PREDICTIVE ANALYSIS AND MODELLING FOOTBALL RESULTS USING MACHINE LEARNING APPROACH FOR ENGLISH PREMIER LEAGUE

DATA SOURCE

http://www.football.co.uk

ABSTRACT

The introduction of machine learning, a sub-field of artificial intelligence has created room for producing state of the art and robust predictive models. Machine learning has been used in virtually all fields in both the sciences and art; examples include recommender systems, sentiment analysis, fraud detection, image and sound recognition. Using exploratory analysis and feature, we as scientist are able to extract wealth of information from data; this information provides meaningful insights in solving real life problems. We were able to achieve an accuracy score of 62% which was 4% from our benchmark of 66%.

INTRODUCTION

Machine learning is the process of learning from data. There are two approaches, supervised and unsupervised learning. Supervised learning is when we have both inputs and output labels; we fit our model on the input and validate our results on the output. We mostly do not care how our model works on our train data but we care if our model is able to generalize well enough on new data.

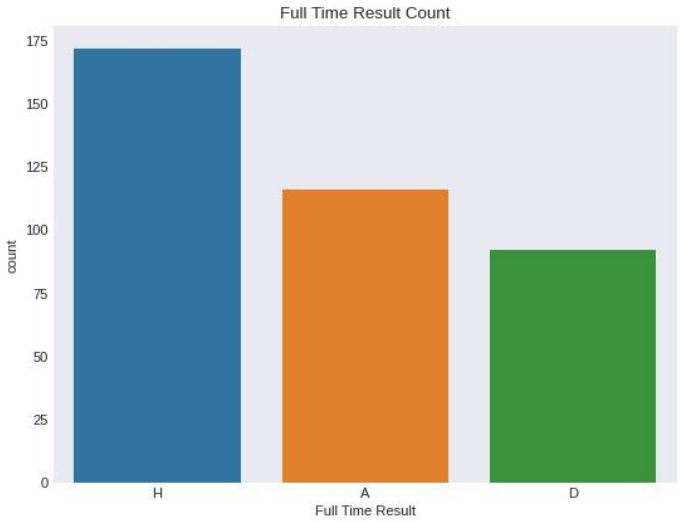
Football is a sport that is enjoy by more than 4.7 billion people all over the world. We decided to focus on the English Premier league because it is the most popular of all leagues. Our analysis focuses on the use of feature engineering as well as the application of robust, advanced machine learning techniques and algorithm. Many people and academia have tackled the problem of predicting soccer matches, this analysis is not meant to refute any analysis already on ground but to consolidate and add meaningful insights to it.

We obtained data from https://football.co.uk, we used data from 6 seasons, though we had access to other season, we chose not to include them due to missing values and some features were not present.

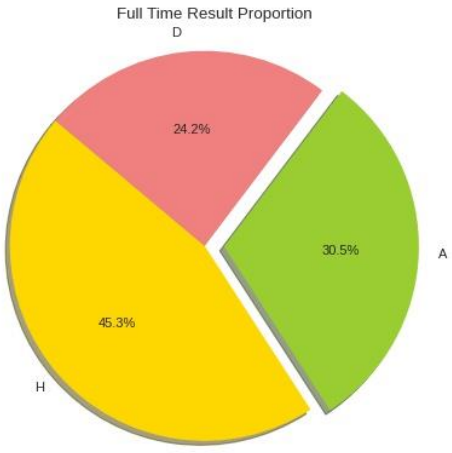
Previous models show that a quality model is a product of careful analysis and usage of the right and meaningful features. We employed this approach whilst producing our model.

ANALYSIS

For the analysis, we only make used of the 2019/2020 season, this season saw the victory of Liverpool FC. Our data had three possible results, a win, a loss or a draw. This is an example of a multi-class problem. We can see from the plot bellow that Home wins had more occurrence than the rest two, this mean home teams have more chances of winning a game compared to their counterpart.



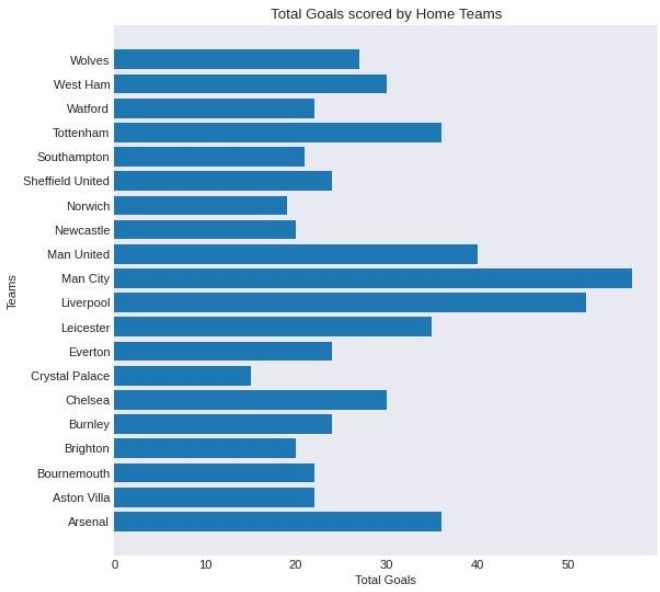
A null model would have an accuracy score of 45% because this is the probability of a home team winning a match. This accuracy would acts as our bench mark when we start validating the performance of our model.

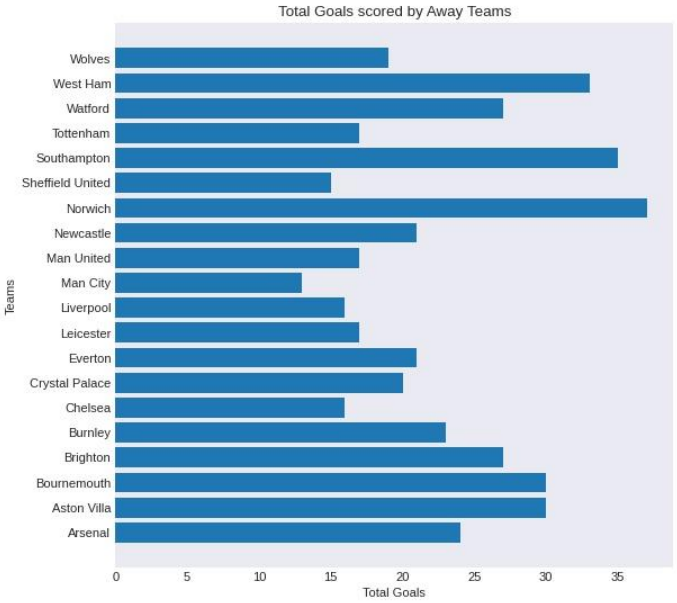


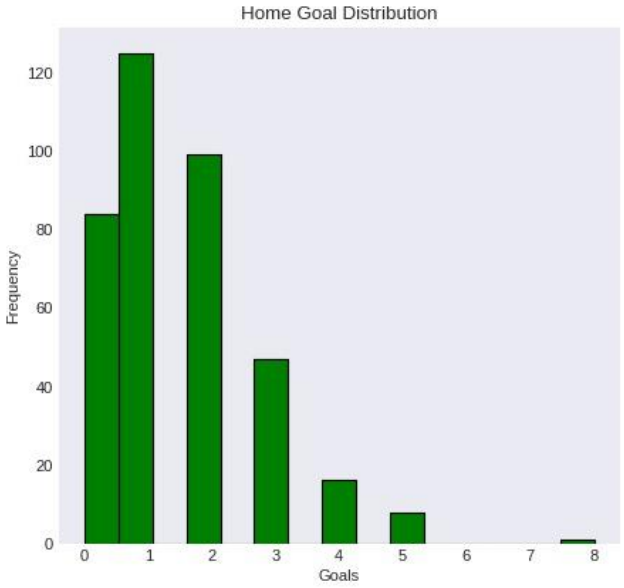
Prior to the analysis, we had intuitions that home factor would play an important role in predicting the result of a game. With this in mind, we include

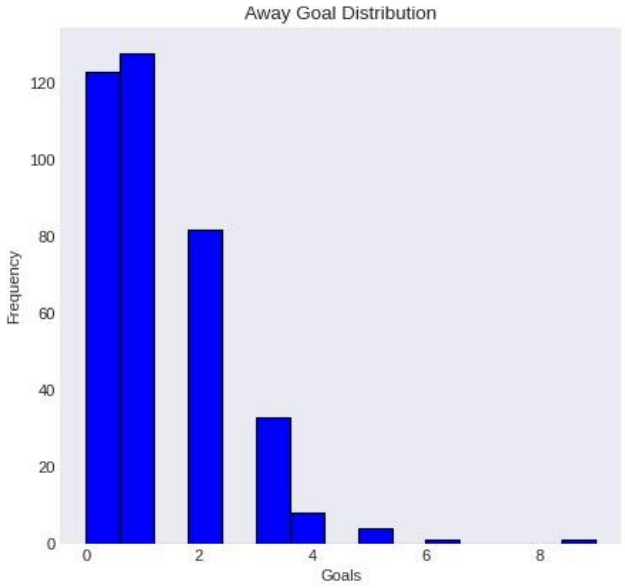
home and away features for every feature we engineered. The features were created independently so we can analyze it easily and perform inferential statistics on it.

We analyze the total goals scored by home team and away teams









Looking at the distribution of Home Goals and Away Goals, we see a similar trend. The goals left skewed, with most goals falling into the region of 0 to 3.

A team's form is a measure of the team strength in past matches, exploiting this feature lead to useful insights, we see that a team with good form has more chances of winning a game.

Form is initialized to 1.0 at the beginning of the league and updated after each match according to the result of the match in such a way that a higher value indicates a better form. When a team loses a match, a certain fraction α is multiplied to the recent form of the team and the resulting value is subtracted from the team's form

$$\xi^{\beta}_{(j-1)} = \xi^{\beta}_{(j-1)} - \alpha(\xi^{\beta}_{(j-1)})$$

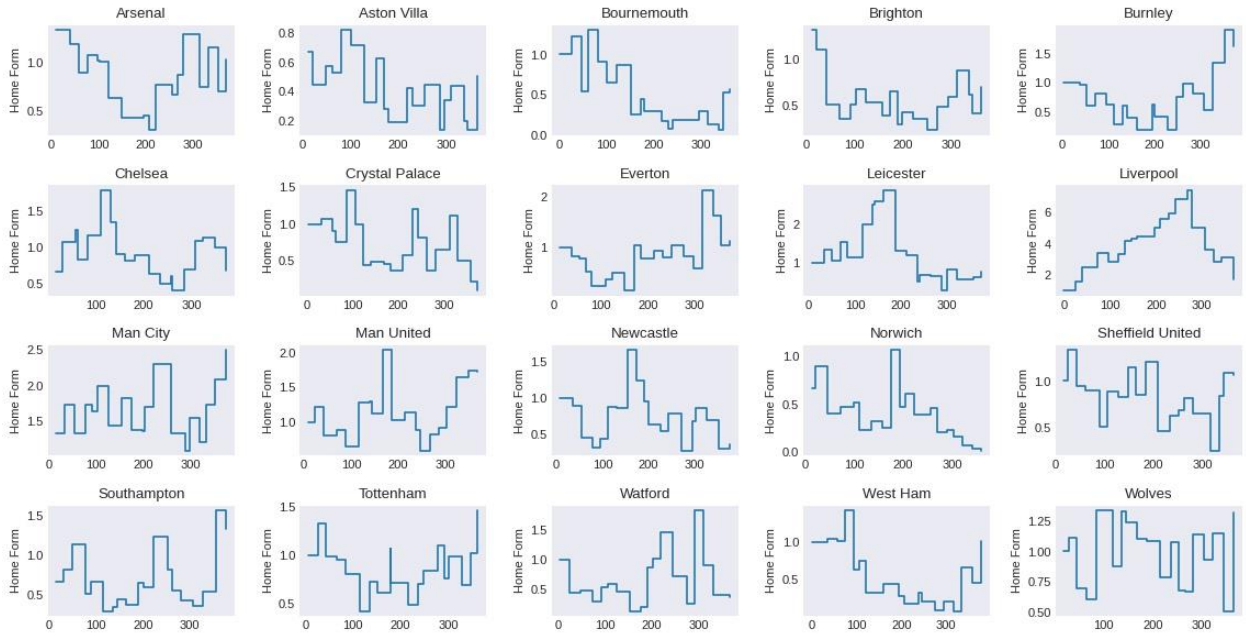
A winning team is similar but with addition

$$\xi^{\rho}_{(j-1)} = \xi^{\rho}_{(j-1)} + \alpha(\xi^{\rho}_{(j-1)})$$

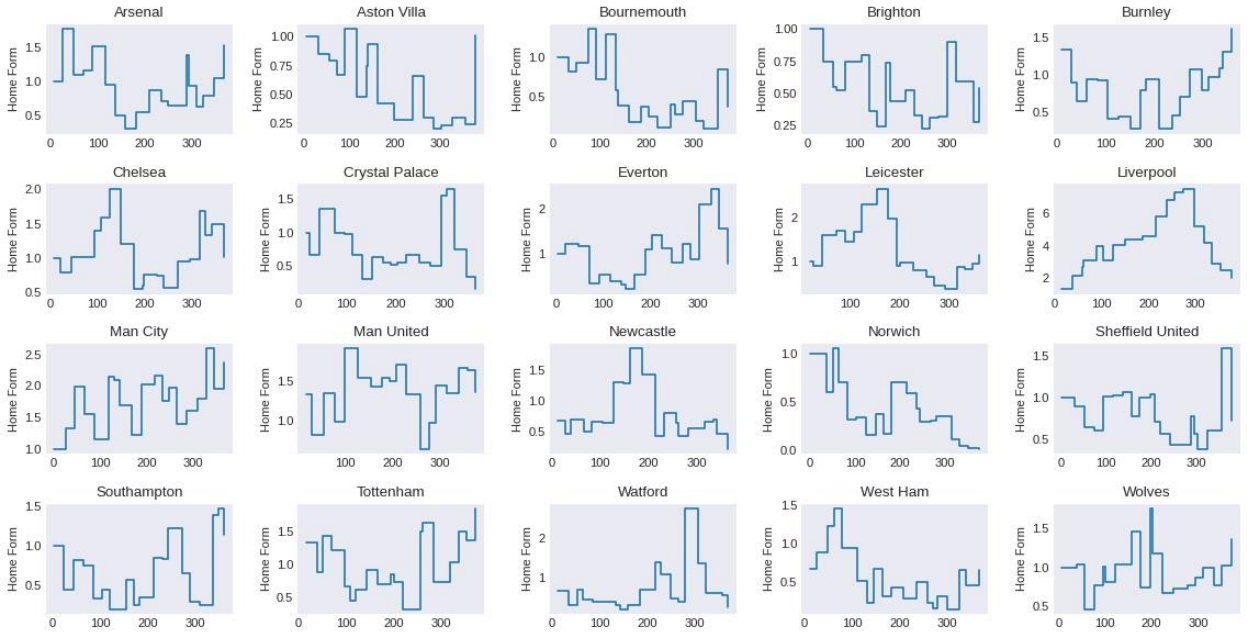
In case of a draw, the form is calculated so that a smaller team (a team with low form) steals a huge fraction from the bigger team.

For the smaller team, we calculate the form as

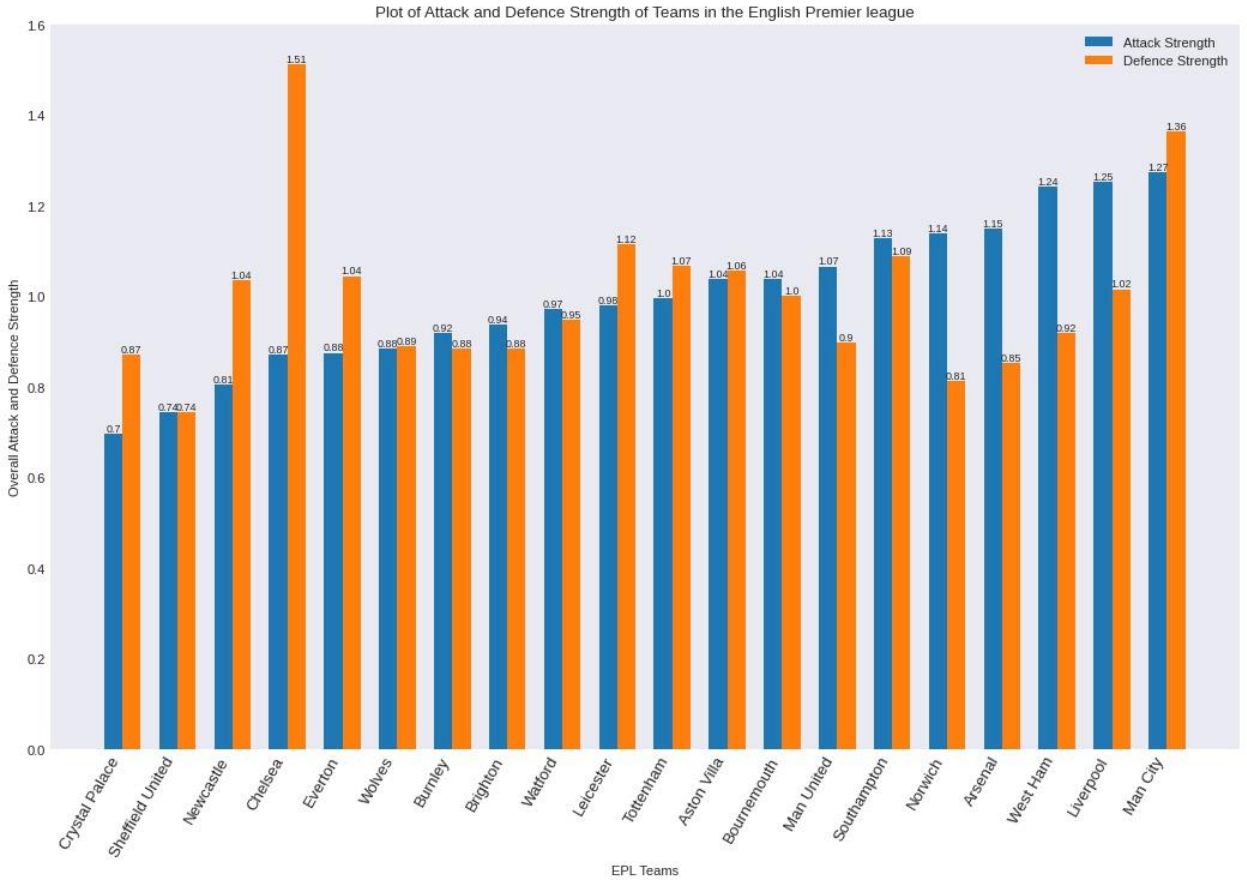
$$\xi^{\beta}_{(j-1)} = \xi^{\beta}_{(j-1)} - \alpha(\xi^{\beta}_{(j-1)} - \xi^{\rho}_{(j-1)})$$



The plot above shows the form of all teams in their home matches, Liverpool had a really impressive form which dropped during the last hour of the league. A similar occurrence is seen in their away games



We also engineer features such as the Home Overall Strength of a team and the Away Overall Strength of a team with the intuition that a team with higher strength is likely to conquer a team with a relatively lower strength



Finally, we considered the impact of derby matches on English Premier League games. We developed two hypotheses

Null hypothesis: There is no difference between derbies and non-derbies matches.

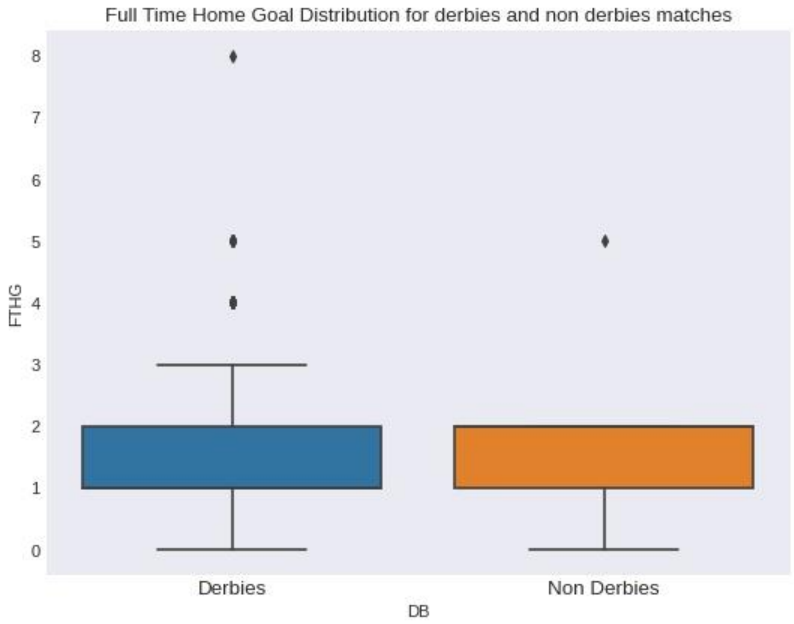
Alternative hypothesis: There is a difference between derbies and non-derbies matches.

We set the p value to 0.05 and ran a t-statistic test on the data. We obtained the following results

P value = 0.466

T value = 0.729

Since the p value was very high, we do not reject the null hypothesis and conclude that there is no difference between derbies and non-derbies matches.



FEATURE ENGINEERING

From our data, we were able to engineer the following features

Feature Code	Feature Description
HAS	Home Attack Strength
AAS	Away Attack Strength
HDS	Home Defence Strength
ADS	Away Defence Strength
HXG	Home Expected Goals
AXG	Away Expected Goals

PastFTHG	Past Full Time Home Goals for the last k matches
PastFTAG	Past Full Time Away Goals for the last k matches
PastHST	Past Full Time Home Shots on Target for the last k matches
PastAST	Past Full Time Away Shots on Target for the last k matches
PastHS	Past Full Time Home Shots for the last k matches
PastAS	Past Full Time Away Shots for the last k matches
FTR	Full Time Result

Using cross validation approach, we found an optimal k = 3. Using a high dimensional data results to building an overly complex model, we need an approach to reduce the data, we considered taking the differential of the features instead as this would reduce our dimension to half.

We standardize our data and we took the difference between the home and away features and standardize the results so as to have same scale for each feature. We ended up with the following features

Feature Code	Feature Description
AttackDiff	Attack Differential
DefenceDiff	Defence Differential
ExpGoalDiff	Expected Goals Differential
PastGoalDiff	Past Goal Differential
PastShotsOnTargetDiff	Past Shots On Target Differential
PastShotsDiff	Past Shots Differential

PREDICTIVE MODEL

Having engineered the necessary features, we proceeded to fitting our model to the train data. We chose accuracy score as a metric and we decided to use flexible models so we can improve the predictive performance

Kneighbors Classifier

This is a simple but efficient algorithm, this algorithm works by using a distance function to locate the nearest neighbor. In a classification setting, it predicts the most occurring class. This algorithm being quite a simple algorithm has some drawbacks. We fitted this model to our training data and got a cross validated accuracy of 61.32%.

Tree Classifier

This is a class of supervised machine learning where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcome. We fitted this model to our training data and got a cross validated accuracy of 58.16% **.** *We also noticed that this model was over fitting the data***.**

Random Forest Classifier

This is a class of supervised machine learning. The 'forest' it builds, is an ensemble of decision trees, usually trained with the 'bagging' method. The general idea of the bagging method is that a combination of learning models increases the overall results. We fitted this model to our training data and got a cross validated accuracy of 63.42%. *We also noticed that this model was over fitting the data*

XGB Classifier

This is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. We obtained a cross validated accuracy score of 65.00%. *We also noticed that this model was over fitting the data*

Gaussian Naïve Bayes Classifier

This is a variation of the Naïve Bayes Classifier that follows the Gaussian distribution. Naïve Bayes are a group of supervised machine learning classification algorithm based on the bayes theorem. We obtained a cross validated accuracy score of 65.26%.

Support Vector Machines

In machine learning, these are supervised methods with associated learning algorithms that analyze data for classification and regression analysis. This algorithm builds a model that assigns new examples to one category or the other using a decision boundary. We obtained a cross validated accuracy score of 61.50%.

After careful analysis of each model precision and recall, we finally chose Support Vector Machines as our model.