PREDICTING THE OUTCOME OF ENGLISH PREMIER LEAGUE FOOTBALL MATCHES USING MACHINE LEARNING

A PREPRINT

Group Name: `Group K`
Department of Computer Science
University College London
London, WC1E 6BT

January 25, 2021

ABSTRACT

How does one *beat the bookie*? In a game so seemingly complex, dominated by a combination of both skill and chance, predicting football match outcomes and making score predictions seems like an intractable task. On that note, the development of complex learning-based systems have allowed us to start effectively exploring this problem domain. Historically, many attempts have been made to predict the outcome of football matches by using the respective number of goals scored by each team as a measure and proxy for that team's ultimate success. In contrast, this project focuses on exploring new model design hypotheses, trained with an enhanced data set that consists of in-game match events, to effectively reflect the complex, multivariate nature of football. Furthermore, we constructively assess our models' performance using customised evaluation metrics and compare them to that of the bookmakers' models.

1 Introduction

1.1 Background

Success in team oriented sports is primarily based on three leading factors: individual skill, collective strategy, and situational chance. Furthermore, the importance of each factor — respective to each team — is constantly changing. On top of that, teams, players, and the way that contemporary football is played are never stable. As a result, you have a multivariate game that is extremely unpredictable, making any systematic approach to predict its outcome unsurprisingly difficult. On the other hand, the ability to do so is an enticing goal with extremely desirable financial returns as football is one of the most popular and profitable sports in the world. On that note, many have attempted this seemingly unfeasible task by developing both intelligent and expert-based systems to predict the outcome of football matches. Moreover, it was exhibited by Hill [1] that the outcomes of football games are not purely up to chance and, in fact, there is an aspect of skill based on the outcome. This leads us to believe that with sufficient background data, it is possible to build and train a model to predict the outcome of a game. Previous efforts in predicting English Premier League games, such as the work done by Herbinet have had a success rate of 51.1% [2]. However, we must note that even major bookmakers, such as *Bet365* and *pi-football*, do not predict the outcome of games any better, as shown by Constantinou et al. [3].

1.2 Our Approach

To estimate the relative skill levels of players in zero-sum games, an ingenious rating system was developed in 1978 by Arpad Elo — now called the *ELO rating system* [4]. The rating system, although originally created for chess, assigns each entity a numerical rating that changes based on game performance, and when two entities compete, the rating system predicts that the entity with the higher rating is expected to win more often than the lower-rated entity. In a competitive albeit team sport like football, this rating system could possibly transition well to predict the outcome of matches.

In our project, we use a similar ELO rating methodology based on *expected goals* metrics (predicted using learning-based regression analysis of a game's shots-based data and non-shot-based data) to evaluate a team's performance and, in turn, build a classification model to predict the outcome of future matches — specifically of the games played in the weekend of 16th January 2021.

2 Data Transformation & Exploration

2.1 Source

We have obtained the majority of our data from three publicly available sources: *FBRef*, *Fantasy Premier League (FPL)*, and *football-data.co.uk*. A simplified view of the structure of our datasets and their connections is presented in Figure-1.

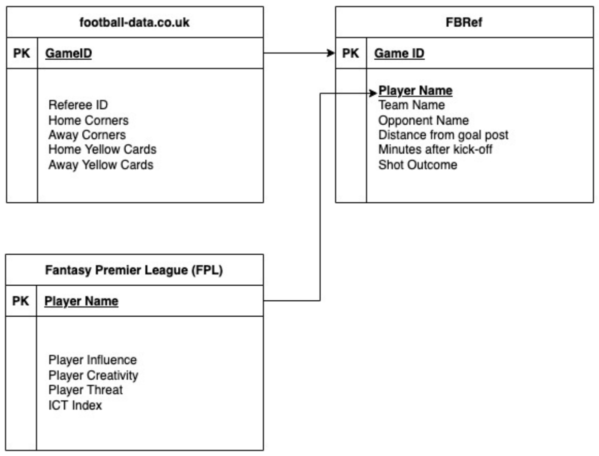


Figure 1: A simplified structural view of the datasets

The complete dataset from *FBRef* consists of the following:

- Every shot made from 2016-2021 in the English Premier League along with accompanying information including, but not limited to, player name, distance from the goal, minutes after kick-off, team of player who made the shot, and the type of shot e.g. volley, header, free kick, etc.

The complete dataset from *Fantasy Premier League (FPL)* consists of the following:

- An overview of the statistics of each season dating back to 2016 along with gameweek-specific statistics of each season dating back to 2016.
- Player-specific statistics including, but not limited to, FPL ratings for their influence on the team, their creativity on the pitch, and their threat in front of goal.

The complete dataset from *football-data.co.uk* consists of the following:

- Full time and halftime results of English Premier League matches dating back to the 1993/94 season.
- Additional match statistics since 2000/01, including shots on goal, corners, fouls, off sides, bookings, red cards and referees.
- Historical match betting odds data from up to 10 major online bookmakers dating back to the 2000/01 season.

For the purposes of our model(s), we have trimmed the datasets down to include match data from the past 5 seasons — including the ongoing 2020/21 season — and exclude all available betting odds data.

2.2 Preparation

2.2.1 Pre-processing

To ensure that our data was in a usable format for us to use when training and testing different models, a crucial step was to analyse and pre-process the data from its source.

The first pre-processing step was to build a web scraper to extract data from *FBRef*. To do this, URLs were constructed — based on the format specified by *FBRef* — to obtain all (raw) match logs in HTML format. These match logs were then parsed, sanitised, and filtered to extract relevant features. An extract of the HTML for a shot taken in a match is presented below:

```
<tr>
   <td>0</td>
   <td align="right">7</td>
   <td>Andros Townsend</td>
   <td>Crystal Palace</td>
   <td>Manchester Utd</td>
   <td>Goal</td>
   <td align="right">8.0</td>
   <td>Right Foot</td>
   <td> </td>
   <td>Jeffrey Schlupp</td>
   <td>Pass (Live)</td>
   <td>Tyrick Mitchell</td>
   <td>Pass (Live)</td>
   <td>2020-09-19 17:30:00</td>
   <td align="right">1.0</td>
   <td align="right">0.0</td>
</tr>
```

2.2.2 Standardisation

Since our data was obtained from two different sources, standardisation was necessary in order to make it internally consistent. Internal consistency, for the purposes of our project, is ensuring that each fixed data type has the same content and format.

The first standardisation procedure we used was to distinguish match data. Initially, the *FBRef* dataset distinguished each game based on the kickoff timestamp. However, further data exploration revealed that multiple games could have the same kickoff timestamp. Therefore, a decision was made to give each match a numerical ID in order to uniquely identify match data and link/relate both datasets. For example, the match played between Liverpool and Leeds with the kickoff timestamp: *2020-09-12 17:30:00* was standardised as *MatchID: 1523*. Next, we made each team name and referee's name to be consistent across both datasets and, again, assigned them a unique numerical ID. For example, Wolverhampton was standardised as *Wolves, TeamID: 40*. Similarly, referee Mike Dean's name was standardised as *M Dean, RefereeID: 20*.

3 Methodology Overview

This section describes the overall design of our model pipeline, a brief description of how each model was chosen, trained, and evaluated, along with an experiment-based analysis of the choices we made.

3.1 General Pipeline

Our model pipeline consists of multiple classification and regression models, each serving a unique purpose and playing a vital role in generating the final predictions.

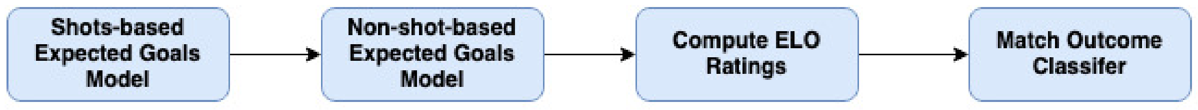

Shots-based Expected Goals Model
Non-shot-based Expected Goals Model
Compute ELO Ratings
Match Outcome Classifer

Figure 2: General pipeline of the model (Original image)

The components are as follows and are presented using the example of a single match:

- **Shots-based Expected Goals Model**
  This component's objective is to compute an expected goals metric for the home and away team that represents the sum of the probability of each shot taken in a match resulting in a goal based on the characteristics of that shot and the events leading up to it. Adjustments are made to distinguish and categorise each shot (for example, a volley, a header, a back heal, etc.) based on learning-based classification techniques.
- **Non-shot-based Expected Goals Model**
  Next, two regression models are created: one to predict the expected goals for the home team and one for the away team — both based completely on non-shot-based match data, such as the number of red cards received, the number of fouls committed, number of corners, etc., along with the expected goals metrics from the shots-based model.
- **Compute ELO Ratings**
  The objective of the next step is to assign a defensive and offensive ELO rating to both teams based on expected goals values and the actual game performance.
- **Match Outcome Classifier**
  Finally, a classifier is run on the two teams' ELO ratings to generate a prediction for a match between the two. Possible predictions include: a Home Win, an Away Win, or a Draw.

3.2 Shots-based Expected Goals Model

As the first model in the pipeline, the shot-based classification model's main objective is to predict whether a shot made by a player will result in a goal being directly scored. The classification data is derived from the *FBRef* and *Fantasy Premier League (FPL)* datasets.

Each shot data has its own type, referring to how the shot was taken. These include: *volley, header, free kick, overhead kick, back heel, and penalty kick*. All other shots are categorised as *normal shots*. We split our data based on these features based on the assumption that, from a fixed distance, each shot type's probability of being a goal would vary depending on the type of shot. A prime demonstration of this would be comparing the success rates of a penalty kick and an overhead (more commonly known as bicycle) kick. The former being highly likely to result in a goal while the latter has a low probability of resulting in a goal from a long distance.

Consequently, for each shot type mentioned above, we trained a separate classification model where the remaining input features are unchanged apart from dummy variables which categorise each shot into a sub-type.

The input shot-based data fed to the model consists of the following:

- Distance from the goal-post
- Threat level of player making the shot (taken from the *Fantasy Premier League (FPL)*)
  NB Refer to Figure-1 to view the connection between the two datasets.
- Score advantage or disadvantage of the team
- Player number advantage or disadvantage of the team
- A boolean value that is set *true* if the shot was taken in last 15 minutes of the match
- A set of dummy variables over the number of sub-types of the shot

The modelling of such specific shot data allows us to more effectively assess a team's attacking performance and potential and predict their likelihood to score goals and win the match. Alongside this, adjustments like adding a boolean value to indicate if the shot is taken in the last 15 minutes also allow us to consider the *behavioural* aspect of the game. As illustrated in the research by Kivetz et al. [5], humans (like other biological creatures) tend to expend more effort as they approach a reward. Making this adjustment therefore could be used to reflect the case when a team that is losing will become more dangerous in front of goal as the end of the match approaches.

While this model is a classifier of whether a shot results in a goal or not, the model is used to compute the sum of the probabilities over all shots made in a match. This sum represents our expected goals value for the team and we pass it on as a feature for the non-shot based expected goals model.

3.3 Non-shot-based Expected Goals Model

As displayed in Figure-1, two regression models capable of predicting an expected number of goals for the home and away teams given the different in-game statistics and the shots-based expected goal values are created in this component.

The non-shot-based data is primarily derived from the *football-data.co.uk* dataset and is integrated with the shots-based expected goals values using the standardisation procedures described in *2.2.2*.

Both regression models take as input the following non-shot-based match information and take as output the number of goals each team actually scored in that match:

- Referee ID
- Total number of corners
- Total number of fouls committed
- Total number of fouls committed against
- Total number of yellow cards received
- Total number of yellow cards received by the opponent
- Total number of red cards received
- Total number of red cards received by the opponent

Using non-shot based information allows us to consider dangerous, potential goal-scoring opportunities on the pitch that do not involve a shot being taken — one possible example of this is a player receiving a red card, limiting the squad to 10 players. This, in turn, allows us to more effectively assess a team's overall performance on the pitch and predict their likelihood to score goals and win the match.

To train our regression models, a range of popular techniques were considered. This was scaled down to a list of three models which were then tested and compared:

- SVM Regressor with Linear Kernel
- Random Forest Regressor (with a forest of 100 decision trees)
- Logistic Regression

The results we obtained by training the models on the same dataset, using the industry standard 80/20 training/test split, employing the same parameters as above, and keeping the previous component in the pipeline constant are displayed in Figure-3.

NB the process of how these models were tested is explained in *4.2.1*

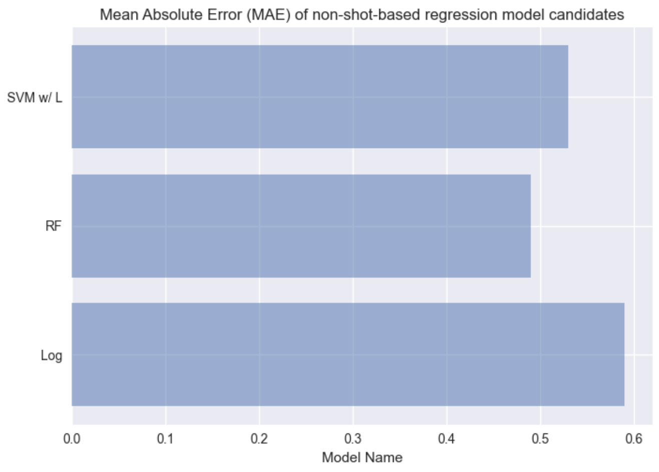


Figure 3: Mean Absolute Error (MAE) measure of 3 different non-shot-based expected goals models

Based on the results above, we can infer that the Random Forest Regressor (with a forest of 100 decision trees) produces predictions with the lowest mean absolute error (MAE) values. This was subsequently chosen to be the learning-based technique for both the home and away team models albeit with optimised hyperparameters, which were derived using Grid Search. A range of robust performance metrics, such as mean absolute error (MAE), mean squared error (MSE), and coefficient of determination (more commonly known as the R2 score), were later used to evaluate model performance in conjunction with *k*-fold cross validation in order to generate sufficiently accurate expected goals values to be used by the ELO rating classifier.

3.4 ELO rating

In order to compare any two teams in a hypothetical match, we use an ELO ratings system. As mentioned before, ELO was invented for use in chess where two players play each other and the outcome is win, loss or draw [6]. It is therefore a great candidate for modelling football matches as they have two teams and the result is either win, loss or draw.

An ELO rating system gives each team a rating and then updates their rating after a match based on who was expected to win and who actually won. A team with a higher rating is expected to win against a team with a lower rating. If a team with a higher rating beats a team with a lower rating, there will be a small rating change. If a team with a lower rating beats a team with a higher rating, the team with the lower rating will gain a significant number of points and the team with the higher rating will lose a significant number of points (often in ELO rating systems, the number of points gained by the winner is equal to the number of points lost by the loser) [2].

In a traditional ELO rating system the expected result of a match is E_H.

$$Q_H = 10^{\frac{R_H}{400}}$$

$$Q_A = 10^{\frac{R_A}{400}}$$

$$E_H = \frac{Q_H}{Q_H + Q_A}$$

Where R_H is the rating of the team and R_A is the rating of their opponent. Where E_H is 1 for a win, 0.5 for a draw and 0 for a loss.

After the match their ratings are updated using the equation.

$$R'_H = R_H + K * (S_H - E_H)$$

Where S_H is the actual result encoded as 1, 0.5 or 0. K is a parameter which controls the learning rate. This is repeated for both teams.

For our model, we could use an ELO rating system to predict the outcome of matches. However, unlike chess, the team with the most goals wins hence it would be better to predict the number of goals each team will score and then work out the expected result. To predict the number of goals both teams score, we need two ELO ratings: an offensive and a defensive rating. To find the expected number of goals a team will score, we take their offensive rating and subtract their opponent's defensive rating. We start both ratings for all teams at 0.

$$EG_H = OR_H - DR_A$$

Where EG_H is the expected goals scored by the home team, OR_H is the offensive rating of the home team and DR_A is the defensive rating of the away team.

Equivalently, we can find the the expected goals conceded by the home team by considering the expected goals scored by the away team (EG_A).

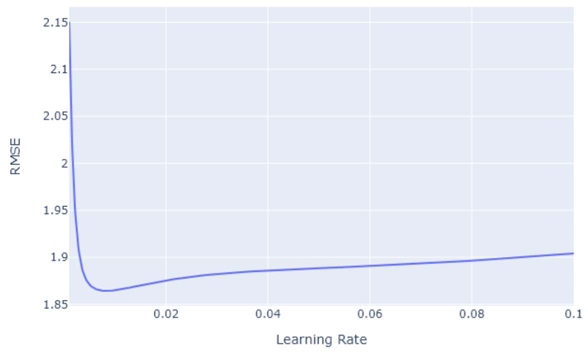
$$EG_A = OR_A - DR_H$$

After a match, we can update the teams offensive and defensive ratings using the expected goals of the ELO system and the expected goals from the non shot model.

$$OR_H = OR_H + K * (G_H - EG_H)$$

$$DR_H = DR_H + K * (EG_A - G_A)$$

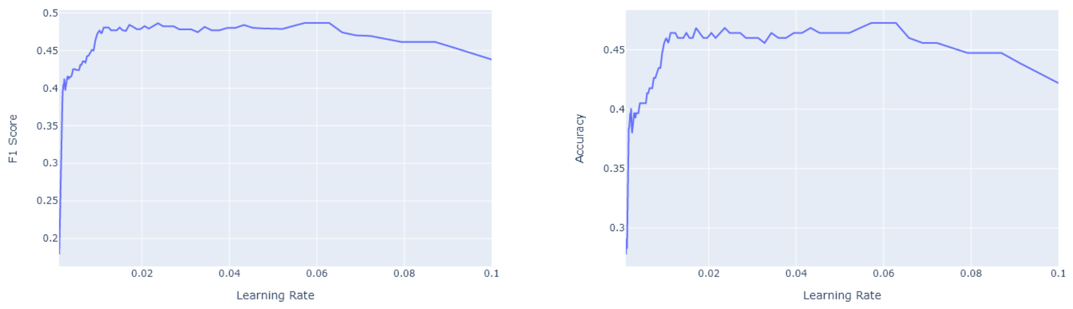
We can optimise the K parameter which is the learning rate and how much weight is given to historical matches verses recent matches.



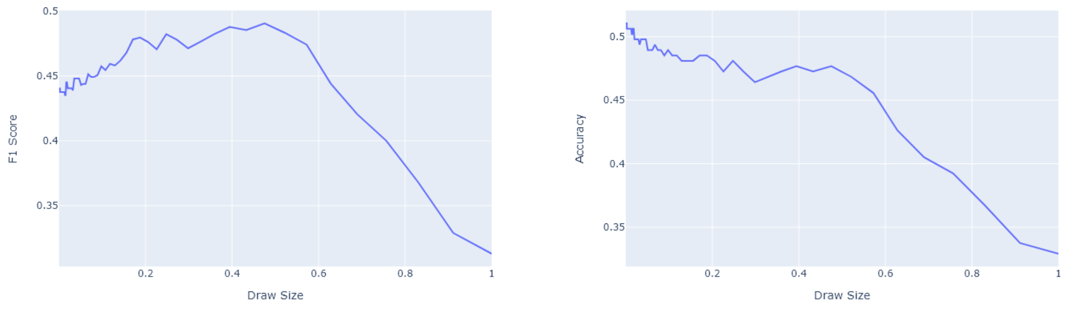
Using this model, we can make predictions about the number of goals that will be scored by both teams in a match. In order to predict the result of the match, we can imagine a simple classifier which will choose the team with the larger number of goals. We also want to predict draws so we can say that if the difference between the predicted number of goals is within a range, we can predict a draw.

$$f(EG_H, EG_A) = \begin{cases} D & |EG_H - EG_A| \leq d \\ H & EG_H > EG_A \\ A & EG_H < EG_A \end{cases}$$

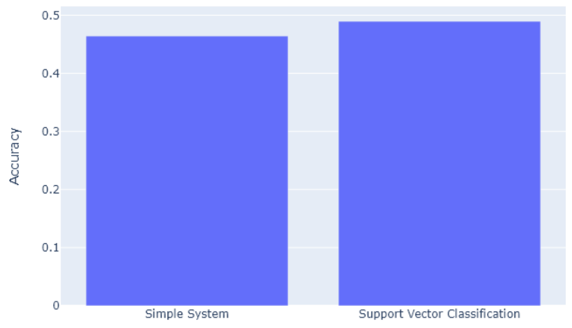
We can look at the F1 score and accuracy for predicting results with different learning rates.



We can optimise the draw size d by varying the draw size and measuring the F1 score/accuracy. This shows that the optimal learning rate is 0.05.



This shows that the optimal draw size is around 0.4. Alternatively, instead of using a simple function, we can use a support vector classifier. We give it input EG_H, predicted goals scored by home and EG_A, predicted goals scored by away and we train it to output the full time result (Home Wins, Draw, Away wins). Comparing the accuracy of the techniques shows us that the simple system has 46% and the support vector classification has 49%. The support vector classification has better accuracy than the simple piecewise function.



4 Model Training & Validation

In this section we will explore how our models' performance was evaluated and further optimised to produce more accurate predictions.

4.1 Validation techniques

Cross-validation is widely used in machine learning models to estimate and comparatively measure how well the model generalises to new data. Generally speaking, cross-validation produces a more unbiased model resulting in a less optimistic evaluation of model performance. The first step of all cross validation techniques is to shuffle the data. Next, we choose one of the following options: [7]

Train-test split: The train-test procedure is a suitable evaluation technique for a sufficiently large dataset that can be split such that both sets are adequate representations of the problem domain. On the other hand, when the dataset is too small to be effectively split, the model could fail to explore the entire problem space during training, and the insufficient testing examples could lead to overly optimistic or pessimistic results. Hence, given the fact that we limit our match data to the past 5 seasons, a decision was made to adopt a different approach. [8]

k-fold Cross Validation is a common re-sampling procedure that is generally used for a small to medium-sized dataset. In a nutshell, k-fold CV splits the dataset into k smaller sets and iteratively selects one as the test set while training on the others until all combinations of train-test folds have been exhausted. The results of a k-fold run are often summarized with the mean of the model skill scores.

Choosing k: A useful heuristic is to chose k such that each group is statistically representative of the complete dataset, as previously mentioned above. However, note that as k gets larger, the difference in size between the training set and the resampling subsets gets smaller. Additionally, as the difference decreases, the model bias does too. On that note, we chose $k = 5$ because it provided a balanced mix between the two. [7]

4.2 Testing Metrics

In order to accurately evaluate the performance of our regression and classification models, we chose to apply different evaluation metrics to reflect the nature of our task.

4.2.1 Regression Model

For our (non-shot-based) regression models, we used 3 different metrics to monitor its performance: 'Mean Squared Error', 'Mean Absolute Error' and the Coefficient of Determination'.

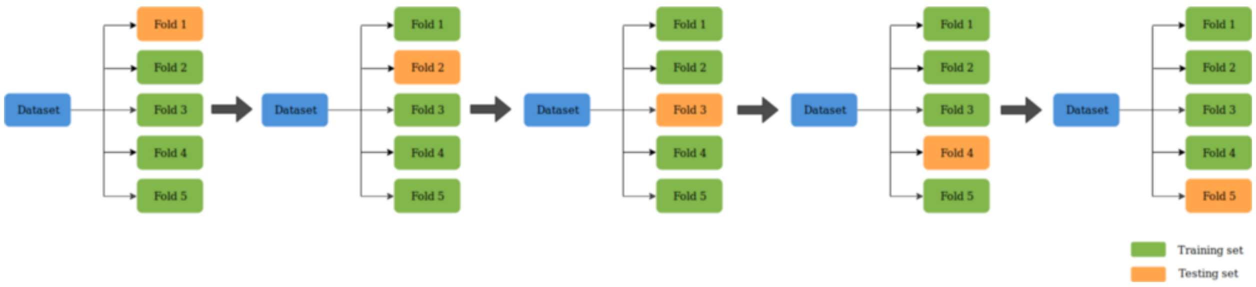


Figure 4: Visual representation of the K-Fold technique [7]

- **Mean Square Error (MSE)**

$$\textbf{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{1}$$

  Where:

  - Y_i is the actual value
  - $\hat{Y}_i$ is the predicted value

  The mean squared error measures the square of the difference between the actual and the predicted values. This value is always non-negative, which makes it an unbiased estimator of the variance from the true values.

- **Mean Absolute Error (MAE)**

$$\textbf{MAE} = \frac{\sum_{i=1}^{n}|Y_i - \hat{Y}_1|}{n} \tag{2}$$

  Where:

  - Y_i is the actual value
  - $\hat{Y}_i$ is the predicted value

  The mean absolute error (MAE) computes the sum of the absolute difference between the real and the predicted values. It is a scale-dependant measure in that it uses the same scale as the dataset, which makes it very useful when comparing different model outputs.

- **Coefficient of Determination (CoD)**

$$\mathbf{R^2} = 1 - \frac{RSS}{TSS} \tag{3}$$

  Where:

  - R^2 is the Coefficient of Determination
  - RSS is the sum of squares of residuals
  - TSS is the total sum of squares

  Finally, the coefficient of determination can be seen as a measure of the number of data points that lie on the regression line. The greater the percentage, the better fit the model is to the data. This is useful as it allows us to assess the likelihood of future events. [9].

4.2.2 Classification Model

For the classification model, we tested two different metrics: *Goal Score* and *F1*. Before we analyse the use cases and advantages of each, we must define some common measures that will be used in our calculations.

- **Accuracy**

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

Accuracy is the measure of all correctly identified examples. Accuracy is particularly useful in cases where each of the resulting classes have equal weight. Unfortunately, this is not the case for football matches, where home wins are more likely to occur than draws or away wins.

- **Precision**

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Precision is the measure of correctly identified positive examples from all the predicted positive cases and is used in domains where the cost of false positives is high. This makes it an especially viable metric for assessing the shots-based expected goals model since a significant number of false positives, i.e., undeserving shots being predicted to result in a goal could considerably bring down the overall accuracy of the model.

- **Recall**

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

Recall is the measure of the correctly identified positive cases from all the actual positive cases. Formulaically, recall draws parallels with precision and serves as a measure for scenarios where the cost of false negatives is high. Akin to precision, a large number of false negatives can result in critical disparities in the final predictions.

- **F1 Score**

$$F1 = \frac{TP + TN}{TP} \tag{7}$$

The F1 Score is a widely used measure due to its unique properties. Firstly, the F1 score provides a better measure of the incorrectly classified cases when compared to accuracy alone since it is the harmonic mean of precision and recall. Consequently, this makes the F1 score innately penalise extremities, resulting in a model that is incentivised to produce more consistent results. Additionally, the F1 score isn't negatively biased towards even class distributions — making it ideal for our project's requirements. [9]

- **False Goals Score**

$$FalseGoals = \frac{FP}{TP + FN} \tag{8}$$

The "False Goals" score is a measure of the model's success in identifying missed goals. Mathematically, False Goals represent goals that the classifier predicted as a positive example, but that never occurred in reality. Due to the skewed nature of our dataset (most shots are negative examples), the false positives (FP) have much more weight than the false negatives (FN) in producing the desired results. Hence, it is desirable that we train to minimise these occurrences.

4.3 Hyperparameter Optimisation

4.3.1 Shots-based Expected Goals Model

Because the Gradient Boosting Classifier model (xgbc) produces results similar to that of our leading (logistic) model, we perform parameter tuning using the built-in optimisation feature in *scikit-learn* to see whether we can generate more accurate predictions. This is done internally by exploring the hyperparameter space for the best cross validation score. Applied to our model, however, this technique did not deliver any improvement in results; hence, we choose to move forward with Logistic Regression as the shot-based goal classifier. [10]

4.3.2 Non-shot-based Expected Goals Model

Random Forests operate by building a large amount of decision trees during training, taking a different part of the dataset as the training set for each tree. For regression problems, like our non-shot-based expected goals prediction, the final output is the mean of the outputs of each decision tree. This results in a model with much better performance compared to a simple decision tree, due to less overfitting.

To further improve the performance of our random forest regression models, we used an exhaustive search (more commonly known as Grid Search) over a grid of hyperparamter ranges — performing cross validation with each combination of values — in order to estimate optimal hyperparameter values. The search was conducted for the following hyperparameters:

1. Number of trees in the forest — search space: $(10, 50, 100, 500, 1000)$
2. Maximum Depth of a decision tree — search space: $(3, 4, 5, 6, 7)$

The results of the search were as follows:

Home team model

1. Optimal number of decision trees in the forest: 50
2. Optimal maximum depth of a decision tree: 5

Away team model

1. Optimal number of decision trees in the forest: 1000
2. Optimal maximum depth of a decision tree: 4

5 Results

5.1 Absolute Results

5.1.1 Shots-based Expected Goals Model

For our shot-based classification model, we obtain an average Accuracy value of 0.902, an average Recall value of 0.203, an average F1 score of 0.2, and a False Goals (FG) score of 0.245. Accuracy is high across the board for all models due to the high frequency of missed shots in the dataset. On the other hand, metrics like Recall, F1, and FG appear surprisingly low. This is because they represent more fine-tuned measurements on a specialized subset of the data, each highlighting some relation between actual goals and expected goals.

5.1.2 Non-shot-based Expected Goals Model

For our non-shot-based regression models, we obtain an average MAE value of 0.450, an average MSE value of 0.562, and an average R2 score of 0.644 across both (home and away) models after k-fold cross validation testing. This means that, on average, our models' expected goals values are 0.450 goals away from reality. Although this number might seem quite large, an additional analysis has shown that there exists a high number of outliers, such as very high number of goals, something that is nearly never predicted by the model, which tends to be more conservative.

5.1.3 Match outcome (ELO Rating) classifier

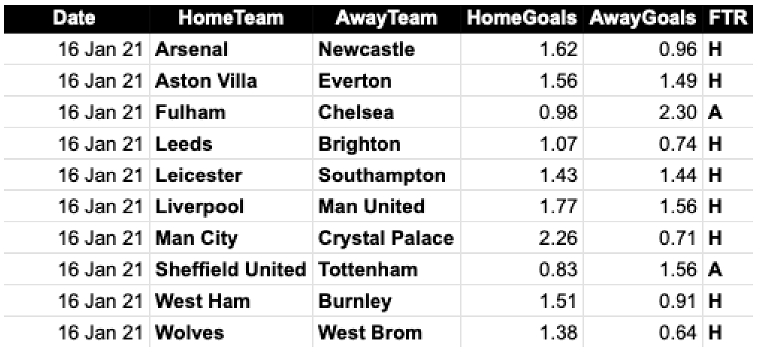
For our final (match outcome) classification model, we obtain an average accuracy of 50.5% and an average F1 score of 0.409. This means that our model is able to predict the correct outcome for just over 50% of matches. As we are looking to obtain the best possible model, there is no specific accuracy that we need to attain — a comparison with other benchmarks therefore will be the best way of effectively and objectively evaluating our model performance.

5.2 Benchmark comparison

We know that the gold standard of predicting the outcome of English Premier League matches comes from major bookmakers — their predictive accuracy being around 53%. When comparing this to our own results, we find that we fall just short of this at around 50%. This means that with a few possible adjustments and modifications to our system and overall model pipeline — possibly using more robust hyperparameter tuning techniques — we could theoretically improve our model performance and produce more accurate predictions. Alternatively, by employing stricter validation techniques, our model could exhibit relatively poorer performance and we could possibly see a dip in our overall accuracy.

6 Final Predictions on Test Set

Figure-5 illustrates our final predictions for all the games being played on the week ending 16 January 2021.



Date	HomeTeam	AwayTeam	HomeGoals	AwayGoals	FTR
16 Jan 21	Arsenal	Newcastle	1.62	0.96	H
16 Jan 21	Aston Villa	Everton	1.56	1.49	H
16 Jan 21	Fulham	Chelsea	0.98	2.30	A
16 Jan 21	Leeds	Brighton	1.07	0.74	H
16 Jan 21	Leicester	Southampton	1.43	1.44	H
16 Jan 21	Liverpool	Man United	1.77	1.56	H
16 Jan 21	Man City	Crystal Palace	2.26	0.71	H
16 Jan 21	Sheffield United	Tottenham	0.83	1.56	A
16 Jan 21	West Ham	Burnley	1.51	0.91	H
16 Jan 21	Wolves	West Brom	1.38	0.64	H

Figure 5: Final Predictions for the games being played on the week ending 16 January 2021

7 Conclusion

In conclusion, we created a model that looked at the technical data of football games played in the last 4 years and predicted the outcomes of those games with some accuracy. Using various adaptations of machine learning algorithms, such as random forest regression and logistic classification models, allowed us to manipulate and transform the data set we gathered to produce a rating system for each team. These ratings were then used to predict the outcome of a game. The resulting accuracy of our model comes close to those of modern systems designed by major bookmakers. Looking forward, the introduction of more informative features, paired alongside a more finely tuned, sophisticated model could allow us to increase the reliability of the system to produce even more accurate predictions.

References

[1] ID Hill. Association football and statistical inference. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 23(2):203–208, 1974.

[2] Corentin Herbinet. Predicting football results using machine learning techniques. *MEng thesis, Imperial College London*, 2018.

[3] A Constantinou and NE Fenton. Evaluating the predictive accuracy of association football forecasting systems. Technical report, working paper, University of London, 2010.

[4] Arpad E. Elo. *The Rating of Chessplayers, Past and Present*. Arco Pub., New York, 1978.

[5] Ran Kivetz, Oleg Urminsky, and Yuhuang Zheng. The goal-gradient hypothesis resurrected: Purchase acceleration, illusionary goal progress, and customer retention. *Journal of Marketing Research - J MARKET RES-CHICAGO*, 43:39–58, 02 2006.

[6] Prof. Mark E. Glickman. A comprehensive guide to chess ratings.

[7] Krishni. K-fold cross validation, 2018.

[8] Scikit Learn. Cross-validation: evaluating estimator performance. *línea]. Available: https://scikit-learn.org/stable/modules/cross_validation. html# crossvalidation [Último acceso: 26 Mayo 2020]*, 2017.

[9] Statistics How To. Coefficient of determination (r squared): Definition, calculation, 2018.

[10] Meng Zhao and Jinlong Li. Tuning the hyper-parameters of cma-es with tree-structured parzen estimators. In *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, pages 613–618. IEEE, 2018.