# Football Result Prediction using Data Science

**ADITYA IYENGAR**

*adityaiyengar@iitb.ac.in*

**Abstract:**

Predicting the outcome of professional football matches is a topic of great interest among data scientists and sports enthusiasts. Traditional techniques have used the number of goals scored by each team as the primary metric for this prediction. However, the number of goals scored by a team has an inherent random element in it and more often than not, the goals scored by a team is the quantity with the largest variance in the entire problem.

The first stage of this project aims to explore the application of **Deep Neural Networks (DNNs)** to classify the results of football matches as a Home Win, an Away Win or a Draw. The emphasis of the model will be on crunching historical head-to-head data and the current ELO rating of the players of the teams involved, rather than predicting the result solely based on relatively uncertain and interdependent quantities such as the number of goals scored by each team.

The second stage of the project uses the **Expected Goals (xG)** model to predict the results. Each match is modelled as a Poisson Random Process and the parameters are tuned to reflect the head-to-head data. The approach here is the opposite of the previous stage - the number of goals scored by a team is directly predicted. The result is just a consequence of these numbers.

Finally, these results are stacked up against real-world predictions offered by several betting companies.

## Contents

## 1. Introduction

### *1.1. Motivation*

Being the most followed sport in the world, prediction of the results of professional football matches has been of special interest to a lot of people, across professions. The advent of technology in sport has led to an enormous amount of in-game data available from the matches played in the last decade. This data has helped in making more and more accurate predictions.

However, as football is a sport in which each match is played for a fixed amount of time, unlike tennis (where the game goes on until a player wins), the scope for 'upsets' is significantly larger. This is what makes 100% accurate predictions over an entire season virtually impossible.

The application of data science in football is of particular significance to the betting industry. As betting on live sports increases phenomenally, it is imperative that betting odds reflect the 'true' probabilities as much as possible. Data science in football is not just of external importance, rather, it is also at the forefront of teams' strategy and preparation.

### *1.2. Objective*

The project aims to build a model to predict the result of a football match by classifying it as a Home Win, Draw or an Away Win. The model immanently assumes that there is a home team and an away team and the game isn't being played at a neutral venue. More often than not, this is indeed the case, and when it is, home advantage does indeed play a major role, hence this is a safe assumption to make.

The first step in this regard is the collection of data and pre-processing. The dataset containing historical records must at least have some basic statistics such as the home and away team name, the year and the result. Any further statistics, for example, the total number of shots in the game or the number of yellow/red cards received, is a bonus. In addition to this, we also need a player database, with teams or individual players rated as per an international standard, say ELO. This will help us better gauge a team's current level, which is certainly consequential in result prediction. Traditional score prediction algorithms only take the final result into account, so to gain an edge, we choose to account for a high number of various statistics in the model.

Having trained the models on historical data (which has been split appropriately into training and validation sets), we test the model on the ongoing 2019-20 English Premier League season. The accuracy of the model is tested, both against the actual result and a benchmark predictive method such as bookmakers' odds.

A successful outcome for the project would be a prediction model that matches the industry benchmark for prediction and preferably extends the state of the art. The model should also be easily reusable for predicting following seasons. It should also be scalable in order to predict results for other football tournaments, given data.

Following is the workflow pipeline for the execution of the project:

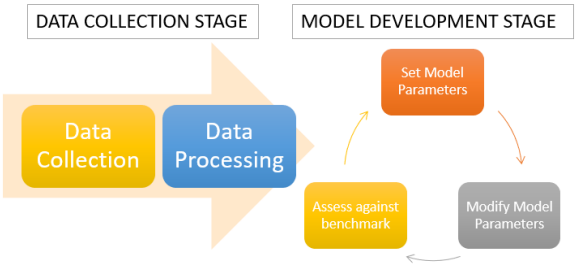


Fig. 1. Project Workflow

## 2. Background

### 2.1. Rules of the Game

We consider an extremely simplified model of the English Premier League (the top domestic football league in England) as presented below. The English Premier League is chosen due to its worldwide appeal and the easy availability of data.

- The league has 20 teams, every team plays every other team twice, once at their ground ('home') and once at the opposition's ground ('away'). Thus, each season of the league has 380 matches and each team plays 38 matches.
- Each team has 11 players - one goalkeeper, some defenders, some midfielders and some attackers. The object of the game is to put the ball into the other team's goal - a guarded target. Every goal scored by any team at any time has the same weightage.
- Every match lasts 90 minutes. The team that scores the most goals during the 90 minutes wins the game. If the teams have scored the same number of goals, the result is a draw.
- A team is awarded three points for a win and one point for a draw.
- At the end of the season, the team with the most points wins the league. The bottom three teams are relegated and replaced by new teams for the next season (This adds a new layer of uncertainty to predictions, we will see how to tackle this later).

### 2.2. Prediction Metrics

#### 2.2.1. History

Statistical football predictions have been around for a relatively long time, primarily with the purpose of outperforming the predictions of bookmakers, who use them to set odds on the outcome of matches. Publications about statistical models for football predictions started appearing in the 90s, but the first model was proposed much earlier by Moroney[1] (1956). He modelled football games using Poisson distributions and negative binomial distributions. The next attempt was provided by Reep and Benjamin[2] (1971), by modelling the series of ball-passing between players as a negative binomial distribution. In 1974, Hill[3] concurred that the outcome of a football match is predictable to some extent.

The first major breakthrough in this regard originated from the works of Maher[4] (1982). Drawing the goals scored by the home and away team from a Poisson distribution, with parameters adjusted for attacking strength, defensive strength and home advantage, he was able to predict the mean goals scored for each team. We will use a modified version this as one of our prediction models. Improvements on this model were proposed by Dixon and Coles[5] (1997). They used a correlation factor for low scores such as 0-0, 1-0, 0-1 and 1-1, which the previous model did not account for.

A major concern that remained was that team skills change during the season and across seasons. Rue and Salvesen[6] (1999) introduced a novel time-dependent rating method using time-dependent Markov Chain Monte Carlo simulations. However, a large portion of the persisting inaccuracy was due to the fact that the predicted quantity was always the number of goals scored by each team - inherently uncertain and interdependent quantities, due to the plethora of factors affecting them. The first attempt at diverting the problem statement from score prediction to result prediction was done by Forrest and Simmons[7] (2000). At the same time, Kuypers[8] (2000) developed a discrete choice regression model to predict results.

Machine learning techniques entered the fray with Goddard and Asimakopoulos[9] (2004) who presented an ordered probit regression model. It was one of the few papers at the time to consider other features apart from the match results. The model appeared to show positive

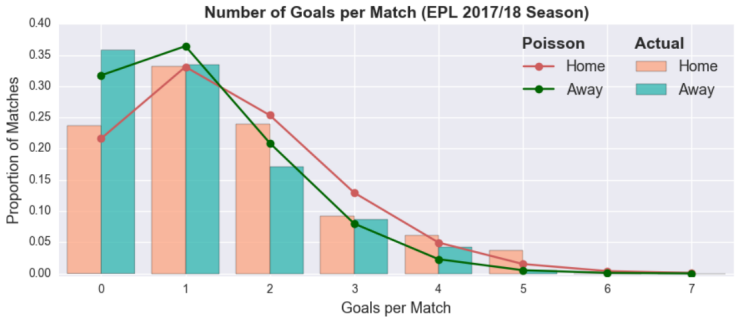


Fig. 2. Poisson Distribution as a reasonable model for prediction

betting returns. Other attempts at the time include the use of Bayesian Networks by Joseph[10] (2006) and LogitBoost by Hucaljuk et al.[11] (2011). More recently, Adam[12] and Tavakol[13] have shown that a GLM trained with gradient descent can be used to obtain decent results, provided feature selection and aggregation techniques are used.

Lastly, it might help to look beyond football, at the prediction models used in other sports with similar rules. Marek, Toupal and Sediva[14] (2014) used a modified bivariate Poisson distribution to predict the results of the Czech ice hockey league. Having had a broad overview of historical techniques, we are now aware of the benchmark set in the industry. Some potential issues have also been highlighted.

### 2.2.2. Rating System

If we are to include team strength as one of the features, we need a metric to judge which players and teams are better than others. Also since player rosters are much larger than the 11 players who play, it also boils down to the optimal choice of 11 players. This is not as straightforward, as teams could have exceedingly many good defenders but not enough good attackers. Simply selecting the 11 best players doesn't work as teams must meet a constraint of at least a certain number of players in each position. Posed thus, is a pretty complex optimisation problem, but relevant only if we are able to quantify players in the first place.

Luckily, there is an existing resource provided by EA Sports' football simulation video game series titled *FIFA*. The ratings are calculated based on a weighted average of a player's pace, shooting, passing, dribbling, defending and physical attributes among others, coupled with the player's international recognition - the final value is an integer from 0 to 100. Players are also given positional ratings, which ensure that a player holds his true rating only if he is played in position. Ratings are updated every year with changing rosters, and thus, we have an extensive database of player ratings from 2015 to 2020.

The rating data provided by FIFA solves another problem. It also includes the formation that the team used most of the time in real-life that season. Now, given the real-life formation, it is easy to pick the best players to fit the bill. For simplicity, we use the exact same starting eleven obtained from this data for the entirety of the season. This isn't an entirely accurate assumption because of injuries, form and several other constraints, but we make do with it. We use this as an important feature for the neural network part of the model.

### 2.2.3. Expected Goals (xG)

The second part of the project involves simulating a football game as a Poisson random process. For this, an important parameter is the Expected Goals (xG). While there is debate regarding the origins of the term, the first usage of the term in published literature dates back to the works of

| Statistic | CB Coefficient | Van Dijk | CB Overall |
|---|---|---|---|
| Defensive Awareness | 0.15 | 91 | 13.65 |
| Standing Tackle | 0.15 | 92 | 13.8 |
| Sliding Tackle | 0.15 | 86 | 12.9 |
| Heading Accuracy | 0.1 | 86 | 8.6 |
| Strength | 0.1 | 92 | 9.2 |
| Aggression | 0.08 | 82 | 6.56 |
| Interceptions | 0.08 | 90 | 7.2 |
| Short Passing | 0.05 | 79 | 3.95 |
| Ball Control | 0.05 | 76 | 3.8 |
| Reactions | 0.05 | 88 | 4.4 |
| Jumping | 0.04 | 90 | 3.6 |
| **TOTAL** | **1** | **952** | **87.66** |

Fig. 3. Example of the rating system: here the final rating of the player (Virgil Van Dijk) is 88[15].

Barnett and Hilditch[16] (1993). The next step in this regard was taken by Ensum et al.[17] (2004) where they analysed data from the 2002 FIFA World Cup , seeking "to investigate and quantify 12 factors that might affect the success of a shot". The probability that a shot resulted in a goal was narrowed down to a few key metrics - the attacking team's attacking strength, the defending team's defensive strength and a factor to include home advantage.

We now quantitatively define these metrics in order to be able to use them in a statistical model.

$$\text{Attack strength of a home team} = \frac{\text{Home team's average goals scored per home game}}{\text{Average home league goals scored per game}}$$

$$\text{Defence strength of a home team} = \frac{\text{Home team's average goals conceded per home game}}{\text{Average home league goals conceded per game}}$$

Similarly, we define the attack and defence strength for the away team using available data. If the attack strength of a home team is 1.235, this means that they scored 23.5% more goals at home than the theoretical average home team. Similarly, if the defensive strength of a home team is 1.094, it means that they conceded 9.4% more goals at home than the theoretical average home team.

Now how do we use this information to predict the expected goals for the team in a match? The null hypothesis, say for an away team, would be the average goals scored by an away team per game in the league. Now what factors can modify this directly? The answer is clear, the away team's attack strength and the home team's defence strength. Keeping this in mind, we define the expected goals for the home and away teams.

$$\text{xG for home team} = \text{Home team attack strength} \times \text{Away team defence strength} \times \text{Average league home goals}$$

$$\text{xG for away team} = \text{Away team attack strength} \times \text{Home team defence strength} \times \text{Average league away goals}$$

So there we have the expected values of the respective Poisson Distributions. We will use these values later in the model design stage.

## 3. Datasets

### 3.1. Data Sources

The raw datasets were collected from the following sources:

1. **'PL2020.csv'**: Schedule for the 2019-20 Premier League season from https://fixturedownload.com/results/epl-2019.
2. **players_complete.csv'**: FIFA rating database from 2015 to 2020 from Kaggle which in turn is credited to sofifa.com.
3. **H2H_PL.csv**: Head to head Premier League records from 2007 to 2019 from Kaggle which in turn is credited to optasports.com.

Some manual pre-processing was done on each of the datasets. The processed datasets can be found here.

### 3.2. Dataset Overview and Pre-processing

- The first dataset is the **Head to Head results** data. This dataset contains the results from every Premier League match played from 2007 to 2019. This will serve as our ultimate record of each team's performance in the past. The source dataset had a minor difference - the season column was specified as 2006-2007 instead of a single year. This was modified to show a single year (the latter). The result column shows H, D or A corresponding to a win for the home team, a draw or a win for the away team.

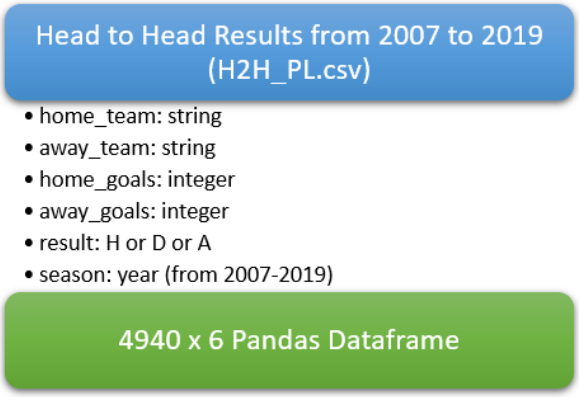


Fig. 4. Structure of the H2H_PL dataframe

- The next dataset is the Fixture List for the 2020 English Premier League season. This will serve as the test set. The structure is similar to the H2H_PL dataset with a few other columns that aren't relevant to the model.
- The last dataset is the FIFA rating database. The source dataset has a lot of features but to prevent a very sparse training matrix with incredibly many features, we have chosen only the bare essentials. Also, player potential has been accounted for in the effective overall rating. This accounts for the fact that players with a relatively low overall rating but a high potential are likelier to play than fully-developed players with the same overall.

  The player_positions column shows all the possible positions that a player plays in, whereas the team_positions column shows the position the player plays in the starting eleven.


Fig. 5. Structure of the PL_2020 dataframe

Players with *SUB* or *RES* in this column are substitutes or reserve players. These players aren't a part of the default starting eleven. We use this information to get the average rating of the default starting eleven of each team for every season. This number is of great consequence in the neural network model.


Fig. 6. Structure of the players_complete dataframe

## 4. Model Design and Implementation

### *4.1. Artificial Neural Network for Prediction*

#### 4.1.1. Preparing the Input Layer

Before we begin with the details of the neural network, we combine the relevant data from all the datasets into one dataframe. Shown below is a glimpse of this dataframe.

| | home_team | away_team | home_score | away_score | result | year | home_rating | away_rating |
|---|---|---|---|---|---|---|---|---|
| 0 | Sheffield United | Liverpool | 1 | 1 | D | 2007 | 72.0 | 84.0 |
| 1 | Arsenal | Aston Villa | 1 | 1 | D | 2007 | 85.0 | 76.0 |
| 2 | Everton | Watford | 2 | 1 | H | 2007 | 81.0 | 78.0 |
| 3 | Newcastle United | Wigan Athletic | 2 | 1 | H | 2007 | 78.0 | 72.0 |
| 4 | Portsmouth | Blackburn Rovers | 3 | 0 | H | 2007 | 72.0 | 72.0 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 4934 | Leicester City | Chelsea | 0 | 0 | D | 2019 | 80.0 | 85.0 |
| 4935 | Liverpool | Wolverhampton Wanderers | 2 | 0 | H | 2019 | 84.0 | 78.0 |
| 4936 | Manchester United | Cardiff City | 0 | 2 | A | 2019 | 85.0 | 74.0 |
| 4937 | Southampton | Huddersfield Town | 1 | 1 | D | 2019 | 78.0 | 77.0 |
| 4938 | Tottenham Hotspur | Everton | 2 | 2 | D | 2019 | 85.0 | 80.0 |

4939 rows × 8 columns

Fig. 7. Preview of the combined dataset

There are still a few steps that must be taken even before this dataset can be split into training and validation sets. For starters, the team names are strings and unintelligible to the neural network. A simple alternative to this is to assign the teams numbers in alphabetical order. Another possibility could be to arrange the teams in order of their rating, but that makes, in a way, the rating feature redundant. We stick with the former encoding strategy.

Another problem that was tackled before we could actually combine the datasets was that our player rating data was from 2015 to 2019. On the other hand, the head to head data dates back to 2007. The fairly obvious problem here is that we have no data for the ratings columns for a vast majority of the dataset. For teams that have actual entries post 2015, we use their maximum rating from 2015-2020. We could have easily used the average, but the maximum does the job just as well. That solves the problem for most of the teams, but there are still some N/A entries. These are teams that played for a season or more from 2007 to 2014 and didn't play ever again. Here, knowing the distribution of ratings (minimum of 70, maximum of 87), we award them a default rating of 72. A relatively lower value is preferred. This is because if they aren't playing in the top flight anymore, it is likely that they had finished in the bottom three and been relegated, hence were one of the weakest teams around.

The result column appears as single letters (H, D or A). This needs to be encoded into 0, 1 and 2, again, because letters are unintelligible to the neural network. Having done this, we are left with only one task. The year and the rating column are normalized using min-max scaling. Having done this, here is the dataset, almost ready to be split into training and validation sets.

The result column is our target variable and we separate it out from the feature matrix. The home score and away score columns are also removed, since we aren't interested in predicting the scores here, although this can be done separately as a regression problem. Having done this, the set is split into training and validation sets. 5% of the data is randomly selected for validation.

#### 4.1.2. Model Specifics

The feed-forward neural network architecture is implemented as shown below. The input layer has 32 neurons. There are two hidden layers, with 16 and 8 neurons respectively. The output layer has three neurons, as required for classification. The activation function in all but the output layer is **ReLU**. Apt for the task, the output layer is activated by the **softmax** function.

| | home_team | away_team | home_rating | away_rating | year | result | home_score | away_score |
|---|---|---|---|---|---|---|---|---|
| **0** | 28 | 19 | 0.117647 | 0.823529 | 0.0714286 | 1 | 1 | 1 |
| **1** | 0 | 1 | 0.882353 | 0.352941 | 0.0714286 | 1 | 1 | 1 |
| **2** | 14 | 34 | 0.647059 | 0.470588 | 0.0714286 | 2 | 2 | 1 |
| **3** | 23 | 37 | 0.470588 | 0.117647 | 0.0714286 | 2 | 2 | 1 |
| **4** | 25 | 3 | 0.117647 | 0.117647 | 0.0714286 | 2 | 3 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **4934** | 18 | 11 | 0.588235 | 0.882353 | 0.928571 | 1 | 0 | 0 |
| **4935** | 19 | 38 | 0.823529 | 0.470588 | 0.928571 | 2 | 2 | 0 |
| **4936** | 21 | 9 | 0.882353 | 0.235294 | 0.928571 | 0 | 0 | 2 |
| **4937** | 29 | 16 | 0.470588 | 0.411765 | 0.928571 | 1 | 1 | 1 |
| **4938** | 33 | 14 | 0.882353 | 0.588235 | 0.928571 | 1 | 2 | 2 |

4939 rows × 8 columns

Fig. 8. Preview of the modified dataset

Having tried different depths, neurons per layer and activation functions, the following combination appears to be the most optimal, in terms of training accuracy.

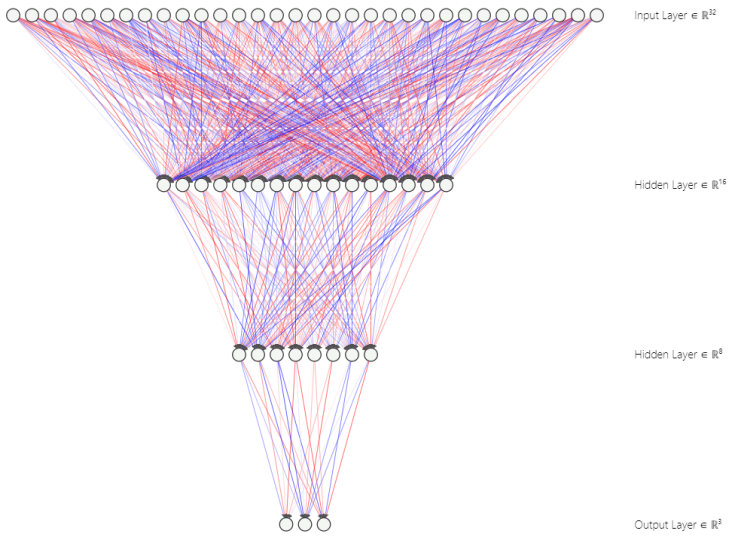


Fig. 9. Neural Network Architecture[18]

Having decided upon the overall architecture of the neural network, we now tune some of the hyper-parameters, namely the number of epochs and the optimizer used. The loss function used for the model is fixed to be **cross-entropy loss**. We try out several optimizers and compare their training accuracies, each for a set number of epochs. Keeping overfitting in mind, we need to look out for the right number of epochs and the optimizer that gives a reasonable training accuracy.

Keeping these results in mind, we choose the **Adagrad** optimizer and 30 epochs. So there we have our final model. The training accuracy over the number of epochs is shown below. This model can now be directly used on the fixtures for the 2019-2020 season (test set).

| ACCURACY | | | | | |
|---|---|---|---|---|---|
| | Number of Epochs | | | | TEST |
| **Optimizer** | **10** | **20** | **30** | **40** | 30 |
| *Adam* | 69.96% | 70.69% | 70.87% | 70.79% | 51.43% |
| *Adagrad* | 68.61% | 68.79% | 70.80% | 70.74% | 53.80% |
| *Adadelta* | 69.95% | 70.65% | 70.70% | 70.61% | 50.03% |
| *Adamax* | 70.20% | 70.21% | 70.67% | 70.86% | 53.27% |
| *SGD* | 64.81% | 66.78% | 67.54% | 67.56% | 46.25% |
| *RMSProp* | 69.88% | 69.97% | 70.30% | 70.64% | 51.32% |

Fig. 10. Accuracies across optimizers and number of epochs

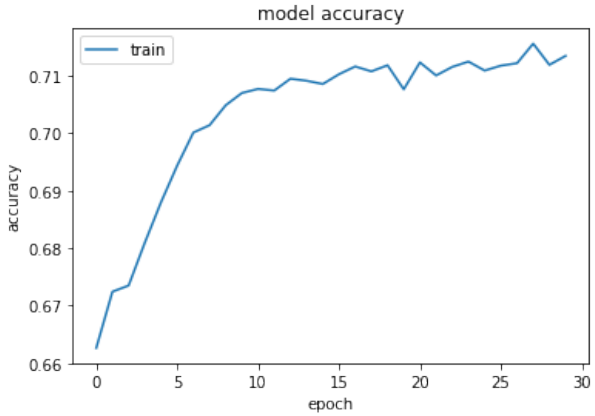


Fig. 11. Training accuracy vs. Number of Epochs

### *4.2. Poisson Distribution Model*

Recall that we defined the expected goals for the home and away team previously. Modelling the football match as a Poisson distribution, if $\lambda$ is the expectation value of the number of the goals scored and $k$ is the number of goals scored, then:

$$\text{P(k goals in the match)} = \frac{\lambda^k e^{-\lambda}}{k!}$$

This allows us to distribute the 100% probability across multiple goal outcomes. It is observed that the probability that a team scores 6 goals or more, in the extreme case, is not more than 2%, hence we can restrict the possible outcomes from 0 to 5 goals.

Now we have the respective probabilities that the home and away teams score 0 to 5 goals each. At this stage, without any further information, we need to make the crucial assumption that the number of goals scored by each team are conditionally independent. This isn't the case, to be precise, but given the simplicity of the model, we make the assumption and see where that takes us.

Given that the number of goals on either side are independent of each other, the probability of a particular scoreline is simply the product of the individual Poisson probabilities. The most likely result is simply the scoreline corresponding to the element in the probability matrix with the maximum value.

Let us take an example to illustrate this. Consider a game between Manchester City and Bournemouth, played at the home of the former. The probability matrix returned for this match is shown below.

| POISSON PROBABILITIES | | Manchester City | | | | | |
|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 |
| Bournemouth | 0 | 2.60323 | 7.36128 | 10.408 | 9.81038 | 6.93533 | 3.92228 |
| | 1 | 2.13638 | 6.04114 | 8.54144 | 8.05103 | 5.69158 | 3.21888 |
| | 2 | 0.876624 | 2.47888 | 3.50483 | 3.3036 | 2.33544 | 1.32081 |
| | 3 | 0.239805 | 0.678109 | 0.958762 | 0.903715 | 0.638871 | 0.361314 |
| | 4 | 0.0491998 | 0.139125 | 0.196706 | 0.185412 | 0.131075 | 0.0741294 |
| | 5 | 0.0080753 | 0.022835 | 0.0322858 | 0.0304321 | 0.0215137 | 0.0121671 |

Fig. 12. Probability Matrix for a match between Manchester City (home) and Bournemouth (away). Green scorelines are the likeliest where as the red ones are relatively unlikely.

Thus, the likeliest scoreline is **2-0 to Manchester City**, closely followed by 3-0 and 2-1 with the same winner. Now, let us consider a slightly more interesting case. Here is the probability matrix for a match between Manchester City and Liverpool at the home stadium of the former.

| POISSON PROBABILITIES | | Manchester City | | | | | |
|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 |
| Liverpool | 0 | 5.47517 | 9.78691 | 8.74709 | 5.21183 | 2.32905 | 0.832638 |
| | 1 | 6.11817 | 10.9363 | 9.77434 | 5.8239 | 2.60257 | 0.930422 |
| | 2 | 3.41834 | 6.11031 | 5.46111 | 3.25393 | 1.4541 | 0.519845 |
| | 3 | 1.27326 | 2.27597 | 2.03415 | 1.21202 | 0.541624 | 0.193632 |
| | 4 | 0.355698 | 0.635813 | 0.56826 | 0.33859 | 0.151308 | 0.0540928 |
| | 5 | 0.0794942 | 0.142096 | 0.126999 | 0.0756707 | 0.0338155 | 0.0120891 |

Fig. 13. Probability Matrix for a match between Manchester City (home) and Liverpool (away). Colour scheme as before.

Upon inspection, it is clear that the likeliest scoreline is a 1-1 draw. Now, we may be tempted to say that, when Manchester City play Liverpool at the former's home, the likeliest result is a draw between the two sides. **However, this is wrong!** The actual probability of a result encompasses all the possible scorelines giving that result. A 1-1 draw is not representative of all other scorelines that result in a draw. Thus, the probability that Manchester City wins the game is the sum of all elements in the upper triangular part of the matrix (sans the diagonals). Similarly, the probability that Liverpool wins the game is the sum of all elements in the lower triangular part of the matrix. Finally, the probability of a draw is the sum of the diagonal elements. We get these values as 52.06%, 24.69% and 23.25% respectively. Clearly a draw cannot be the likeliest result! In fact, it is, by a narrow margin, the unlikeliest! So, we need to be really careful when we translate the probability matrix for a particular match into 'verbal predictions'. The likeliest scoreline and the likeliest result may point in entirely opposite directions at times, causing the results to be misleading. In conclusion, the element with the maximum value in the probability matrix corresponds to the most likely scoreline. On the other hand, the sum of all possible elements with the same result constitutes the probability of a particular result. These two, in general, need not point to the same result.

Since our overall objective involves result prediction, we use the sum of the respective elements of the matrix as the output probability. We can now use this to predict the results of the 2019-20 season.

## 5. Testing the Model

### 5.1. Artificial Neural Network Model

Since the season was ongoing at the time this was written, we consider all games played until 22 June 2020. This amounts to exactly 300 games out of a possible 380, a pretty reasonable fraction of the season. The model gives us an output of three probabilities, one each for a home win, away win and a draw. Given these probabilities, we choose the result with the maximum probability as the 'predicted result'.

Over 300 games, the model predicts the exact result with **50%** accuracy. Excluding draws, **66%** of the results are correctly classified.

### 5.2. Poisson Distribution Model

We use the same metric here - the result with the highest probability is chosen to be the 'predicted result'. The model predicts the exact result with **52%** accuracy. Excluding draws, the accuracy moves up to **70%**.

Except for games that ended up as draws, 55 out of the 77 matches wrongly classified by the neural network were also wrongly classified by the Poisson model, perhaps suggesting that the results may have been unexpected.

### 5.3. Comparing with a Popular Benchmark

We use the average closing odds from five of the UK's most popular bookmakers - Bet365, Bwin, William Hill, 1xBet and Paddy Power. Using these odds, we are able to predict **53%** results correctly. Excluding draws, the accuracy is at **71%**.

Comparing the performance of the bookmakers' model to our classification model, we see that both our models have marginally worse classification accuracy. Quite impressively, of the 55 results that both our models classified wrongly, 47 have been misclassified by the bookmakers as well. Since the odds we have considered are closing odds, they are updated until the game actually starts. This gives the bookmakers a significant advantage of judging form, injuries, roster changes and other factors in order to modify the odds. Bookmakers also use advanced models along with larger and more exhaustive datasets trained on many seasons. On the other hand, our model gives probabilities of all 380 matches before the league actually starts. To almost match that accuracy with a relatively simpler model is a sign that the model performs quite well in general.

### 5.4. Comparing with a Naive Benchmark

The question may arise as to what the model has actually accomplished compared to a naive prediction algorithm. For starters, we can randomly pick results, considering each result to have an equal probability. The accuracy of a useful algorithm is lower bounded by this value (**33.33%**). All our models do significantly better than this.

A slightly smarter way of predicting the result would be to always go for a home win, considering home advantage to be a major factor. This gives us a prediction accuracy of **44%**. Again, reasonably worse than our model predictions.

## 6. Conclusions

### 6.1. Summary

The main objective of designing a machine learning based algorithm to predict football match results has been accomplished with a reasonable amount of success. We trained a neural network on data comprising team ratings and head to head data to output match result predictions. We also used a Poisson distribution to model a football match and used it to predict match results.

Having optimized the parameters in both these models, we compared our predictions to benchmark methods in order to better understand our models' predictive performance. Quite crucially, we discovered that our models achieve a similar performance to bookmakers' odds, despite being significantly underpowered.

### 6.2. Strengths

A few things that the project has done well are:

- Produced an easy-to-use and reproducible prediction algorithm that matches bookmakers' predictions.
- Using only final match results from one league for the last 11 years, a moderately deep neural network and a simple Poisson distribution, predicted results with an impressive accuracy.
- Understood the significance of expected goals as compared to actual goals. The use of the former automatically accounted for home advantage along with respective defensive and attacking strengths.
- Using an existing ratings database such as the one provided by EA Sports enabled the model to compactly receive real-world insights into player performance and team selection.
- The model can be easily generalized to other leagues around the world, requiring very basic data only (final results only). Detailed datasets might not be available for other leagues making more complex models suffer from sparse training dataframes.

### 6.3. Weaknesses

While the project has done well on several fronts, there are still aspects where it leaves much to be desired.

- Both the models hit an upper bound in prediction accuracy. While this could be due to the inherent variance in the results of football matches, some element of the inaccuracy can be attributed to the design of the model.
- The model oversimplifies the whole idea of a full league season. It doesn't account for form, in-game data such as number of shots, red cards, substitutions, etc., roster changes, fatigue, weather and so on. Also, the prediction accuracy has taken a turn for the worse for recent games. Given that the league was suspended for three months owing to the Covid-19 situation and all games following the resumption are played under closed doors, home advantage is virtually annulled. This has resulted in a larger proportion of away wins, which the model obviously does not anticipate.
- Overfitting in the neural network seems to be a problem as far as test accuracy is concerned. The model does much better on Top 6 teams than on newly promoted teams, primarily because we have the entire package of data for the former, whereas data is scarce for the latter.

- Betting company odds might not be a great benchmark, after all, we are basically modelling a model. Betting odds tend to be conservative at times, so as to prevent losses for the bookmakers, hence may not reflect the *true* probability. Among freely available resources, it is probably the best we have, but the odds by themselves might not paint a perfect picture.

- The model is extremely underconfident at predicting draws. Roughly 24% of Premier League matches finished in a stalemate between 2013-14 and 2017-18. Generally, the probability ranges from anywhere between 14% to 30%, based on the difference in ability of the teams. The problem arises because we are simply choosing the result with maximum probability. The probability of a draw is almost never going to be the likeliest result, hence using this metric, we will rarely predict draws. We automatically lose out on about 24% of the matches due to this. Allowing for variance in match results, we see why we have reached an upper bound on prediction accuracy with the current model.

### *6.4. Opportunities for Future Improvements*

- Given that we have achieved reasonable success for the English Premier League, there is scope for scaling up to different leagues around the world. It can also be used for forecasting more than one season down the line.

- A much more detailed analysis using in-game data can possibly provide a better prediction accuracy. Sports data providers such as Opta provide valuable match data such as ball possession, number of shots on target, events by position on the pitch, disciplinary record, etc. Also, additional information regarding the playing styles of teams can be used in predictions. Some teams tend to hold a lot of the ball and take many shots whereas others prefer to sit back and counter-attack. This information can be used to improve the notion of 'expected goals', resulting in a more accurate model.

- Throughout the model, the players playing are the same. While that obviously isn't the case, including more player-centric information such as fitness history and past disciplinary record can help us customize the team selection for every game. Whenever a significantly inferior team plays a top one, they tend to stack up defensively and wouldn't field the same team as they would when they play a team of a similar standing. Accounting for the difference in ability of teams while choosing a formation can also boost the performance of the model.

- A possible practical application of this model will be looking at maximizing returns on a real betting market. The model should be able to generate a profit in the long run. We can also explore blockchain-based betting systems such as Wagerr and Augur, where the model is not as risk-averse as bookmakers.

## References


1. M.J. Moroney. *Facts from Figures*, Penguin Books, 1951.
2. C. Reep, B. Benjamin. *Skill and chance in ball games*, Journal of the Royal Statistical Society, 1971.
3. I.D. Hill. *Association football and statistical inference*, Applied Statistics, 1974.
4. M.J. Maher. *Modelling association football scores*, Statistica Neerlandica, 1982.
5. M.J. Dixon, S.C. Coles. *Modelling association football scores and inefficiencies in the football betting market*, Applied Statistics, 1997.
6. H. Rue, O. Salvesen. *Prediction and retrospective analysis of soccer matches in a league*, Statistician, 2000.
7. D. Forrest, R. Simmons. *Forecasting sport: The behaviour and performance of football tipsters*, International Journal of Forecasting, 2000.
8. T. Kuypers. *Information and efficiency: An empirical study of a fixed odds betting market*, Applied Economics, 2000.
9. J. Goddard, I. Asimakopoulos. *Forecasting Football Results and the Efficiency of Fixed-odds Betting*, International Journal of Forecasting, 2004.
10. A. Joseph, N.E. Fenton, M. Neil. *Predicting football results using Bayesian nets and other machine learning techniques*, Knowledge-Based Systems, 2006.
11. J. Hucaljuk, A. Rakipovic. *Predicting football scores using machine learning techniques*, IEEE, 2011.
12. A. Adam. *Generalised linear model for football matches prediction*, Sports Analytics Lab, KU Leuven, 2016.
13. M. Tavakol, H. Zafartavanaelmi, U. Brefeld. *Feature Extraction and Aggregation for Predicting the Euro 2016*, Leuphana University of Luneburg, 2016.
14. P. Marek, B. Sediva, T. Toupal. *Modeling and prediction of ice hockey match results*, Journal of Quantitative Analysis in Sports, 2014.
15. R. Murphy. *FIFA player ratings explained: How are the card number and stats decided?*, goal.com, 2019.
16. V. Barnett, S. Hilditch. *The Effect of an Artificial Pitch Surface on Home Team Performance in Football (Soccer)*, Journal of the Royal Statistical Society, 1993.
17. J. Ensum, S. Taylor, M. Williams. *A quantitative analysis of goals scored, World Cup 2002*, Insight, 2004.

## Appendix

### *Libraries Used*

The entire code was written in Python 3.0 using the Google Colaboratory environment. The source code can be found here. The following Python libraries were used while writing the code for this project.

1. Scikit-learn (scikit-learn.org)
2. Pandas (pandas.pydata.org)
3. Tensorflow (tensorflow.org)

### *League Table Prediction*

A look at the current league table compared to the league table predicted by a random simulation of the league. The results appear pretty impressive. 3 teams are at the exact position predicted, 13 are within three positions of their prediction and as many as 17 are within five positions of their predicted position.

```
1. Manchester City 102
2. Liverpool 86
3. Manchester United 85
4. Chelsea 75
5. Arsenal 69
6. Tottenham Hotspur 68
7. Leicester City 54
8. Everton 48
9. Brighton & Hove Albion 45
10. Sheffield United 43
11. Southampton 43
12. Burnley 42
13. West Ham United 41
14. Wolverhampton Wanderers 41
15. Newcastle United 40
16. Bournemouth 38
17. Crystal Palace 36
18. Watford 34
19. Aston Villa 33
20. Norwich City 25
```

Fig. 14. League table for the 2019-2020 season, from a random simulation of the model

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Liverpool | 30 | 27 | 2 | 1 | 66 | 21 | 45 | **83** |
| 2 | Manchester City | 30 | 20 | 3 | 7 | 76 | 31 | 45 | **63** |
| 3 | Leicester City | 31 | 16 | 7 | 8 | 59 | 29 | 30 | **55** |
| 4 | Chelsea | 30 | 15 | 6 | 9 | 53 | 40 | 13 | **51** |
| 5 | Manchester United | 30 | 12 | 10 | 8 | 45 | 31 | 14 | **46** |
| 6 | Wolverhampton Wanderers | 30 | 11 | 13 | 6 | 43 | 34 | 9 | **46** |
| 7 | Tottenham Hotspur | 31 | 12 | 9 | 10 | 50 | 41 | 9 | **45** |
| 8 | Sheffield United | 30 | 11 | 11 | 8 | 30 | 28 | 2 | **44** |
| 9 | Crystal Palace | 30 | 11 | 9 | 10 | 28 | 32 | -4 | **42** |
| 10 | Arsenal | 30 | 9 | 13 | 8 | 41 | 41 | 0 | **40** |
| 11 | Burnley | 30 | 11 | 6 | 13 | 34 | 45 | -11 | **39** |
| 12 | Everton | 30 | 10 | 8 | 12 | 37 | 46 | -9 | **38** |
| 13 | Newcastle United | 30 | 10 | 8 | 12 | 28 | 41 | -13 | **38** |
| 14 | Southampton | 30 | 11 | 4 | 15 | 38 | 52 | -14 | **37** |
| 15 | Brighton & Hove Albion | 31 | 7 | 12 | 12 | 34 | 41 | -7 | **33** |
| 16 | Watford | 30 | 6 | 10 | 14 | 28 | 45 | -17 | **28** |
| 17 | West Ham United | 31 | 7 | 6 | 18 | 35 | 54 | -19 | **27** |
| 18 | AFC Bournemouth | 30 | 7 | 6 | 17 | 29 | 49 | -20 | **27** |
| 19 | Aston Villa | 30 | 7 | 5 | 18 | 35 | 58 | -23 | **26** |
| 20 | Norwich City | 30 | 5 | 6 | 19 | 25 | 55 | -30 | **21** |

Fig. 15. The actual league table for the 2019-2020 season, as of 24-06-2020

### *Live Updates*

I am maintaining a Google Sheet with some more statistics based on the predictions for the rest of the season as it unfolds. The sheet can be accessed here.

### *Game Simulator*

As an aside, I have also created a game simulator that plays out a match between any two teams on the screen, a la EA Sports' FIFA series. 90 minutes are condensed to about 20 seconds. The goals scored by a team are likelier to be players who generally score more goals. The goals are distributed randomly over the 90 minutes. The simulator is available in the source code notebook.

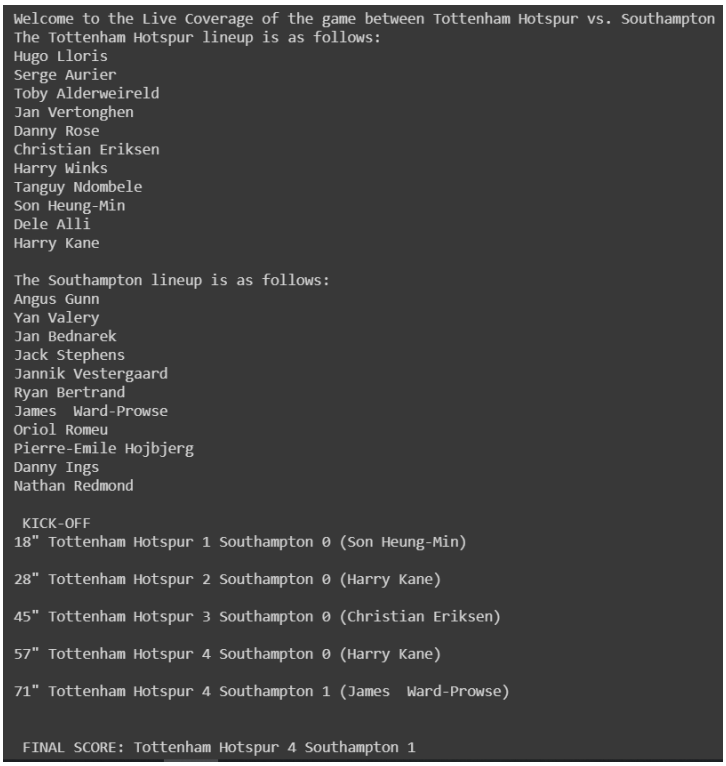


Fig. 16. A screenshot of the Game Simulator