Football Match Outcome Prediction Using Machine Learning

ABSTRACT

Machine Learning (ML) techniques have shown great promise in classification problems. One of the important classification problems is the prediction of the results of sports matches, since sports and sports betting industries are of substantial monetary value. In this project, we aim to predict football match outcomes, particularly that of the English Premier League, one of the most popular football leagues in the world. Real-world in-match events' data of the past 10 years has been used along with EA Sports' FIFA data for the teams involved to build ML-based classifiers which predict the results of football matches.

INTRODUCTION

Football is one of the most popular sports in the world. Predicting football match results is of significant importance, since the football industry involves massive amounts of money. Data science and machine learning can be applied on various aspects of football - from betting on football matches, to understanding how a particular team can be improved. Predicting the result of a football match is a multi-class classification problem – for the home team, the match will result in either a win, a draw or a loss.

PROJECT OUTLINE

Our work was based on predicting the Full Time Result (i.e. Home team WIN, LOSE or DRAW) of an English Premier League match based on the statistics collected during the match and the past history of the two playing teams.

The data used for the project is a combination of real-world statistics and data from EA Sports' FIFA. The 'Dataset' section describes the data and data collection in detail. After combining and processing the data, we tried several models which are well known for classification tasks, for predicting the match result – Logistic Regression, Random Forests, Decision Trees, Multilayer perceptron, Support Vector Machines and a Naive Bayes classifier. We also tried K-nearest neighbors, but the results were not satisfactory. We have used accuracy as the metric to judge the performance of these different models.

RELATED WORKS

Herbinet implemented an expected goals model to generate match outcome and match score predictions. They also found out that the expected goals model performed better than traditional models such as the Dixon & Coles model [1]. Cornman et. al used a random forest approach and developed a simple betting model that used random forest output and odds data to suggest betting on the tennis player who gave maximum returns [6]. Huang et al. [7] used neural networks to predict the 2006 Football World Cup results. Their model did not classify cases of the match being drawn very well.

DATASET
Data from the past ten years (2011 – 2020) has been used for training the model.
The dataset was obtained from the website http://www.football-data.co.uk/englandm.php. It consists of statistics of each premier league match played over the past years. Data was available in csv format and consisted of the match day, time, home team, away team, goals scored (full-time and half-time) and other similar labelled in-match statistics. It also consisted of betting odds for each match, which was dropped since betting odds would have been obtained using algorithms which were not necessarily accurate enough.

To increase the amount of data which we shall be using for the model and to make the model more robust, data was scraped from https://sofifa.com/teams using web-scraping techniques (python library beautiful soup). This data consisted of team ratings and other similar labelled data given to teams by EA Sports' FIFA, a popular football game, which assigns these ratings based on real-life performances.

The columns used for feature selection were:
['HTHG','HTAG','HS','AS','HST','AST','HF','AF','HC','AC','HY','AY','HR','AR','Home_OVA','Home_ATT','Home_MID','Home_DEF','Home_TRANSFER','Home_PLAYERS','Home_HITS','Away_OVA','Away_ATT','Away_MID','Away_DEF','Away_TRANSFER','Away_PLAYERS','Away_HITS','FTR'].

The legend for the features are as follows:
HTHG = Half Time Home Team Goals
HTAG = Half Time Away Team Goals
HS = Home Team Shots Taken
AS = Away Team Shots Taken
HST = Home Team Shots on Target
AST = Away Team Shots on Target
HC = Home Team Corners
AC = Away Team Corners
HF = Home Team Fouls Committed
AF = Away Team Fouls Committed
HY = Home Team Yellow Cards
AY = Away Team Yellow Cards
HR = Home Team Red Cards
AR = Away Team Red Cards
OVA = Team overall FIFA rating
ATT = Team overall FIFA attacking rating
MID = Team overall FIFA midfield rating
DEF = Team overall FIFA defense rating
TRANSFER = FIFA Transfer Budget
PLAYERS = Number of Players in Team
HITS = Popularity of the team
FTR = Full Time Result (H=Home Win, D=Draw, A=Away Win)

The best features were obtained using Chi^2 K_Best Feature Selection - ['Home_HITS','Away_HITS','Home_TRANSFER','Away_TRANSFER','HST','HTAG','AST','HTHG','AS','HS'].

DATA PREPROCESSING

In our dataset, some features like HITS, Home_TRANSFER, Away_TRANSFER had non-numerical data which needed to be converted to numerical data in order to use them in models. This was done using functions explicitly made for the type of cleaning required. (eg.- TRANSFER had non-numerical data like €50M, which was converted to 50 by trimming the data).

The categorical data like the match result was encoded to 0,1,2 denoting home team's defeat, draw and win respectively.

The data was transformed using MinMaxScaler from sklearn's preprocessing library to normalize the data values. We tried StandardScaler as one of the scaling methods too but the results were better for MinMaxScaler.

FEATURE SELECTION

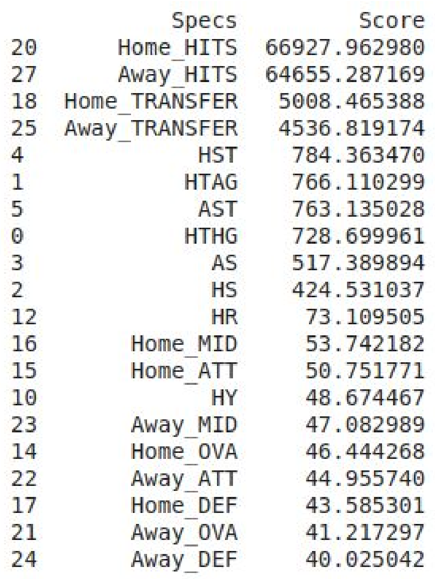
The merged dataset obtained after concatenating the scraped data with the ready csv file offered a large number of potential features that we could use to train our model, including team ratings, defence/attack/midfield ratings, as well as in-match statistics including shots on target, corners, fouls, shots, half-time goals, etc. In addition to these features, we also computed some features to capture impact of home/away conditions. We wanted a model where the labeling of the home/away team names as Manchester United or Liverpool doesn't matter. This would help us to avoid any inherent bias to/for the team and judge the result solely on the match statistics and ratings. First of all we had to remove the redundant features like referee names, team names, betting odds, etc. This led to a dataset of reduced dimensionality (with 28 features)

SelectKBest

We then applied SelectKBest from scikit-learn's feature selection library to find the most significant features from our dataset. We used the score function for SelectKBest as chi2 (chi-squared) since it usually performs well on classification tasks. The score function must return an array of scores, one for each feature, then simply retains the first k features with the highest scores.

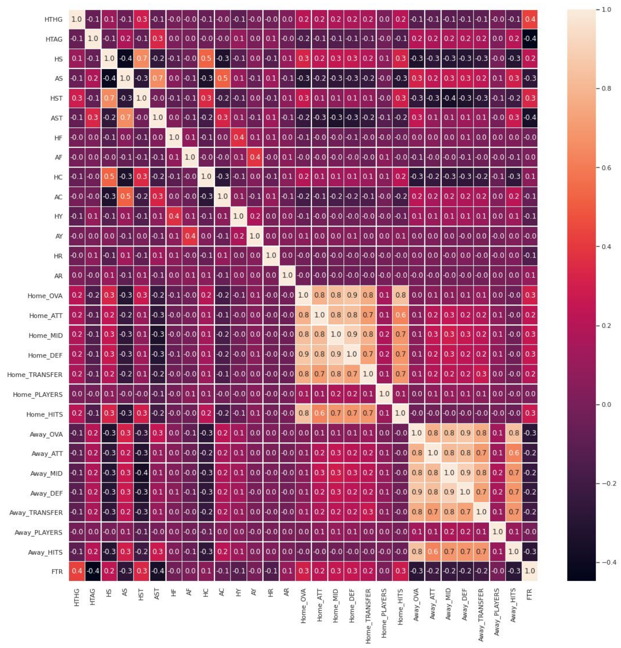
We also tried f_classif which is another commonly used score function for classification tasks. However, we got better predictions using the chi2 scores, and so we proceeded with the 10 most important features in our list of features using their chi-squared scores, since there was a significant drop in chi-squared score after the 10th feature.

We also verified the results of SelectKBest using a heatmap of correlation of features. We also tried printing the feature importance of each model separately.



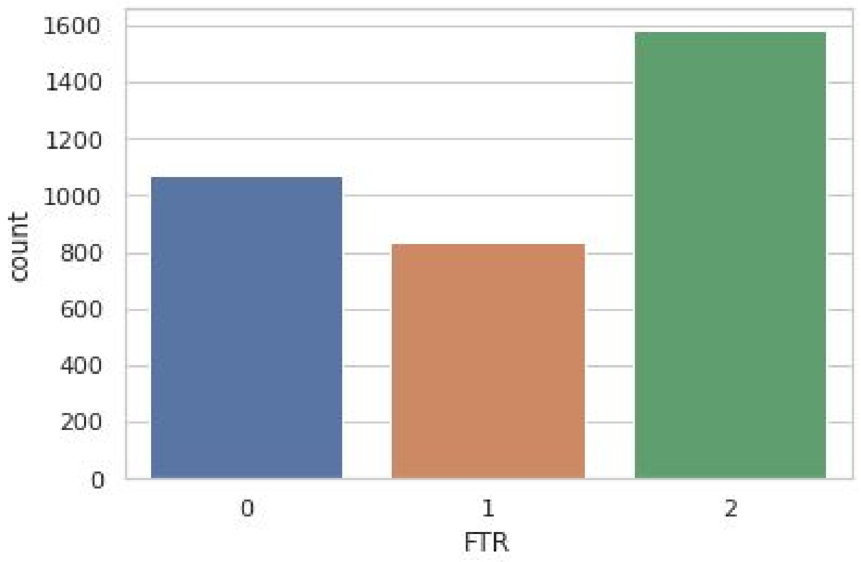
```
               Specs         Score
20         Home_HITS  66927.962980
27         Away_HITS  64655.287169
18     Home_TRANSFER   5008.465388
25     Away_TRANSFER   4536.819174
4                HST    784.363470
1               HTAG    766.110299
5                AST    763.135028
0               HTHG    728.699961
3                 AS    517.389894
2                 HS    424.531037
12                HR     73.109505
16          Home_MID     53.742182
15          Home_ATT     50.751771
10                HY     48.674467
23          Away_MID     47.082989
14          Home_OVA     46.444268
22          Away_ATT     44.955740
17          Home_DEF     43.585301
21          Away_OVA     41.217297
24          Away_DEF     40.025042
```

Chi-squared scores of SelectKBest

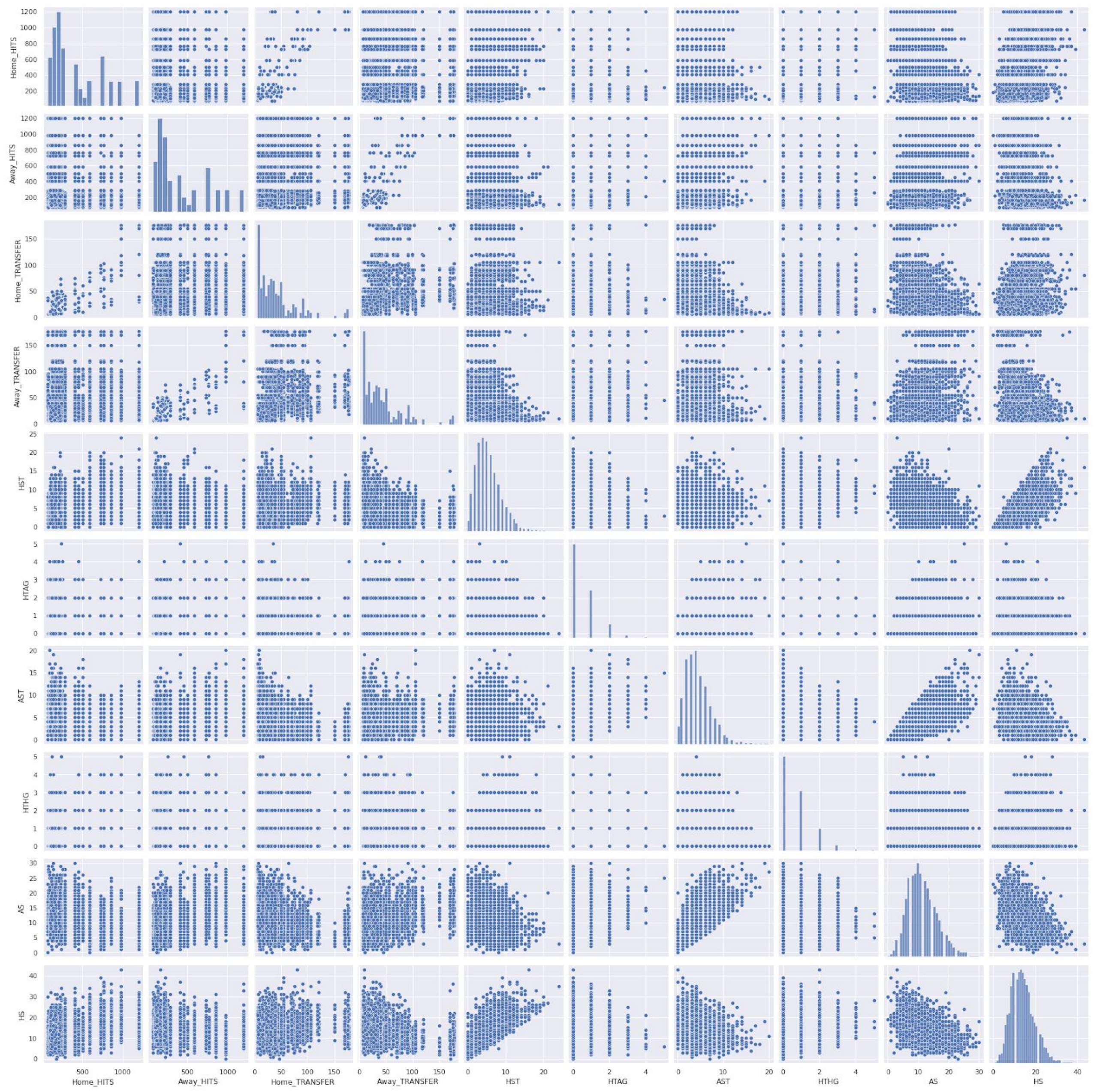


Correlation Heatmap

DATA VISUALIZATION



Count Plot



Pair Plot

METHODS

Using K_best feature selection method, we formed a reduced dataset of top ten features. Train Test Split method is used to split the data set into train and test sets in 80-20 ratio . The features are then scaled using the MinMaxScalar method. Now, the first step in the modelling process is to select which candidate models are suitable for multi-class classification. We decided to implement a number of machine learning and neural network models provided by Sklearn library.

Logistic Regression
Logistic Regression predicts the probability of occurrence of a binary event applying a logit (sigmoid) function on a linear equation of features. This squeezes the output between 0 and 1. The sigmoid function is of the form:-

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)}$$

It is assumed that labels are drawn such that from a distribution such that:-

$$P(y = 1 \mid x; \theta) = h_\theta(x)$$
$$P(y = 0 \mid x; \theta) = 1 - h_\theta(x)$$

Given a set of labeled set of training examples, we choose θ to maximize the log-likelihood:

$$\ell(\theta) = \sum_i y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))$$

We can maximize the log-likelihood by stochastic gradient ascent under which the update rule is the following (where α is the learning rate) and we run until the update is smaller than a certain threshold:

$$\theta \leftarrow \theta + \alpha(y^{(i)} - h_\theta(x^{(i)})x^{(i)}$$

For multiclass case, the one-vs-rest (OvR) scheme is used if the ‘multi_class’ option is set to ‘ovr’ with solver as LBFGS.

Gaussian Naive Bayes
Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of conditional independence between every pair of features given the value of the class variable.

$$P(y|x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n|y)}{P(x_1, \ldots, x_n)}$$

Where:

- X_i is the input feature
- Y is the target class

The likelihood of the features is assumed to be Gaussian.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

where the σ_y and μ_y parameters are estimated using Maximum Likelihood Estimation.

Support Vector Machines(SVM)
Support Vector Machines (SVMs) are Machine Learning models for both classification and regression. The training data is represented as points in space so that examples falling in different categories are divided by a hyperplane. SVCs prove an efficient classification method when the feature vector is high dimensional.

$$\min_{\gamma,w,b} \frac{||w||^2}{2} \text{ s.t. } \gamma^{(i)} \geq 1 \text{ for all } i = 1, 2, ...m$$

where the geometric margin $\gamma^{(i)}$ is equal to

$$\gamma^{(i)} = y^{(i)}((\frac{w}{||w||})^T x^{(i)} + \frac{b}{||w||})$$

The multi class classification problem is broken down into multiple binary classification cases also termed as One vs One.

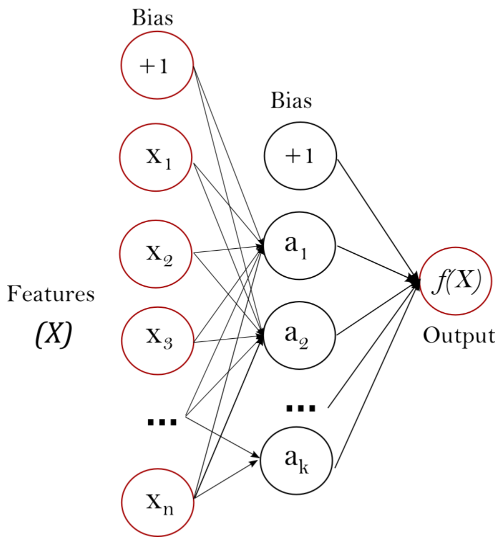
Decision Tree
Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Decision Trees can handle both binary and multi class data.

Decision trees classify the examples by sorting them down the tree from the root to some leaf/terminal node, with the leaf/terminal node providing the classification of the example.

Random Forest Classifier
A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

A tree is grown by continually “splitting” the data (at each node of the tree) according to a randomly chosen subset of the features. The split is chosen as the best split for those features (meaning it does the best at separating positive from negative examples). A random forest is made up of many trees (how many is a hyperparameter of the model) trained like this. Given a data point, the output of a random forest is the average of the outputs of each tree for that data point.

MLPClassifier
Multi-layer Perceptron (MLP) is a neural network implementation that learns a function $f(\cdot)$: $R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output.
Given a set of features $X=x_1,x_2,...,x_m$ and a target y , it can learn a nonlinear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers.This model optimizes the log-loss function using lbfgs or stochastic gradient descent.

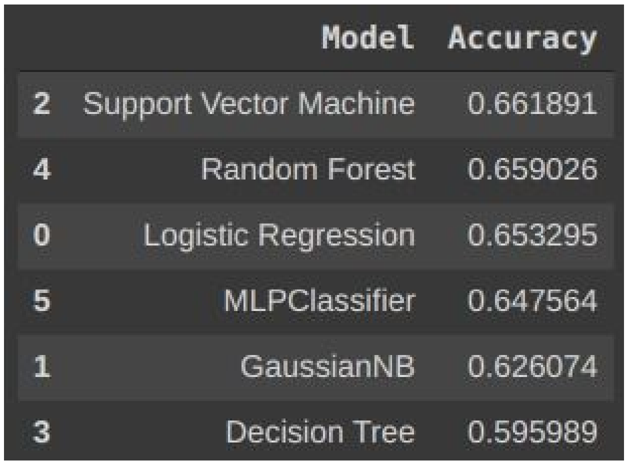


EVALUATING MODEL'S PERFORMANCE

To evaluate model performance, one would classify match results into away wins, home wins and draw and then look at the number of matches that the model has correctly identified, using classification report and confusion matrix. There is unlikely to be a great degree of imbalance in the class values for the dataset, although given the commonly observed home advantage phenomenon, one is likely to see a slight skew in favor of home wins. In this case, classification accuracy is a reasonable measure of evaluation. In cases where the data is highly imbalanced, ROC curve evaluation may be more appropriate.

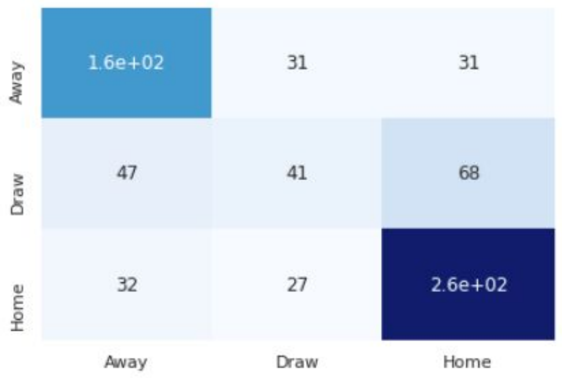
RESULTS

We performed hyper-parameter tuning to optimise the models for accuracy as the evaluation metric using GridSearchCV. The final model accuracies have been tabulated and are as follows:-

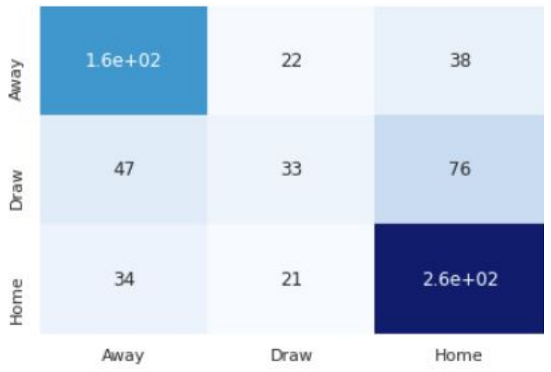


	Model	Accuracy
2	Support Vector Machine	0.661891
4	Random Forest	0.659026
0	Logistic Regression	0.653295
5	MLPClassifier	0.647564
1	GaussianNB	0.626074
3	Decision Tree	0.595989

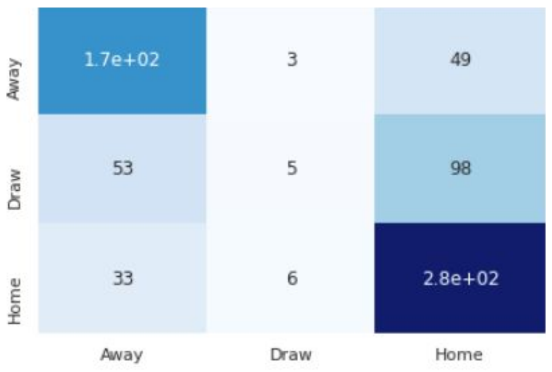
SVC
Support Vector Classifier gives an accuracy score of around ~% on hyperparameter tuning using Grid Search. The tuned parameters were: C: 1000, gamma: 0.001, kernel: rbf.



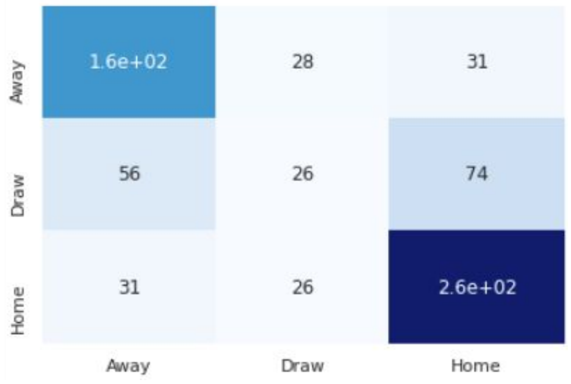
Random Forest Classifier
Random Forest Classifier gives an accuracy score of around ~ 65.9% on hyperparameter tuning using Grid Search. The tuned parameters were: max_depth=8, min_samples_leaf= 4, min_samples_split=10, n_estimators=500, criterion='entropy'.



Logistic Regression
Logistic Regression gives a good accuracy of around ~ 65.3% after hypertuning it using Grid Search. The tuned parameters were: 'C': 0.1, 'multi_class': 'multinomial', 'penalty': 'l2', 'solver': 'lbfgs'.



MLP (Multi-Layer-Perceptron)
MLP gives an accuracy score of around ~ 64.7% on hyperparameter tuning using Grid Search. The tuned parameters were: activation: tanh, hidden_layer_sizes: (100,50,100), learning_rate: adaptive, solver: adam.



Other Models
We tried KNN, Decision Tree, GaussianNB but they were not giving good results.

CONCLUSION
Our work was based on predicting the Full Time Result (i.e. Home team WIN, LOSE or DRAW) of an English Premier League match based on the statistics collected during the match and the past history of the two playing teams. The problem was a three-class classification. We thereby applied different classification models for the same. Out of these Support Vector Machines produced the best results, predicting with an accuracy of 66.2% . Followed by Random Forest Classifiers and MLP both giving an accuracy of around 65.5% on the test dataset. The features that came out to be the most important during the analysis were, Home_HITS, Away_HITS, Home_TRANSFER, Away_Transfer, HST, HTAG, AST, HTHG, AS and HS.

These features were important due to the following reasons:

- HITS represents the team's popularity amongst football fans and FIFA players. Clubs with large fan bases are usually teams which perform well and have good squads. FIFA players pick players and teams which perform well. A winning team is likely to have a better HITS rating.
- Transfer budget denotes the financial state of the club. A rich club will tend to have a better squad and hence perform better.
- Shots taken and Shots on target is an important in-match statistic since shots lead to goals if the goalkeeper fails to stop shots. A greater number of shots usually signifies a better performance.
- Half time goals give information about the momentum of each team. A team scoring a greater number of goals at half-time goes into the second half with a better momentum and hence may win.

FUTURE WORK

For our prediction model, it would be very interesting to further explore and properly validate the models that demanded more computational power than we had available to do our project. Grid Search could be applied over a more vivid dictionary of hyperparameters given more computational resources. With regards to the model we could work on predicting future matches entirely based on each of the playing teams history instead of using the statistics collected during the match. We could bring in sentiment analysis, features such as individual players and studying the trending hash-tags on twitter on match day etc to further enhance the accuracy of the model. Also drawing inference from some of the past research, we can plan on adding more features such as ‘expected goals’ instead of using actual numbers of goals scored. All of these modifications could help in greatly enhancing the performance of the models.

REFERENCES

Dataset
http://football-data.co.uk/englandm.php

https://sofifa.com/teams?type=national

Research Papers

1. https://www.imperial.ac.uk/media/imperial-college/faculty-of-engineering/computing/public/1718-ug-projects/Corentin-Herbinet-Using-Machine-Learning-techniques-to-predict-the-outcome-of-profressional-football-matches.pdf

2. https://www.kaggle.com/rambierestelle/match-outcome-predictions

3. https://www.kaggle.com/agostontorok/soccer-world-cup-2018-winner

4. https://www.kaggle.com/angps95/fifa-world-cup-2018-prediction#3.-Classification-Models-to-predict-match-results-(Win/Draw/Lose)

5. https://www.kaggle.com/saife245/football-match-prediction

6. Machine Learning for Professional Tennis Match Prediction and Betting. Andre Cornman, Grant Spellman, Daniel Wright.

7. Kou-Yuan Huang and Wen-Lung Chang. A neural network method for prediction of 2006 world cup football game. In The 2010 international joint conference on neural networks (IJCNN), pages 1–8. IEEE, 2010.