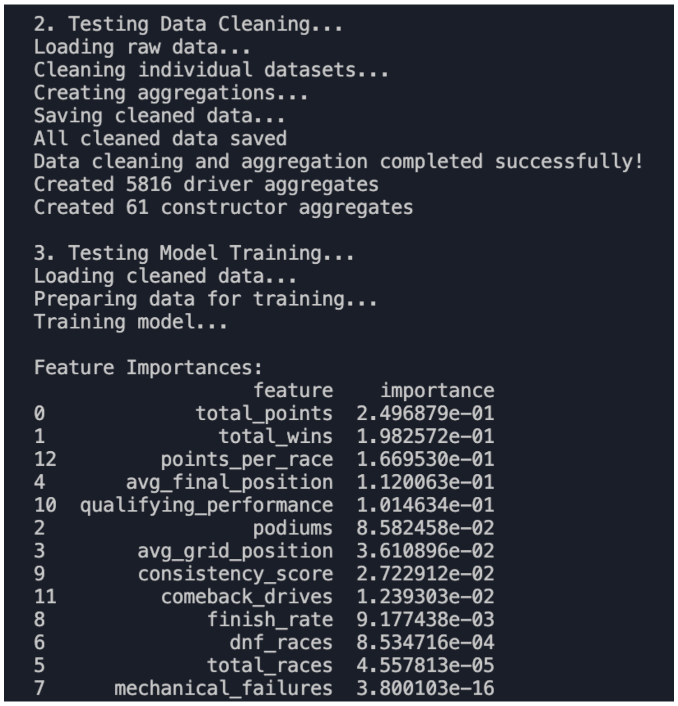
This can be followed from the main.py in the root directory of our program. By running the main.py file, we begin collecting race data at the specified year into a sqlite database. We then clean and merge this data, replacing the raw data. This data is split and used to train our random forest model. The testing split of our data is what we used to test our data, which in this case is the 2020 season. Ultimately it will display the likelihood of the predicted champions in a top 5 format. For 2020, the prediction was overwhelmingly Hamilton, which is true to the actual season.

These are some of the screenshots of the results of running our code from 2015 to 2020:

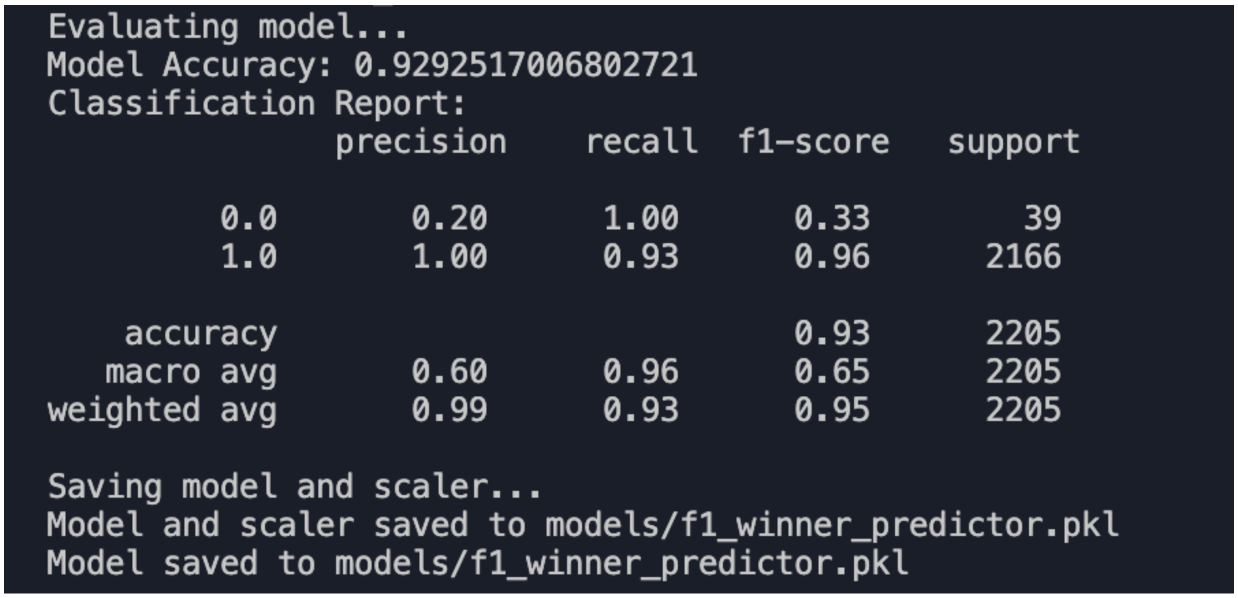


```python
def test_data_collection(start_year=2015, end_year=2020):

    print("\n1. Testing Data Collection...")
    try:
        db = Database()
        db.create_tables()

        print("Collecting race data...")
        collect_race_data(start_year=start_year, end_year=end_year)

        session = db.get_session()
        races_df = pd.read_sql("SELECT * FROM races", session.bind)
        rounds = get_rounds(races_df)

```

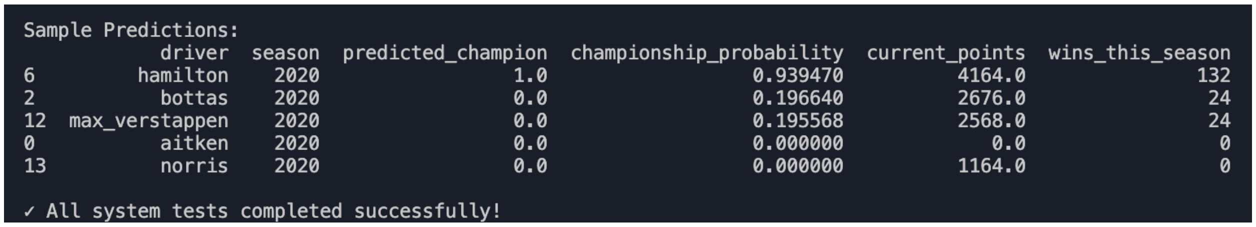
```
(venv) jason_m1210@Jasons-MacBook-Air f1_ml % python3 main.py
Starting F1 Prediction System Test
==================================================

1. Testing Data Collection...
Collecting race data...
Collecting data for 2015...
Collecting data for 2016...
Collecting data for 2017...
Collecting data for 2018...
Collecting data for 2019...
Collecting data for 2020...
Collecting standings data...
Collecting driver standings for 2020...
Collecting driver standings for 2015...
Collecting driver standings for 2016...
Collecting driver standings for 2017...
Collecting driver standings for 2018...
Collecting driver standings for 2019...
Collecting qualifying data...
Collecting qualifying data for 2015...
Collecting qualifying data for 2016...
Collecting qualifying data for 2017...
Collecting qualifying data for 2018...
Collecting qualifying data for 2019...
Collecting qualifying data for 2020...
Collected 1164 races
```



```
2. Testing Data Cleaning...
Loading raw data...
Cleaning individual datasets...
Creating aggregations...
Saving cleaned data...
All cleaned data saved
Data cleaning and aggregation completed successfully!
Created 5816 driver aggregates
Created 61 constructor aggregates

3. Testing Model Training...
Loading cleaned data...
Preparing data for training...
Training model...

Feature Importances:
                  feature    importance
0            total_points  2.496879e-01
1              total_wins  1.982572e-01
12        points_per_race  1.669530e-01
4      avg_final_position  1.120063e-01
10  qualifying_performance  1.014634e-01
2                 podiums  8.582458e-02
3       avg_grid_position  3.610896e-02
9       consistency_score  2.722912e-02
11         comeback_drives  1.239303e-02
8             finish_rate  9.177438e-03
6               dnf_races  8.534716e-04
5             total_races  4.557813e-05
7     mechanical_failures  3.800103e-16
```



```
Evaluating model...
Model Accuracy: 0.9292517006802721
Classification Report:
              precision    recall  f1-score   support

         0.0       0.20      1.00      0.33        39
         1.0       1.00      0.93      0.96      2166

    accuracy                           0.93      2205
   macro avg       0.60      0.96      0.65      2205
weighted avg       0.99      0.93      0.95      2205

Saving model and scaler...
Model and scaler saved to models/f1_winner_predictor.pkl
Model saved to models/f1_winner_predictor.pkl
```



```
Sample Predictions:
            driver  season  predicted_champion  championship_probability  current_points  wins_this_season
6         hamilton    2020                 1.0                  0.939470          4164.0               132
2           bottas    2020                 0.0                  0.196640          2676.0                24
12  max_verstappen    2020                 0.0                  0.195568          2568.0                24
0           aitken    2020                 0.0                  0.000000             0.0                 0
13          norris    2020                 0.0                  0.000000          1164.0                 0

✓ All system tests completed successfully!
```