NFL Combine Project – Summary

Introduction

This project's dataset contains 337 results of the 2020 NFL Combine. The NFL Combine is an invite only event where college football players perform both physical and mental tests in front of NFL coaches, front office and scouts. The dataset used contain the following columns of interest,

- Player – Player Name
- Pos – Position
- Ht – Height in cm
- Wt – Weight in lb
- 40yd – 40yd dash times in seconds
- Vertical – Vertical jump height in inches
- Bench – Number of reps, 225lb
- Broad Jump – Jump distance in inches
- 3Cone – 3Cone drill times in seconds
- Shuttle – Shuttle drill times in seconds

https://www.kaggle.com/mrframm/nfl-2020-combine

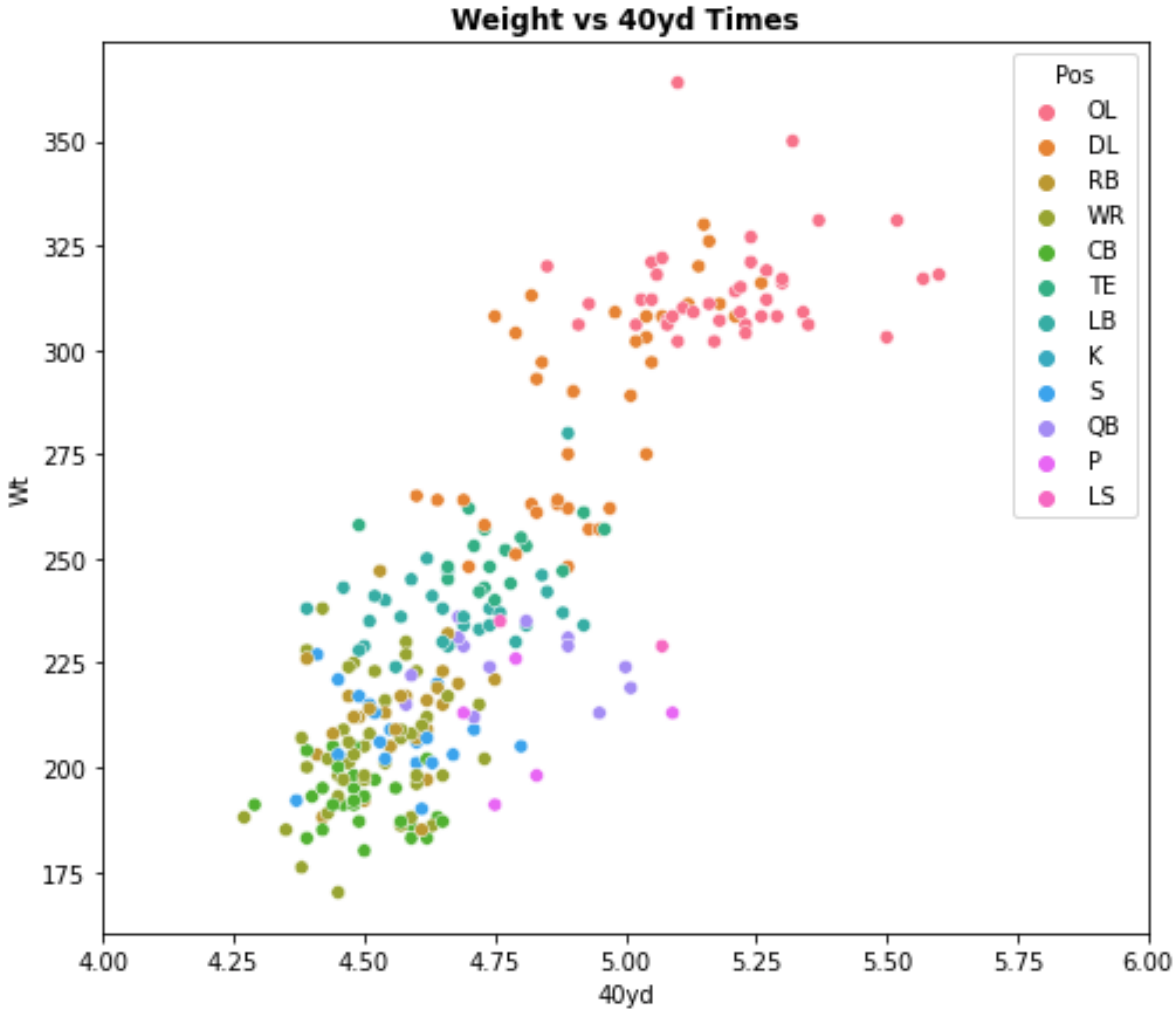
The purpose of this project is to examine the relationships between certain physical drills, position dependent relationships and various predictions.

Main.ipynb

The Jupyter Notebook (Python3) file contains the bulk of this project. We begin with importing in the packages/libraries used along with the dataset whilst extracting information from it.

Upon inspection we find that the dataset is missing data especially in the drills columns. We decide to replace the NaN values to 0. In our data-cleaning section, we also generated many boxplots to see if there were any outliers for each column. Possible conclusions for outliers found include, extraordinary performances, underwhelming performances and differences based on positions.

Further on, we generated multiple plots to visualize some of the relationships between the drills and positions. See the following scatterplot of weight versus 40 yd dash times with the legend based on positions.



From this scatterplot, we see that 40yd times will fall in the 4.2 – 5.75 second range. We can also see that as expected, the heavier players (usually the linemen) are slower than the lighter players (usually the receivers and defensive backs) which is representative of the physical expectations of the actual role on-field in regards to speed.

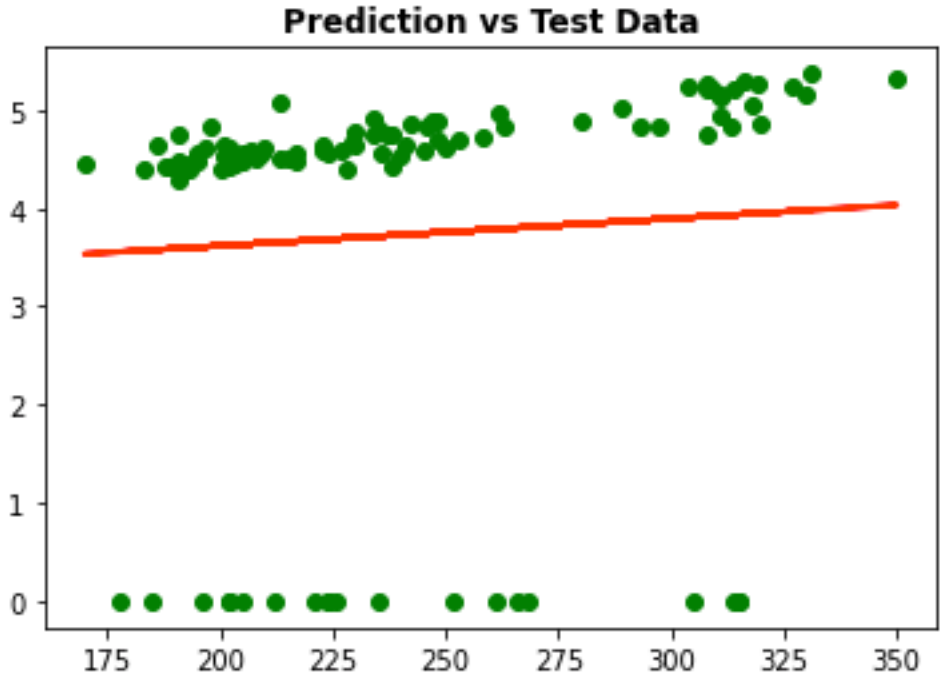
Next, we built the model. The first model used was a Simple Linear Regression using Scikit-Learn to predict 40yd times dependent on weight. After reshaping our data to $(-1, 1)$ we assigned 70% of the data to the training set and the other 30% to the test set. We then commanded the fit and predicted. Our model,

$$y = 3.06 + 0.00277x_1$$

$$40yd = 3.06 + 0.00277Wt$$

which means an additional lb in weight increases the average 40yd time by 0.000277 seconds. Also, 40yd time will be 3.06 seconds when weight is 0. We also generated the following to evaluate the performance of the model,

```
The R-Squared:  0.010880055107137188
```



From the plot, we can see that there is significant inaccuracy between the prediction and actual values. And also the R-squared value indicates that very little of the sample variation in 40yd (y) times is explained by weights (x) – extremely poor fit.

Why? The most probable reason for our model being extremely inaccurate and having a poor fit is due to the number of missing data which have been replaced to 0. This is where **main_v2.ipynb** comes into play.

We also generated multiple predictions on other examinations based on the same linear regression model but those predictions would also be inaccurate due to the reason outlined. The cases used were 7 randomly generated observations for each column, implemented through multiple functions. These random values fell between the minimum and maximums of the representative columns/variables.

We also made another model, Multiple Linear Regression using Scikit-Learn. Our model,

$$y = 2.779 + 0.158x_1 + 0.174x_2$$

$$40yd = 2.779 + 0.1583Cone + 0.174Shuttle$$

which means the 40yd time will increase by 0.158 seconds when 3Cone's time increases by 1 second and will increase by 0.174 seconds when Shuttle's time

also increases by 1 second. Similarly, the 40yd time is predicted to be 2.779 seconds when both 3Cone and Shuttle times are zero.

The same/similar code was used to generate this model and once again, there is evidence that the model was flawed due to the abundance of 0 values used and the R-squared value of 0.185221517278371.

The next section discusses the introduction of the **main_v2.ipynb** file.

Main_v2.ipynb

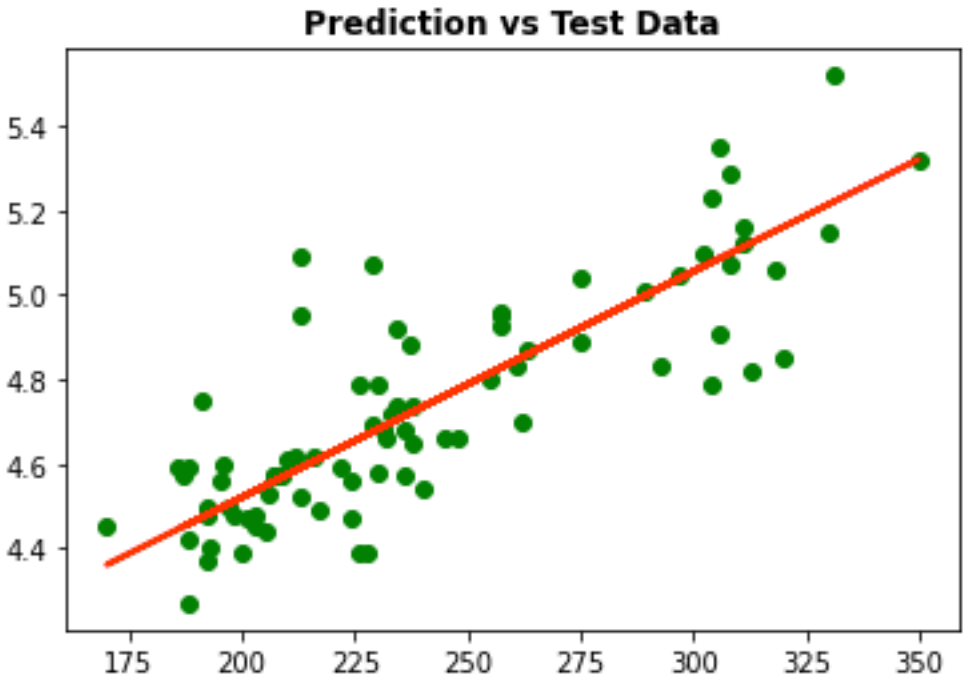
This Jupyter Notebook (Python3) file updates **main.ipynb**. The probable cause of the issues in **main.ipynb** revolves around the original NaN values. Here, in this notebook we simply drop missing data from the respective columns used.

For the first Simple Regression Model based on 40yd times based on weight, we removed all data that was missing/NaN/altered to 0 in both columns. The updated model,

$$y = 3.454 + 0.00533x_1$$

$$40yd = 3.454 + 0.00533Wt$$

and the accompanying plot and R-squared,




```
The R-Squared:  0.6777193681030442
```

Here, we see a better and more accurate prediction compared to the one in **main.ipynb** – an improved fit.

Based on the updated version of this regression model, the test-cases (7 observations per column) also generated newer and more accurate predictions.

Furthermore, for the Multiple Linear Regression the updated model and its accompanying figures and R-squared,

$$y = 0.575 + 0.471x_1 + 0.166x_2$$

$$40yd = 0.575 + 0.4713Cone + 0.166Shuttle$$

```
The R-Squared:  0.5930477636083388
```

As well, this model has increased in accuracy.

That sums up the purpose of **main_v2.ipynb,** mainly to update and improve the original models in **main.ipynb.**

Conclusion

Whilst both Jupyter Notebook files accompany each other, there can still be methods to improve this project. Some of these include,

- More data
    - We could use other years' data instead of just 2020 as we are limited to the number of athletes due to the invitation only style of the NFL Combine

- Take into account positions
    - We could conduct models specific on positions otherwise outliers based on a certain position could affect the model. *E.G. Linemen are taller and heavier but they aren't the fastest or the highest jumpers and all the other positions are in a way*

- Missing data problems
    - Missing data could be averaged or estimated off previous years or off college historical data

- Using more algorithms