Predicting Formula 1 Lap Times Using Regression Models and LSTM Networks

Srihari Meyoor

December 8, 2025

Contents

NOTE: Please check the `Math582` branch for the regression models and Release 1.0 for the LSTM results. While this is inconvenient, I do not have the time at the moment to make the adjustments for the `Math582` branch and run the full notebook as this required GPU usage to complete.

1 Introduction

This project investigates the problem of predicting Formula 1 lap times using both classical regression techniques such as Lasso and Ridge, and a modern sequence based model, the LSTM. The goal is to understand how different optimization architectures from least square to RNNs can capture the relationships of high frequency telemetry data throughout a race.

Using real telemetry collected from F1TV, the project examines data from British GP from 2022-2024 as the regulations allow the cars to be similar through the 3 year span. The project builds a pipeline that processes raw telemetry for each driver including speed, throttle, brake, RPM, tire compound, tire life, driver, and distance to driver ahead and more. It transforms it into structured normalized data suited for model training. Regression models are used as the baselines for evaluating lap times when applied to lap-level data.

The central feature of the project is the LSTM neural network that operates directly on sequential telemetry samples from each lap. Because each lap has around 1500 telemetry points over the lap, the LSTM is able to learn temporal patterns at the singular lap scope and the overarching race scope. This means that it understands interactions between telemetry points and interactions between consecutive laps. The model understands patterns such as braking zones, throttle usage, gear sequencing, and tire degradation which are not captured by the simple regression models. By comparing the performance of the LSTM and regression models the project shows how modeling decisions influence accuracy and capturing temporal patterns in real world racing data.

1.1 General Formula 1 Background

Formula 1 lap times result from a complex interaction of car performance, driver skill, tire behavior, race strategy and track conditions. Each race weekend provides teams with three types of dry weather tires–Soft, Medium, and Hard– which differ in grip and degradation. Softs are considered the fastest tire but won't last very long. The hards are the opposite, they don't have as strong pace as a soft tyre but they can last a lot longer almost two-thirds of the race at some tracks. Medium is a fresh middle ground of pace and degradation. It gives the drivers the most flexibility and whether they want to save the tyre and go longer or pick up the pace and catch those ahead. Generally, the softs are used during qualifying while the actual race is a mix of hard and medium tyres but it all is track dependent.

There are also 2 wet weather tyres–Intermediates and Extreme Wets. Intermediates are generally used throughout a race for wet weather, as the Extremes even in the worst conditions tend not to be used. The intermediate is like the Medium tyre: its a good

balance between pace and tyre degradation. The Extreme wets are extremely fast in the rain, but can only last 7 to 8 laps which is not good for long races because it means more pitstops.

Team performance also varies as some drivers have always been faster than others. This comes down to some skill but mainly the car they drive. Red Bull for 2022 and 2023 were very dominant as Max verstappen won convincingly in both seasons, but Mclaren was very strong in the 2024 season, while Max Verstappen still managed to retain his crown. In general the top 4 teams consistently are Red Bull, Mclaren, Mercedes, and Ferrari. The creating stronger cars due to higher budgets and better facilities. These teams are at the top regardless of racing conditions while the backmarker teams tend to have more variety as some are strong in the rain, while others are strong in extreme heat. These competitive and technical differences create good structure for telemetry data and make Formula 1 and gold mine for data scientists.

2 Theoretical Foundations

This project is influenced by several concepts covered in class including linear regression, regularization and optimization techniques used in solving high dimension problems.

2.1 Ordinary Least Squares Regression

Predicting F1 lap times can first be solved as a linear regression problem in the form of

$$Y = XB$$

Where each row of X represents the telemetry features and B is the parameter vector. OLS minimizes the sum of the squared residuals:

$$\min_b \ \|y - XB\|^2. \tag{1}$$

Properties used:

$$B = (X^T X)^{-1} X^T y$$

- connects to matrix inversion and operations
- This estimator is unbiased under standard assumptions
- OLS can overfit when features are correlated (this hurts with telemetry such as throttle and brake, speed and RPM)

This is why regularization is needed.

2.2 Regularization: Ridge and LASSO

Telemetry variables are tightly coordinated such as speed, RPM and gear sequences are all interdependent. Ridge and Lasso mitigates this through penalty terms:

Ridge uses L2 penalties

$$\min_{\beta} \ \|y - X\beta\|^2 + \lambda\|\beta\|_2^2$$

This reduces coefficients smoothly and is useful for data where independent variables are highly correlated.

Lasso uses L1 penalties

$$\min_{\beta} \ \|y - X\beta\|^2 + \lambda\|\beta\|_1$$

This helps select only the most important features and identifies which telemetry signals matter most for determining lap time.

2.3 Gradient-Based Optimization

Neural networks are trained with iterative optimization like SGD or Adam which updates parameters with:

$$\theta_{t+1} = \theta_t - \eta \, \nabla_\theta L(\theta_t)$$

- The gradient goes through back propagation through the sequential data (BPTT for short).
- This is required because each lap contains around 1500 sequential telemetry points.
- Adam uses adaptive learning rates instead of a static learning rate.

2.4 LSTM Architecture

LSTM is essentially a non-linear regression model with hidden layers. It's trained with optimization principles discussed in class. The use case is solving an high dimensional problem that a simple linear regression model can't solve.

LSTMs consist of:

- Forget gate: discards irrelevant past information allowing for certain bits of recency bias.
- Input gate: adds the new telemetry patterns.
- Output gate: figures out what will impact the final lap time prediction.

These allow the model to capture the temporal patterns discussed earlier which cannot not be recognized by a simple regression model.

3 Modeling and Application Plan

The coding portion of this project implements both regression models and an LSTM based model using real time telemetry data obtained from the FastF1 API. The coding work is organized into 3 major components: data collection, data cleaning/preprocessing and then modeling and optimization.

3.1 Data Collection

Using the FastF1 library I retrieved data for the British Grand Prix from 2022 to 2024. I chose drivers that remained consistent for those 3 years, and didn't change teams with a significant pace advantage, for example Haas, a team at the back of the grid, to Mclaren. That significant change could impact the time to get a consistent understanding of driver and car talent. These were the features collected:

- **DistanceToDriverAhead**
  Distance (in meters) to the next car ahead on track.
- **RPM**
  Engine revolutions per minute; indicates power output and gear usage.
- **Speed**
  Car speed at each telemetry sample (km/h).
- **nGear**
  Current gear (1–8 for most modern F1 cars).
- **Throttle**
  Throttle pedal percentage (0–100%), measuring engine power input.
- **Brake**
  Brake pedal pressure (0–100%); indicates deceleration intensity.
- **DRS**
  Drag Reduction System state (0 = closed, 1 = open); affects straight-line speed.
- **Distance**
  Total distance traveled around the lap at that telemetry point.
- **RelativeDistance**
  Distance relative to the leader or reference car (depends on FastF1's computation).
- **X, Y, Z**
  3D positional coordinates of the car on the track map.
- **LapNumber**
  The lap index (1–52 in your Silverstone dataset).

- **Compound**
  Encoded tire compound (Soft, Medium, Hard, Intermediate).
- **TyreLife**
  Number of laps completed on the current tire set.
- **Driver**
  Encoded driver identifier (0–8 after LabelEncoder).
- **Year**
  Encoded year of the race (0 = 2022, 1 = 2023, 2 = 2024).

The code for this integrates sampling because telemetry is not perfectly shaped so each lap has a different number of samples. Additionally grouping and aggregation are required to build the full race dataset for each driver. Due to having to merge telemetry with general race data. All data as filtered with pandas.

3.2 Preprocessing and Feature Engineering

To prepare the data for model training there were a few adjustments needed to be made. Continuous data such as speed throttle and RPM were normalized using Min Max scaling with corresponds to affine transformations. This strategy was used so the model can training with a smaller range of numbers considering that some features range from 0 to 180 while others may range from 1 to 10, so simply min maxing all of these types of features to between 0 and 1 made training quicker and more efficient. This was also done for the label and then is inversely transformed at the end to get the actual lap time predictions. Categorical features were encoded with the LabelEncoder module from scikit-learn. The tires, drivers, and year values were all handled with this which is especially important because these features are then transformed into embeddings to become learnable vectors and help the model understand the difference between the different tires, drivers, and car development from year to year. Each lap contains a different number of telemetry points and some missing values so to train the LSTM efficiently we need to pass in the same dimensions for each batch. This means that a standard sample number needs to be set for each lap and padded until the row count reaches that value. While most engineers and scientists like working with 0 as a padding number, I used -9999 because of how important 0 was to my dataset so to avoid any misunderstanding within the model I chose a value that wasn't possible.

3.3 Baseline Regression Models

Three classical models are used: Linear Regerssion, Ridge Regerssion, and Lasso Regression. Each is fit on lap level features such as mean speed throttle, break, and tire life. These models use the properties discussed in section 2 such as OLS, regularization, and comparisons between R^2 and MAE. All results are then visualized to demonstrate theory and concepts into measurable behavior.

3.4 LSTM Model Architecture and Training

To capture the temporal patterns I built a custom Pytorch Model. While I initially had tested with tensorflow, the lack of embedding layers drove me away from it. The LSTM layers process 12 telemetry features as a sequence while the embedding layers manage driver, year, and tire compound. Both are then concatenated and fed through a sequential model consisting of two dense layers which then spits out a min max scaled lap time.

For training I kept it simple and used the standard for the majority of the docs i referenced:

- MSE Loss
- Adam optimizer
- Backpropagation through Time

The model demonstrates how gradient descent learned in class can be applied to more complex models.

3.5 Testing, Evaluation, and Visualization

After training the model is evaluated on test data created by train_test_split from sci kit. Another separate test is run on Max Verstappen's 2024 race laps and then plotted against the actual lap times to compare model performance across the various driving environments such as rain, different compounds etc. The 2024 Grand Prix was chosen for Max because it has the most pitstops and variety in track conditions such as weather, tyre degradation, and combinations of tire to weather conditions(soft tyre in a wet track, wet tyre in dry track, sort of the weird crossovers in the race).

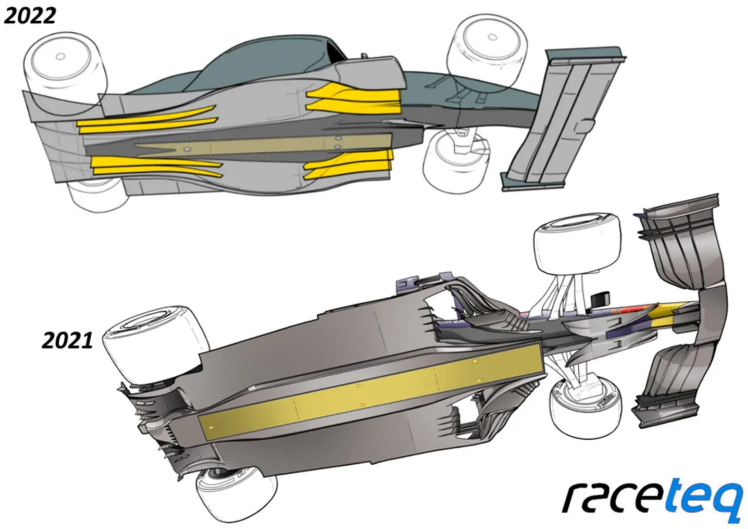
3.6 Reproducibility and Export

The trained model has been saved as a pickle object. The processed dataset will hopefully be uploaded to kaggle if size constraints permit it to be. It is a very large dataset which is difficult to replicate due to various API depreciation issues, and loading times.

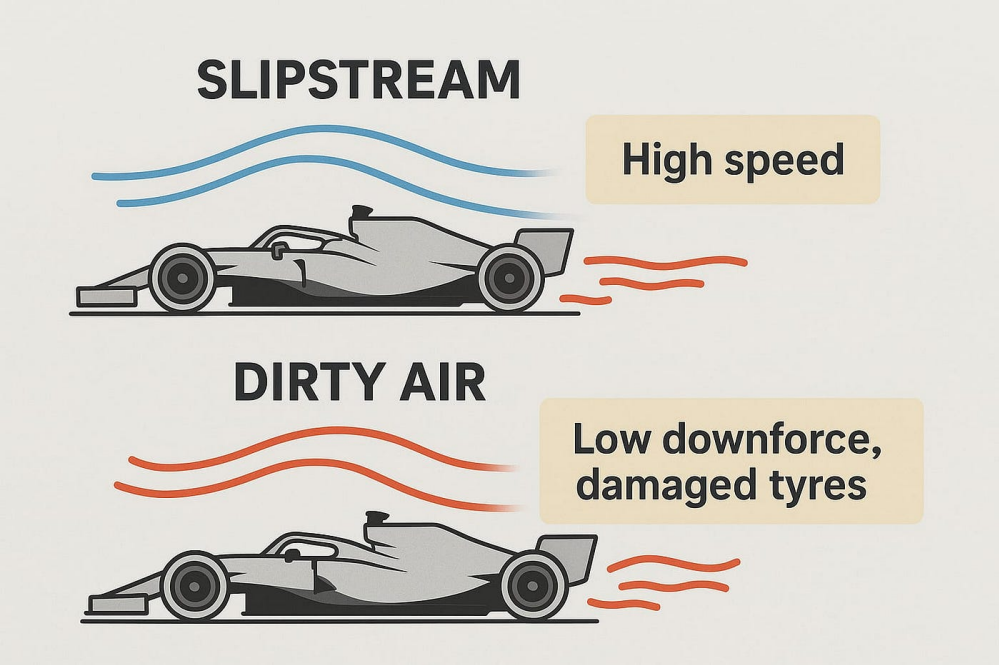
4 Data Description

The dataset used in this project as mentioned earlier was telemetry and timing information for the individual drivers as well as their overarching lap data, such as tyre compounds, tyre life, etc. The data was collected with the help of the FastF1 API. While the data collected was strong, some features and rows were cautiously removed due to interpolation rows calculated during the api request. I managed to filter the data collection source to car sensors and track sensors ignoring all inputs manually calculated by the creator of the API. These three seasons were chosen because of how F1 regulations work. Every 4 years, a full new set of regulations are created changing engine policies, car chassis requirements, and even the size of the cars. The 2022-2025 rules are known as the ground effect era meaning using

the car's underside to create low pressure, which sucks the car to the track and increases downforce meaning drivers can take turns and corners at higher speeds without having to worry about losing the rear end of the car.



This drastic change in the rules turned the tide of driver rankings. Drivers like Lewis Hamilton dominated in the 2017-2021 regulations (5 year regulation due to covid) but struggled due to the drastic change in driving style required. Lewis Hamilton has a smooth style meaning that he prefers one fluid motion when turning the steering wheel and breaking later into the corners. Drivers like Max Verstappen and Charles Leclerc have the opposite style favoring them in these regulations a lot more. Both Charles and Max prefer small quick rotations of the wheel while turning earlier into the corner, which is strong because of the increase in downforce for the cars in the ground effect era. I specifically did not include 2025 data into the pack due to the fact that some of the drivers I added into my model did not get signed to a team and others switched teams, Ocon and Perez are two prime examples. Additionally, tire wear is also affected by the rules. Because these cars run so much lower to the ground, they actually push back more dirt from the track. Driving in dirty air behind another driver hurts the tires causing them to fall off much quicker, but the counter to that point is following close behind a driver creating a slipstream giving the chasing driver a small speed advantage.



4.1 Lap-Level Dataset (Regression)

For the regression models the data is aggregated per lap for each driver. Each lap is made up of the following features:

- Mean Speed
- Mean Throttle
- Mean Brake
- Beam RPM
- Max tyrelife
- Encoded Compound
- Encoded Driver
- Encoded Year
- Lap Number

Each aggregated row includes the corresponding lap time, taken from the official timings and converted into seconds. Using the means instead of all the telemetry points keeps it dense and interpretable from a simpler model like linear regression.

4.2 Sequence Dataset (LSTM)

The LSTM model uses full telemetry sequences for each lap storing every recorded sample for a driver's 52 laps. This makes up a 3 dimensional dataset which is then split into its various embedding features, and telemetry features. This was the split:

- **Telemetry:** `['DistanceToDriverAhead', 'RPM', 'Speed', 'nGear', 'Throttle', 'Brake', 'DRS', 'Distance', 'RelativeDistance', 'X', 'Y', 'Z']`
- **Lap Context:** `['LapNumber', 'TyreLife']`
- **Tyre Embedding:** `['Compound']`
- **Driver Embedding:** `['Driver']`
- **Year Embedding:** `['Year']`

Because each lap contained a different number of telemetry samples the sequences were padded to maximum telemetry sample length and then used by the `pack_padded_sequence()` in Pytorch during model training.

4.3 Dataset Size

Across all the races and drivers the totals were:

- $\sim$`894,000` telemetry points
- 450 laps
- 27 driver-year combinations

4.4 Why Silverstone

Silverstone was chosen because it appears in all three seasons. Additionally, it is the one track that produces an amazing race every year packed with various weather conditions, strategies, and the ability to overtake. While there are amazing circuits like Suzuka in Japan and Marina Bay in Singapore, those tracks are tight, compact and don't provide much opportunity to overtake, and win on strategy. By restricting to one race, the model is able to learn meaningful tendencies without the track to track variation.

5 Expected Outcome

The expected outcome of this project is to demonstrate how classical regression and RNNs behave when applied to real life applications. The regression models are expected to show strong baseline performance missing the smaller details, while the LSTM should be able to be strong with the edge cases, and understand more hidden patterns not directly reflected in the data.

The project is expected to show:

- The LSTM captures non-linear and time based patterns that the regression models can't see.
- Regression models show good baseline metrics but fail to understand underlying F1 concepts like tire degradation and track evolution.
- Predictions should be close or match real lap times under normal racing conditions.
- Visualizations show the points above with statistical backing.

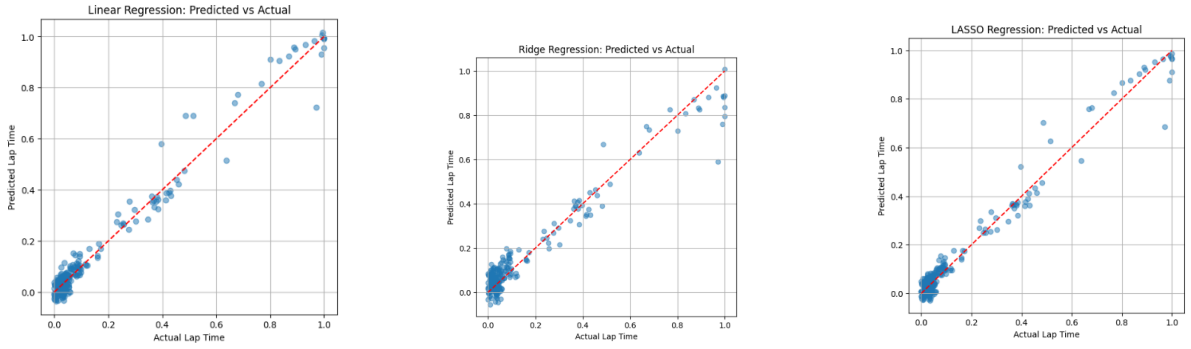
6 Results and Evaluation

6.1 Regression Baselines

Regression models were trained on the per-lap aggregated data set which means and single feature values being used to represent each lap.

Model	MAE	R^2
Linear Regression	0.0292	0.9715
Ridge Regression	0.0425	0.9442
LASSO Regression	0.0299	0.9722

Linear and Lasso performed almost the same, having high R^2 values. However, Ridge underforms most likely due to its L2 penalty bias causing it to underfit and over simplify the relationship between the telemetry and laptime. All the models were able to capture the overarching trends but struggled with laps with a pitstop such as outlap and inlap, and then safety car laps. However, I knew that for all models that it would struggle because I didn't include any pitstop or safety car data. This project was just meant to predict lap times under racing conditions.

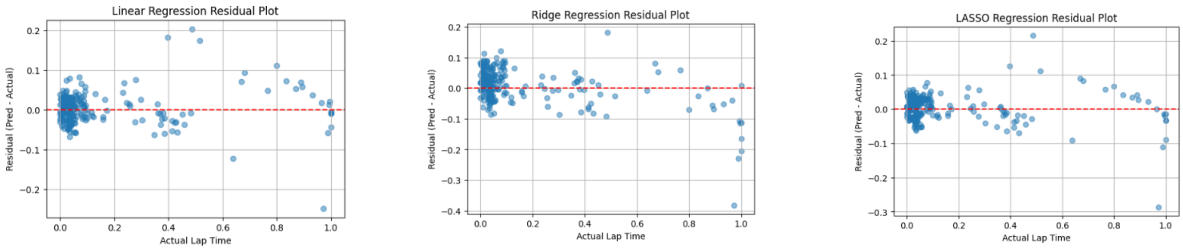


(a) Linear Regression (b) Ridge Regression (c) LASSO Regression

Figure 1: Regression Model Performance Comparisons

The Predicted versus Actual graphs show clustering near the line of best fit, increasing error at higher lap times where more variability such as rain, tyre degradation, and punctures. The outliers generally represent laps under the safety car and or pitstop laps.

6.2 Residual Analysis



(a) Linear Regression Residual (b) Ridge Regression Residual (c) LASSO Regression Residual

Figure 2: Regression Model REsidual Performance Comparisons

The residual Plots show randomness around 0 showing that there is not much systematic bias, and then some spikes for slow laps most likely due to not understanding tyre degradation, and lap anomalies discussed in the previous plots. However, this is not an underperforming model because based on the summary data we gave it did a relatively good job, especially when going in blind about tyre degradation, and dirty air struggles.

6.3 LSTM Training Results

```
Tyre: 0 3
Driver: 0 8
Year: 0 2
Epoch [1/20], Train Loss: 0.0620
Epoch [2/20], Train Loss: 0.0470
Epoch [3/20], Train Loss: 0.0439
Epoch [4/20], Train Loss: 0.0411
Epoch [5/20], Train Loss: 0.0390
Epoch [6/20], Train Loss: 0.0391
Epoch [7/20], Train Loss: 0.0363
Epoch [8/20], Train Loss: 0.0371
Epoch [9/20], Train Loss: 0.0359
Epoch [10/20], Train Loss: 0.0362
Epoch [11/20], Train Loss: 0.0347
Epoch [12/20], Train Loss: 0.0376
Epoch [13/20], Train Loss: 0.0323
Epoch [14/20], Train Loss: 0.0346
Epoch [15/20], Train Loss: 0.0325
Epoch [16/20], Train Loss: 0.0318
Epoch [17/20], Train Loss: 0.0315
Epoch [18/20], Train Loss: 0.0308
Epoch [19/20], Train Loss: 0.0304
Epoch [20/20], Train Loss: 0.0305
```

```
Test Loss: 0.0281
```

The LSTM was trained on the full telemetry sequences including coordinates on the grid, distance to the next driver, learning about breaking zones, acceleration points, and DRS activation. The training converged smoothly as the loss throughout training slowly

decreased indicating a stabilizing gradient. The final test loss is similar to the regression models but with more trends understood and a richer dataset. The LSTM model has the ability to learn about temporal patterns that aren't linear as well as understanding the difference between various drivers, tire compounds, and tire degradation.

6.4 Race Simulation: Verstappen 2024

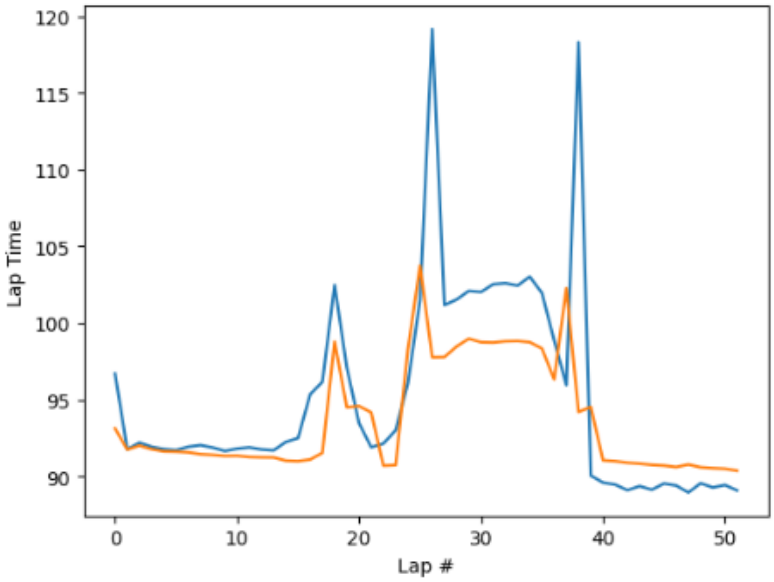


Figure 3: LSTM Verstappen Predicted vs. Actual Plot

After completing the general training and testing, it was time to see how strong it did against a full driver's race. Above the blue is the actual and orange is the predicted. Testing on the 2024 Verstappen dataset shows that the model generally performs well across standard race laps. Performance dips occur primarily during pit stop laps and wet-weather segments, where the model's predictions deviate more noticeably from the actual lap times. This is likely caused by the limited availability of wet-race and pit-related data in the training set, which can lead to underfitting in these edge conditions. Compared to the regression models the LSTM model shows smoother predictions and is more understanding of lap to lap variation especially with different racing lines and tyre degradation for each type of tyre. After manually going through and checking the pitstop laps, it did pretty well. When Max pitted onto a fresh hard tyre around Lap 38, when I subtracted the pitstop time it was very close to the actual lap time which is a good sign and indicator that it understands track evolution. Track evolution is the phenomenon where the more a racing line is driven on, the faster that part of the track becomes, because the dirt gets spread off the line, and the rubber from the tires provide more grip to the next racers. This is something I wasn't expecting to understand mainly due to the variation in tyres across different parts of a race. Sometimes a driver might use a soft at the beginning and a hard at the end and vice versa so the laptimes could have been way off.

7 Conclusion

This project showed how regression models and complex neural networks behave when applied to real Formula 1 telemetry. The regression models provided strong baseline performance using aggregated lap-level summaries of the features, capturing overall performance trends but missing deeper racing dynamics such as tire degradation and track evolution. In contrast, the LSTM model—with full telemetry laps—was able to learn sequential patterns like braking zones, acceleration points, and lap by lap degradation with dirty air and slip streaming, leading to more stable and context-aware predictions, especially in the crossover parts of the race such as wet to dry and vice versa. Overall, the work demonstrates how the optimization techniques learned in class extend naturally into the real-world, highlighting the strengths of neural networks over simple regression techniques when modeling non-linear, telemetry data like in Formula 1 racing.