Overview

From what I have learned, predicting lap times is a crucial responsibility of a strategist. On top of that, being able to understand what feature inputs are valuable in order to train the best performing model is instrumental in being confident about decision making during a race.

An important factor to consider during a race weekend is that conditions can constantly change and evolve, both within a session and from one session to another.

There are many approaches to performing machine learning regression (using an algorithm that takes feature inputs to predict a numerical value), however selecting the right method is dependent on the granularity and characteristics of the input data avaliable. For this example, I'll be using a simple Multiple Regression model taking the Tyre Life and Tyre Compound to predict the lap times during the race. I'll use the data gathered by the team during practice, and validate the model against what actually occurred during the race.

Other more involved processes such as using a robust package like XGBoost or other black box algorithms is better suited for much larger datasets. This may be useful when having more data to complement what can be extracted from the API. Data such as fuel load, track/tyre temperatures, track conditions, etc, may be useful inputs for building an XGBoost model. However, due to the limitations of the data collected, the simpler regression model will be used.

About Multiple Regression

Multiple Regression is based off linear regression, which uses evaluates the relationship between the independent and dependent variable by fitting the "best fit line" between them. This "best fit line" can be used to predict future values.

Multiple regression however, is the practice of taking multiple independent variables to predict a single outcome variable.

In [2]:
```
# load packages
import numpy as np
import pandas as pd
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

Similar to my other data cleaning procedures, I've used SQLite to clean and organize the data from all that was returned from the API.

I have also isolated data collected by the team.

In [3]:
```
# load all practice sessions from bahrain-21 ++ np times removed
df = pd.read_csv('bahrain_21_all_p.csv')
# data specific to team
df = df[df["Team"] == 'Williams']
```

In [4]:
```
# isolate columns to be used for model
df = df[["Tyre","TyreLife","LapTime"]]
df = pd.get_dummies(df, columns=["Tyre"])
df = df[["Tyre_HARD", "Tyre_MEDIUM", "Tyre_SOFT", "TyreLife","LapTime"]]
df.head()
```

Out[4]:

	Tyre_HARD	Tyre_MEDIUM	Tyre_SOFT	TyreLife	LapTime
63	0	0	1	4	95.264
75	0	0	1	3	93.959
96	1	0	0	2	96.979
98	1	0	0	3	128.245
101	1	0	0	4	96.282

In [335…
```
# check if any null values present
df.isnull().values.any()
```

Out[335… False

In [6]:
```
# split data, predictors and target variable
X = df[["Tyre_HARD", "Tyre_MEDIUM", "Tyre_SOFT", "TyreLife"]]
y = df[["LapTime"]]
```

In [7]:
```
regr = linear_model.LinearRegression()
regr.fit(X, y)
```

Out[7]: LinearRegression()

In [289…
```
# Predicted Lap time for lap 5 on hard tire is 102.65 seconds
# Predicted Lap time for lap 5 on medium tire is 102.49 seconds
# Predicted Lap time for lap 5 on soft tire is 97.73 seconds
predicted_lt_h = regr.predict([[1,0,0,5]])
predicted_lt_m = regr.predict([[0,1,0,5]])
predicted_lt_s = regr.predict([[0,0,1,5]])
```

I fit the model on the dataset, and have generated lap time predictions for each compound.

The predict function takes an array of the same dimensions as the data that was trained.

This format is the following = [tyre_hard == (0 = no, 1 = yes), tyre_medium == (0 = no, 1 = yes), tyre_soft == (0 = no, 1 = yes), tyrelife = (unit = laps)]

In [290…
```
# delta between hard and soft tire
diff_hs = predicted_lt_h - predicted_lt_s
diff_ms = predicted_lt_m - predicted_lt_s
diff_hm = predicted_lt_h - predicted_lt_m
print(diff_hs, diff_ms, diff_hm)
```

```
[[4.92071036]] [[4.75490027]] [[0.16581009]]
```

The predicted time deltas between each compound are the following:

- Hard and soft = 4.92 seconds
- Medium and soft = 4.75 seconds

- Hard and medium = 0.16 seconds

While the lap times may look inaccurate, it is largely in part due to the data collected during practice. Most notably, fuel load is unknown. Williams only ran the Hard compound for 5 laps in FP1, and none of the lap times for both drivers were under 100 seconds. However, I'm hoping this can outline my thought process on how to interpolate practice data to predict the lap times during the race. I believe it is crucial to use the evolving data from the race as those conditions may be unique to that session.

In [291…
```
# multiplier between hard and soft tire
mult_hs = predicted_lt_h/predicted_lt_s
mult_ms = predicted_lt_m/predicted_lt_s
mult_hm = predicted_lt_h/predicted_lt_m
print(mult_hs, mult_ms, mult_hm)
```

```
[[1.05039051]] [[1.04869253]] [[1.00161914]]
```

In addition to calculating the difference, I also calculated the factor of how much quicker/slower each tire is compared to another compound. For example, at lap 5 of a stint, the Hard tire is 1.05 times slower than the soft compound at that point in the tyre's lifespan. This is the same approach I'll take to apply data from the actual race in order to predict lap times later on.

Loading Race Data

In [340…
```
# load csv
# load all practice sessions from bahrain-21 ++ np times removed
race_df = pd.read_csv('bahrain_21_race_lt.csv')
# data specific to team
race = race_df[race_df["Team"] == 'Williams']
```

In [323…
```
# isolate columns to be used for model
race = race[["Tyre","TyreLife","LapTime"]]
race = pd.get_dummies(race, columns=["Tyre"])
#race = race[["Tyre_HARD", "Tyre_MEDIUM", "Tyre_SOFT", "TyreLife","LapTime"]]
race.head()
```

Out[323…

	TyreLife	LapTime	Tyre_MEDIUM	Tyre_SOFT
571	6	100.223	0	1
572	7	99.318	0	1
573	8	98.317	0	1
574	9	98.415	0	1
575	10	98.386	0	1

According to the dashboard and data, both Williams cars started on the soft compound targeting an aggressive start. Coupling this data with what was found by predicting safety car laps, a safety car was likely to come out at the start of the race. Fitting the Soft compound would allow both Russell and Latifi to fight for track position off the start, and fit the medium or hard compound if a safety car came out.

In FP2, Russell and Latifi each went on a 13 lap stint on the Soft and Medium compounds respectively. The Soft compound was ~1.0 seconds slower until the 8th lap of the tyre's lifespan. The times on the Soft became faster than those of the Medium after the 8th lap, perhaps this is due to Russell cooling the tyre temperatures? Nonetheless, the Medium compound was consistent throughout the 13 lap period.

The first 8 laps are justified to be covered on the Soft tire. But what compound should be fitted next?

There are many more factors that could be used to justify this decision, but based on the data the choice should be between the Medium and Hard compound. As stated previously, Williams only went for a few laps on the Hard in FP1, and a longer stint on the Medium in FP2. The next compound should get to the next phase of the race when either a safety car is projected or the observed max lifespan of the tyre is reached, taking into account track position and how close the pack is to one another.

In [325…
```
# split race data, predictors and target variable
X2 = race[["Tyre_MEDIUM", "Tyre_SOFT", "TyreLife"]]
y2 = race[["LapTime"]]
```

In [328…
```
regrace = linear_model.LinearRegression()
regrrace.fit(X2, y2)
```

Out[328… `LinearRegression()`

At lap 8 of the race only data from the soft compound was collected, so I interpolated that data based on conversions similar to those above. The strength of this prediction is dependent on the data avaliable.

First, I use the model to predict laptimes on laps 1-5, and lap 8 of the tyre's life. The laps chosen is arbitrary.

Then, I use the data collected from practice to generate the conversion factor to predict lap times on the medium compound.

In [347…
```
# predict lap times for lap 1-5 on the medium

# generate predictions based on practice data (soft compound)
p_r_1 = regr.predict([[0,0,1,1]])
p_r_2 = regr.predict([[0,0,1,2]])
p_r_3 = regr.predict([[0,0,1,3]])
p_r_4 = regr.predict([[0,0,1,4]])
p_r_5 = regr.predict([[0,0,1,5]])
p_r_8 = regr.predict([[0,0,1,8]])
print(p_r_1, p_r_2, p_r_3, p_r_4, p_r_5, p_r_10)
```

```
[[97.99618099]] [[97.91001972]] [[97.82385844]] [[97.73769717]] [[97.6515359]] [[97.3930
5208]]
```

In [355…
```
# compare these predictions to observed data from the race

# subset race dataframe for Williams laps -- only two laps were clear track laps
laps = [1,2,3,4,5,8]
rslt_df = race_df.loc[race_df['LapNumber'].isin(laps)]
```

```
rslt_df = rslt_df[rslt_df["Team"] == 'Williams']

rslt_df.LapTime

# compare lap 8 of prediction to lap 8 of observed
98.317 / p_r_8

# the time generated based on the lap time predictor is 1.001 seconds faster than that

# practice to race conversion unit 1.001
```

Out[355… array([[1.0094868]])

In [9]:
```
# apply this conversion unit to practice data collected for medium and hard tires
# 13 laps used as prediction target as it is the max measured during practice
p_r_13 = regr.predict([[0,1,0,25]])
hp_r_13 = regr.predict([[1,0,0,25]])

med_lap_13_pred = p_r_13*1.001
hard_lap_13_pred = hp_r_13*1.001

print(med_lap_13_pred, hard_lap_13_pred)
```

```
[[100.78389391]] [[100.94986981]]
```

Based on this preiction the Medium compound is faster by 0.1 seconds at lap 13 of the tyre's lifespan. Therefore, the decision to fit the Medium tire after the Soft stint is justified. Additionally, by fitting the Medium tyre, the criteria to fit at least two of the three compounds is fulfilled. Therefore, barring any additional unforseen safety cars, the last stop of the race should also fit the Medium.

In [ ]: