PREDICTING THE OUTCOME OF A FOOTBALL MATCH

Student Name	Student ID	Effort
Rahul Raghavendra Prasad	180587687	50%
Sabahat Zulfiqar	180926565	50%

INDEX

TO ANALYSE AND PREDICT THE OUT COME OF A FOOTBALL MATCH

1. HYPOTHESIS:

To build a model and generated graphs or plots to compare the performance of two Teams randomly selected and predict the outcome of the match.

Key areas to consider at an initial stage

- The number of goals the team scores per game on an average.
- How well they can defend to make it tough for the opponent to score against them.
- Consistency of the performance of a team based on the position they finish in the previous year(s) league table(s).

Key features to consider at an initial stage

- Goals scored (home and away)
- Goals conceded (Home and away)
- Previous year league table position.
- Current league performance (results of 5 prior matches)
- A statistical distribution model to generate plots to aid in understanding and predicting the outcome.

2. DATASET FEATURES AND ITS ACQUISITION

Model 1

The dataset was acquired through http://football-data.co.uk. With direct pointer http://www.football-data.co.uk/mmz4281/1819/E0.csv

Model 2

The data was obtained through the link provided in the project guidance pdf. The dataset that is being used in this model is taken from www.football-data.co.uk for the year 2017-2018. The data for any year can be included by just changing the URL in the model. It will work for any year from the website. It has the complete detailed information of all the matches played in the given time period. The data holds information(columns) like the

home team and the away team for each match, the number of home goals and away goals, the date that match was played on and a lot of other things.

For the predictions, we are making use of the columns:

- Date
- Home Team
- Away Team
- FTHG which contains the Home Goals
- FTAG which contains the Away Goals
- FTR which contains the result of that match

3. DATA PRE-PROCESSING

Big data processing Map / Reduce technique and on MS Excel - > was done for this part to filter only the key features and simple computation.

This step was done to calculate

- The number of games a team had won, drawn or lost.
- The number of goals, a team had scored at home
- The number of goals, a team had scored away from home
- The number of goals, a team had conceded at home
- The number of goals, a team had conceded away from home

4. DATA EXPLORATION

By exploring the dataset, we understand the features present in the obtained dataset.

How to manipulate the dataset by comparing, calculating averages, sum of certain feature to help understand data further and in the end, help us predict the outcome of a match more accurately.

5. DATA VISUALISATION (GRAPHICAL EXPLORATION)

The graphical understanding is also acting as a validator as in some cases we are plotting the actual values as a bar graph and a line graph from the distribution model to visualise the deference between the actual and the prediction.

6. FEATURES FOR DATA ANALYSIS

The dataset that was acquired has a lot of features which are helpful, allowing us to make a more accurate prediction.

Features considered from the dataset:

- **GOALS** - A football match is decided on the final score line at full time and clearly the important feature is Goals. Goals can be further analysed as Home Goals and Away goals. In theory the teams playing at home should have more chances of scoring the goal because of the home advantage compared to Away Goals.
- **RESULT** – This feature is as important as Goals which will tell us the result of a football game helping us understand of the outcome of the game when the 2 given opponents play against each other.
- **OPPONENTS** - This helps us to see how the 2 teams in question square up against each other. We can use this information to see if one team will be dominated by the other regardless of home or away game.
- **HALF TIME SCORE** - Goals more accurately show the performance of the team in a game as it is the most relevant data to predict the outcome such as, the team most / least likely to score or most/least likely to concede.
- **PREVIOUS LEAGUE TABLE POSITION** - Though this is not directly present in the dataset there are many sources with gives the league positions. A team consistently finishing in the top 4 can be considered as a team with constant performers. And this helps us to say top 4 teams are more likely to win the game.
- **ATTACK** - given that the team that is most likely to score more goals are also more likely to win the game.
- **DEFENSE** - how well the team is organised to prevent the opponents from scoring a lot of goals.

7. DETAILS OF THE MODELS

Model 1

Jupyter notebook in anaconda was used with the following packages:

- **Pandas** is a high-performance tool which has easy to use data structures, helpful for data analysis
- **matplotlib.pyplot** A MATLAB tool used to generate plots
- **numpy** array processing tool for strings,objects,records(stored data)
- **scipy.stats** for statistics calculation
- **poisson** a statistic distribution, helpful
- **warnings**
- **seaborn**

Statistic distribution models were used after comparing the result it was decided that Poisson distribution gave us the best result. Hence, it was used in this project to create a distribution for all the teams for Home games and Away games.

For the selected statistical distribution, we are plotting against the actual data and the data generated from the distribution. Hence, we can validate our model if the outcome of the comparison is satisfactory.

Features used in the modelling include

- Home Advantage
- Away Disadvantage
- Home goals
- Away goals
- Team A
- Team B

Model 2

The model makes use of the data explained above. It includes a component of time so that the matches that are played most recently are used as a major source of average calculation of the goals. This means that the newer matches have a higher weightage because the present formation of a team is very important and is the criteria for calculating a team's performance.

For developing a model for the prediction of Football Results, several research papers were studied among which the paper **Modelling Association Football Scores and Inefficiencies in the Football Betting Market**[1] is being used as a base reference for this model.

It is a very famous research and is appreciated till today as it outlines the specifications of a comprehensive and efficient model for prediction of football results.

The language used for programming the model is Python and some of the libraries used are:

- Pandas
- Scipy
- Numpy

[1] Mark Dixon and Coles, "Modelling Association Football Scores and Inefficiencies in the Football Betting Market."

The model first checks for any errors in data by comparing that for every match, there is a home team and away team present in the data or in other words, for each home team, there is an away team playing against it.

The model works by calculating some parameters such as the home advantage, attack and defence for evaluating and predicting the number of goals each team can score. This, then, predicts which team is most likely to win or lose or if there is a likeliness of a draw between the two teams.

In the paper, Mark Dixon and Stuart Coles have mentioned the specifications of the model which are quoted below:

With the aim of developing a profitable betting strategy, various features are required of a statistical model for football matches. For example:

a) The model should consider the different abilities of both teams in a match;

b) There should be allowance for the fact that teams playing at home generally have some advantage - the so-called 'home effect';

c) The most reasonable measure of a team's ability is likely to be based on a summary measure of their recent performance;

d) The nature of football is such that a team's ability is likely to be best summarized in separate measures of their ability to attack (to score goals) and their ability to defend (not to concede goals);

e) In summarizing a team's performance by recent results, account should be taken of the ability of the teams that they have played against.

So, all the specifications mentioned above have been set as the criteria for the predictions of a match's result and all the other predictions made through the model.

The model also takes into consideration the home advantage because it is a very important criterion while predicting the outcome of a match because the predictions differ if the home team and away team are swapped.

Predictions:

For comparison and for observing the accuracy of the model, a simple Poisson model is first programmed so we can compare and see how accurate the results of the Prediction model are.

For predicting the results of the match between any two teams first using the Poisson model, the names of the home and away teams can be passed as parameters in the **match_simulation ()** function and stored in a variable.

The same can be done for the prediction model by passing the names of the home and away teams as parameters in the **model_match_simulation (params_dict, homeTeam, awayTeam, max_goals=10)** function and storing the results in a variable. They can then simply be printed.

To demonstrate this, let us predict the outcome of a match between **Chelsea** and **Watford** by using both the models and the comparing their results.

To get the result of the Poisson Model, the names of the teams i.e. Chelsea and Watford, are passed as parameters in the **match_simulation ()** function and the result is stored in the variable **cheWat** as follows:

```
cheWat = match_simulation(poisson_model, 'Chelsea', 'Watford', max_goals=10)
```

To get the result of the Prediction Model, the names of the teams i.e. Chelsea and Watford, are passed as parameters in the **model_match_simulation ()** function and the result is stored in the variable **cheWat_m** as follows:

```
cheWat_m = model_match_simulation(params, 'Chelsea', 'Watford', max_goals=10)
```

To print the results, a simple print statement could be used and if we want to print the results of both the models together, we can use the following print statement (an example of how the data can be presented) for the above example:

```
print ("Chelsea Win")

print ('; '.join("{0}: {1:.5f}".format(model, prob) for model, prob in zip(["poisson", "model"], list (map (lambda x:np.sum (np.tril(x, -1)), [cheWat, cheWat_m])))))

 print ("Watford Win")

 print ('; '.join("{0}: {1:.5f}".format(model, prob) for model,prob in zip(["poisson", "model"], list (map (lambda x:np.sum (np.triu(x, 1)), [cheWat, cheWat_m])))))
```

```
print ("Draw")

print ('; '.join("{0}: {1:.5f}".format(model, prob) for model,prob in zip(["poisson",
"model"], list (map (lambda x:np.sum (np.diag(x)), [cheWat, cheWat_m])))))
```

This will print out the results and the probabilities of each event (win, lose, draw) using both the Poisson and the prediction model.

8. CHALLENGES

Predicting an outcome of a match is very difficult as some of the factors the impact of the outcome of the game are not even present in the dataset such as referee error (awarding a false penalty, or awarding an offside goal, cancelling a valid onside goal as offside) or condition of the playing surface, goals scored against run of play. Player substitution, based on the impact like fresher legs on the field against a tired opponent leading to a mistake, team formation and change of the formation midway.

All the above-mentioned problems are not available as features and hence cannot be considered for the process of match outcome prediction. One of the reasons as to why one can never predict the outcome with high accuracy.

Many papers (approx. 10) have been studied to gain knowledge as to what features they are considering when they make the prediction. A few papers were found to be very similar concepts and with a slightly different approach There are papers which talk about calculating some sort of ranking co-efficient indicating the performance level of the team and its consistency based on previous league table positions. While there are some that consider a lot of features and even minute details to achieve higher accuracy but end up with a much worse prediction as well as increased computational time than ones that predict with few features as well as increased computational time. A paper calculating performance index for players had better accuracy, but the suspicion is that if the player left the club during previous transfer or missing through a suspension, also say the club buys a replacement we will not have an accurate index for the player as he has to get used to the new league, new team mates, managers tactical style (philosophy).

Idea Implementation to counter challenges

Hence, we decided to go in a different route and say all that will matter is the score at full-time and the consistency in performance they can achieve. This is also because a football match is won or lost because of the number of goals scored vs the number of goals conceded. Other factors that will affect the outcome is if it is played at home or away and the average number of goals scored when they play at home or away.

Also, the approach is to discard all the previous achievements of a team and start fresh where the aim is to focus on the current scenario than considering previous data which may or may not have changed since. Score line at halftime is considered to calculate more accurately the current performance of a team which should help in predicting a more accurate result.

A Poisson statistical distribution has been created indicating how likely a team can score against the opponents and the number of goals the team is likely to score.

9. SUCCESS OF THE MODELS

MODEL 1

Considering some of the important details that is not available in the dataset, we believe that we have a good accuracy of the probability for a team to score. The Purpose of this model was not designed to predict as we have another model to predict the outcome. This is only to validate it by comparing the accuracy of the models.

This information can be used in form of percentage and the average score graph over the week end to predict the scores manually. We got a clearer information when the half time score was used. But when the score was 0-0 or 1-1, which is indication of similar performance, considering half time scores did not give any edge as the performance was very close between the 2 teams.

MODEL 2

The following outcome was predicted after running the print commands:

Chelsea Win

poisson: 0.72177; model: 0.71614

Watford Win

poisson: 0.10648; model: 0.09332

Draw

poisson: 0.17172; model: 0.19050

According to the results, if Chelsea is the home team playing against Watford which is the away team, Chelsea is most likely to win the match with a high probability of 0.71614. Comparing it to the probability calculated from the Poisson model i.e. 0.72177, it is safe to say that the prediction model is quite efficient as the probabilities it is calculating are close to those calculated by using the Poisson model.

This indicates that the prediction model that has been developed has a good accuracy and is reliable for predicting results of football matches.

10. POSSIBLE EXTENSIONS

MODEL 1

In the first model, just the ability of the team to score goals is used but while writing this report the opposite end, which is defence, could also have been observed if we were able to factor in features such as ability of a team to keep clean sheets and make it hard for the opponent to score while win the match with an odd goal from a counter attack or from a set piece.

Another data that can help is how much penalty decisions are going in favour of a team which shows if the team is likely to concede a goal because of individual error.

Extensions would be to improve accuracy using better formulae for calculations, ability to use team formations. To include factors such as pressing, direct play, possession football, attacking or defensive play style. Injuries, players missing through suspensions, fitness. A sub-dataset providing accurate and current data to help the accuracy further. Also, can think of adding a simple team rating index based on the previous league table after the season has ended.

But considering the case where Leicester City fighting a relegation battle the previous season and by appointing a new manager win the Premier league the next season. Somewhat similar happened to Chelsea in previous seasons where they won the premier league during Jose Mourinho's second season and the next season they ended up finishing 10^{th} in the league. But with a new manager appointed they beat all the odds and won the league again. Similarly, to what happened the season before Chelsea finished 7^{th} and out of European championship matches.

What is being explain here is that considering only previous league table position is not giving true indication of the performance of the team. Also, the above reasoning shows that a change in manager and the philosophy that he brings with him, the training session he handles, the formation implemented, the confidence he is able to install in players misfiring, also a sense of something new for other managers to learn about the opponent.

BUSINESS APPLICATIONS

The application can be used by betting companies so they can offer a safer betting outcome making more profit. Can be used in simulation games where the results are more realistic. The application can also be used by punters so they can bet more securely by increasing their chances of winning. This application can be sold in 2 ways charge customer for every prediction or sell the application on the market.

MODEL 2

Challenges are very similar to the data to the first model since the dataset used is from the same source but for different year.

Success of the model is compared to prediction of Poisson distribution where the model performs slightly better. Performance can be further increased slightly adding some of the features that was used in model 1.

Improvements:

The predictions obtained using the prediction model are fairly closer to the results obtained from the Poisson distribution model but if further accuracy has to be achieved, it can be done by adding more features to the model. Adding features, makes the predictions more accurate as more events are then considered to decide the probability of the possible outcomes of the problem under consideration.

In the case of this prediction model, more features like the following could be added to enhance the accuracy of predictions:

- Current form of the team i.e. how many players are playing or if any player is injured or absent in a certain match. For this purpose, the player performance index must be made which can show the individual and collective contribution of the players to the team's performance. To achieve this, a detailed dataset must be used which provides historical data and the required stats to be used for this purpose.
- A team's significance in the league i.e. team's performance index. It will show how strong a team is as compared to other teams in the same league which can help shape the outcomes more accurately.
- The current coach of the team i.e. if there is a new coach or if the team performed better under a certain coach. This will help the prediction model decide the effect of the contribution of the coach to the team's significance under his coaching.

Future Extension idea

By using a Neural Network deep enough to use all the available features in the dataset and using the Generative Adversarial Networks using the generator and discriminator it should be possible to match the generated result to be very close to the actual outcome.

11. COMPARISIONS OF THE TWO MODELS

This section compares the two models used in this project. While the first model is a very simple model taking into consideration only a few selected features to predict the outcome of the game. And mostly understanding with the help of the plot.

Whereas the second model takes few variations into consideration and uses Poisson distribution as a source to validate its prediction. This was done in order to see the difference as to what happens when less features are used compared to the situation where more features are used and hence more increasing the complexity.

Key point to be noted is adding features or by varying the underlying technique indeed increased the performance when compared to the Poisson Distribution indicating that there is still the need to identify a feature that is more likely to increase the accuracy of the result.

Conclusions based on key findings

As explained in the difficulties / challenges section of the project we have come to a conclusion that historical data of the entire team is not very significant in making a prediction for the outcome of a game but having historical information of players will have a very big impact on the ability of the model to make more accurate predictions.

Historical data of the players such as

- **The age of a player** – this will decide his recovery rate in case of injury. This data is important as we can predict when the player can be back to his best fitness levels and expect to know the length of the injury. Also, as the age of the player increases his physical condition also deteriorates which can impact performance
- **Performance** – having a performance rating it show the level of his performance and a graph can further show his consistency.
- **Injury prone** – to identify how likely a player is likely to get injured
- **Goals** – the ability to score goals for the team. This data also can be used by oppositions it identifies the threat and prepare to deal with a possible threat to decrease the chance to concede a goal and increase the chances of winning.
- **Assists** – identifying the key creator of chances for the team to help in defensive setup
- **Heat map generation** – by analysing the heat map of the player it can be seen the most likely position he might take on the pitch this can be used to nullify the effect of that player from oppositions point of view
- **Confidence** – a players especially for a striker confidence in front of goal or when 1 on 1 against a goal keeper confidence plays a huge part in scoring a goal. Generally speaking, a player with high confidence is less likely to make mistakes and performs better.
- **Red Card liability** – This is sometimes very significant as it can turn the match on its head a team winning comfortable can end up losing the game.
- **Set pieces** – Some players have great ability to score from set pieces I.e. from free kicks and corners.
- **Concentration** – A player need to read the game and must concentrate on the game in failing to do so can lead to a mistake with can hurt the team's chances of winning a match sometimes might also lead to a loss.

USING CLASSIFICATION TO IDENTIFY KEY STATISTICS OF THE PLAYER

A football player can be classified into following types and it is necessary they have the right attributes to operate in that position.

Key attributes to consider as

- **Striker** – chance to goal conversion rate, amount of goals he scores on an average every season.
- **Defender** – A defender's performance can be rated based on important tackles he makes to get the position of the ball. The number of clearances he makes that can be from open play or set pieces. And Blocks which might lead to a difference from the opponents scoring a goal and stopping from conceding a goal.
- **Midfielder** - Midfielders are key in build-up of play or helping in defence. The key stats for any midfield player are the accuracy at which he passes the ball, pass completion ratio i.e. the number of attempted passes and the number of passes that reach its intended target.
- **Goalkeeper** – The ability of the goal keeper to make important saves will sometimes result in not losing or even winning instead of a draw. Less goals conceded means the strikers will also need to score fewer goals and still have a good chance of winning a match.

Final Thoughts

Overall by creating a dataset for the above-mentioned player performance rating and key attribute of a player based on classification can lead to more accurate predictions.

Constantly updating the dataset will also ensure that the most recent data is available for analysis.

Paper Reference

1. https://www.researchgate.net/publication/328486514_Predicting_Match_Outcomes_in_Football_by_an_Ordered_Forest_Estimator
2. https://pdfs.semanticscholar.org/22cf/d53bd9d43bb54c90d5b80e611de7a56783bd.pdf
3. http://www.ijcte.org/papers/802-N30016.pdf
4. https://www.idi.ntnu.no/~helgel/papers/LangsethSCAI14.pdf

   Thank you, Mr Anthony Constantinou for providing these 3 papers to study for this project:

5. https://www.researchgate.net/publication/236944355_pi-football_A_Bayesian_network_model_for_forecasting_Association_Football_match_outcomes_Knowledge-Based_Systems_36_322-339
6. https://www.researchgate.net/publication/228808917_Evaluating_the_Predictive_Accuracy_of_Association_Football_Forecasting_Systems
7. https://www.researchgate.net/publication/266874470_-1_-Evidence_of_an_intended_inefficient_Association_Football_gambling_market

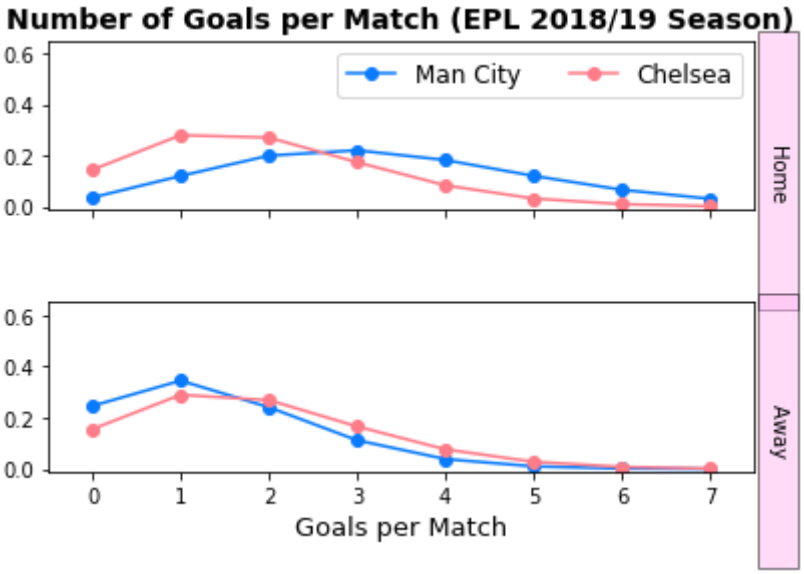
12. APPENDIX

Code Model 1

Pre-Processing:

```
"""
@author: RRP
"""
#to import the dataset which is used for predicting the outcome
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import poisson,skellam
import warnings
warnings.filterwarnings('ignore')

epl_1617 = pd.read_csv("http://www.football-data.co.uk/mmz4281/1819/E0.csv")
epl_1617 = epl_1617[['HomeTeam','AwayTeam','FTHG','FTAG']]
epl_1617 = epl_1617.rename(columns={'FTHG': 'HomeGoals', 'FTAG': 'AwayGoals'})
epl_1617.head()
#this is used to generate a plot for home goals and away goals for man city and chelsea
fig,(ax1,ax2) = plt.subplots(2, 1)

pois1, = ax1.plot([i for i in range(8)], chel_home_pois,
        linestyle='-', marker='o',label="Man City", color = "#0a7bff")
pois1, = ax1.plot([i for i in range(8)], sun_home_pois,
        linestyle='-', marker='o',label="Chelsea", color = "#ff7c89")
leg=ax1.legend(loc='upper right', fontsize=12, ncol=2)
ax1.set_xlim([-0.5,7.5])
ax1.set_ylim([-0.01,0.65])
```

```
ax1.set_xticklabels([])

ax1.text(7.65, 0.585, '              Home              ', rotation=-90,
      bbox={'facecolor':'#ffbcf6', 'alpha':0.5, 'pad':5})
ax2.text(7.65, 0.585, '              Away              ', rotation=-90,
      bbox={'facecolor':'#ffbcf6', 'alpha':0.5, 'pad':5})

pois1, = ax2.plot([i for i in range(8)], chel_away_pois,
            linestyle='-', marker='o',label="Man City", color = "#0a7bff")
pois1, = ax2.plot([i for i in range(8)], sun_away_pois,
            linestyle='-', marker='o',label="Chelsea", color = "#ff7c89")
ax2.set_xlim([-0.5,7.5])
ax2.set_ylim([-0.01,0.65])
ax1.set_title("Number of Goals per Match (EPL 2018/19 Season)",size=14,fontweight='bold')
ax2.set_xlabel("Goals per Match",size=13)
ax2.text(-1.15, 0.9, '', rotation=90, size=13)
plt.tight_layout()
plt.show()
```



#this section is to create a distribution for out dataset to see the amount of goals scored

```
import statsmodels.api as sm

import statsmodels.formula.api as smf


goal_data =
pd.concat([epl_1617[['HomeTeam','AwayTeam','HomeGoals']].assign(home=1).rename(colu
mns={'HomeTeam':'team', 'AwayTeam':'opponent','HomeGoals':'goals'}),


epl_1617[['AwayTeam','HomeTeam','AwayGoals']].assign(home=0).rename(columns={'Away
Team':'team', 'HomeTeam':'opponent','AwayGoals':'goals'})])


#to generate a statistical model (poisson)

poisson_model = smf.glm(formula="goals ~ home + team + opponent", data=goal_data,

                        family=sm.families.Poisson()).fit()

poisson_model.summary()
```

Generalized Linear Model Regression Results

Dep. Variable:	goals	**No. Observations:**	200
Model:	GLM	**Df Residuals:**	160
Model Family:	Poisson	**Df Model:**	39
Link Function:	log	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-254.57
Date:	Mon, 01 Apr 2019	**Deviance:**	157.66
Time:	20:41:27	**Pearson chi2:**	139.
No. Iterations:	6	**Covariance Type:**	nonrobust

	coef	**std errz**	**P>|z|**	**[0.025**	**0.975]**
Intercept	0.7522	0.368	2.046	0.041	0.032 1.473
team[T.Bournemouth]	-0.3740	0.311	-1.202	0.229	-0.984 0.236
team[T.Brighton]	-0.6436	0.377	-1.705	0.088	-1.383 0.096
team[T.Burnley]	-0.8997	0.388	-2.318	0.020	-1.661 -0.139
team[T.Cardiff]	-0.9050	0.407	-2.226	0.026	-1.702 -0.108
team[T.Chelsea]	-0.0462	0.310	-0.149	0.881	-0.653 0.561
team[T.Crystal Palace]	-1.2145	0.440	-2.759	0.006	-2.077 -0.352
team[T.Everton]	-0.5155	0.336	-1.532	0.125	-1.175 0.144
team[T.Fulham]	-0.7516	0.376	-1.997	0.046	-1.489 -0.014
team[T.Huddersfield]	-1.6669	0.545	-3.061	0.002	-2.734 -0.600
team[T.Leicester]	-0.3279	0.342	-0.959	0.338	-0.998 0.342
team[T.Liverpool]	-0.0932	0.315	-0.296	0.767	-0.710 0.524
team[T.Man City]	0.0603	0.300	0.201	0.841	-0.529 0.649
team[T.Man United]	-0.2624	0.327	-0.801	0.423	-0.904 0.379

team[T.Newcastle]	-1.2963	0.466	-2.779	0.005	-2.210	-0.382
team[T.Southampton]	-1.3002	0.466	-2.790	0.005	-2.214	-0.387
team[T.Tottenham]	-0.3944	0.333	-1.184	0.236	-1.047	0.258
team[T.Watford]	-0.5115	0.342	-1.495	0.135	-1.182	0.159
team[T.West Ham]	-0.8149	0.407	-2.004	0.045	-1.612	-0.018
team[T.Wolves]	-0.9850	0.400	-2.465	0.014	-1.768	-0.202
opponent[T.Bournemouth]	0.0224	0.404	0.055	0.956	-0.770	0.815
opponent[T.Brighton]	-0.0189	0.404	-0.047	0.963	-0.810	0.772
opponent[T.Burnley]	0.4821	0.362	1.332	0.183	-0.228	1.192
opponent[T.Cardiff]	0.4284	0.360	1.189	0.234	-0.278	1.134
opponent[T.Chelsea]	-0.5214	0.484	-1.077	0.281	-1.470	0.427
opponent[T.Crystal Palace]	0.0727	0.407	0.178	0.858	-0.726	0.871
opponent[T.Everton]	0.1512	0.403	0.375	0.708	-0.639	0.941
opponent[T.Fulham]	0.6828	0.348	1.961	0.050	0.000	1.365
opponent[T.Huddersfield]	0.3385	0.359	0.944	0.345	-0.365	1.042
opponent[T.Leicester]	0.2460	0.393	0.626	0.532	-0.525	1.017
opponent[T.Liverpool]	-1.1410	0.576	-1.980	0.048	-2.271	-0.011
opponent[T.Man City]	-1.3861	0.652	-2.124	0.034	-2.665	-0.107
opponent[T.Man United]	0.2995	0.377	0.794	0.427	-0.440	1.039
opponent[T.Newcastle]	-0.0640	0.394	-0.163	0.871	-0.836	0.708
opponent[T.Southampton]	0.0769	0.397	0.194	0.846	-0.700	0.854
opponent[T.Tottenham]	-0.4650	0.458	-1.015	0.310	-1.363	0.433
opponent[T.Watford]	-0.0041	0.419	-0.010	0.992	-0.826	0.818
opponent[T.West Ham]	-0.0839	0.393	-0.214	0.831	-0.854	0.686
opponent[T.Wolves]	-0.3536	0.441	-0.802	0.423	-1.218	0.511
home	0.1192	0.123	0.969	0.333	-0.122	0.360

#man city at home to chelsea

```
poisson_model.predict(pd.DataFrame(data={'team': 'Man City', 'opponent':
'Chelsea','home':1},index=[1]))
```

#chelsea at home to man city

```
poisson_model.predict(pd.DataFrame(data={'team': 'Chelsea', 'opponent': 'Man City',
'home':0},index=[1]))
```

#using simulation tool to get the prediction

```
def simulate_match(foot_model, homeTeam, awayTeam, max_goals=10):
    home_goals_avg = foot_model.predict(pd.DataFrame(data={'team': homeTeam,
                                        'opponent': awayTeam,'home':1},
                                    index=[1])).values[0]
    away_goals_avg = foot_model.predict(pd.DataFrame(data={'team': awayTeam,
                                        'opponent': homeTeam,'home':0},
```

```
                                    index=[1])).values[0]
    team_pred = [[poisson.pmf(i, team_avg) for i in range(0, max_goals+1)] for team_avg in
[home_goals_avg, away_goals_avg]]
    return(np.outer(np.array(team_pred[0]), np.array(team_pred[1])))
simulate_match(poisson_model, 'Chelsea', 'Man City', max_goals=3)


chel_city = simulate_match(poisson_model, "Chelsea", "Man City", max_goals=10)


# Man City win
np.sum(np.tril(chel_city, -1))
0.19483972307622643
# draw
np.sum(np.diag(chel_city))
0.22942171658926716
# chelsea win
np.sum(np.triu(chel_city, 1))
0.5757353697235998
```

Model 2

```
"""
Created on Mon Mar 18 17:06:59 2019

@author: Sabahat
"""
import pandas as pd
import numpy as np
from scipy.stats import poisson,skellam
from scipy.optimize import minimize, fmin
import statsmodels.api as sm
import statsmodels.formula.api as smf
```

```
data1718 = pd.read_csv("http://www.football-data.co.uk/mmz4281/1718/E0.csv")
data1718['Date'] = pd.to_datetime(data1718['Date'],  format='%d/%m/%y')
data1718['time_diff'] = (max(data1718['Date']) - data1718['Date']).dt.days
data1718 = data1718[['HomeTeam','AwayTeam','FTHG','FTAG', 'FTR', 'time_diff']]
data1718 = data1718.rename(columns={'FTHG': 'HomeGoals', 'FTAG': 'AwayGoals'})


goal_model_data =
pd.concat([data1718[['HomeTeam','AwayTeam','HomeGoals']].assign(home=1).rename(
        columns={'HomeTeam':'team', 'AwayTeam':'opponent','HomeGoals':'goals'}),
       data1718[['AwayTeam','HomeTeam','AwayGoals']].assign(home=0).rename(
        columns={'AwayTeam':'team', 'HomeTeam':'opponent','AwayGoals':'goals'})])


poisson_model = smf.glm(formula="goals ~ home + team + opponent",
data=goal_model_data,
                family=sm.families.Poisson()).fit()


def calculate_means(param_dict, homeTeam, awayTeam):
  return [np.exp(param_dict['attack_'+homeTeam] + param_dict['defence_'+awayTeam] +
param_dict['home_adv']), np.exp(param_dict['defence_'+homeTeam] +
param_dict['attack_'+awayTeam])]


def roh_calc(x, y, lambda_x, mu_y, roh):
  if x==0 and y==0:
    return 1- (lambda_x * mu_y * roh)
  elif x==0 and y==1:
    return 1 + (lambda_x * roh)
  elif x==1 and y==0:
    return 1 + (mu_y * roh)
  elif x==1 and y==1:
    return 1 - roh
  else:
```

```
        return 1.0


def solve_parameters(dataset, debug = False, init_vals=None, options={'disp': True,
'maxiter':100}, constraints = [{'type':'eq', 'fun': lambda x: sum(x[:20])-20}] , **kwargs):
    teams = np.sort(dataset['HomeTeam'].unique())
    # check dataset
    away_teams = np.sort(dataset['AwayTeam'].unique())
    if not np.array_equal(teams, away_teams):
        raise ValueError("Incorrect data")
    n_teams = len(teams)
    if init_vals is None:
        # random initialisation of model parameters
        init_vals = np.concatenate((np.random.uniform(0,1,(n_teams)), # strength of attack
                            np.random.uniform(0,-1,(n_teams)), # strength of defence
                            np.array([0, 1.0]) # roh (score correction), gamma (home advantage)
                            ))

    def dc_log_like(x, y, alpha_x, beta_x, alpha_y, beta_y, roh, gamma):
        lambda_x, mu_y = np.exp(alpha_x + beta_y + gamma), np.exp(alpha_y + beta_x)
        return (np.log(roh_calc(x, y, lambda_x, mu_y, roh)) +
                np.log(poisson.pmf(x, lambda_x)) + np.log(poisson.pmf(y, mu_y)))


    def estimate_paramters(params):
        score = dict(zip(teams, params[:n_teams]))
        defend_coefs = dict(zip(teams, params[n_teams:(2*n_teams)]))
        roh, gamma = params[-2:]
        log_like = [dc_log_like(row.HomeGoals, row.AwayGoals, score[row.HomeTeam],
defend_coefs[row.HomeTeam],
                    score[row.AwayTeam], defend_coefs[row.AwayTeam], roh, gamma) for row in
dataset.itertuples()]
        return -sum(log_like)
```

```
    opt_output = minimize(estimate_paramters, init_vals, options=options, constraints =
constraints, **kwargs)
    if debug:
        return opt_output
    else:
        return dict(zip(["attack_"+team for team in teams] +
                        ["defence_"+team for team in teams] +
                        ['roh', 'home_adv'],
                        opt_output.x))


def match_simulation(foot_model, homeTeam, awayTeam, max_goals=10):
    home_goals_avg = foot_model.predict(pd.DataFrame(data={'team': homeTeam,
                                            'opponent': awayTeam,'home':1},
                                        index=[1])).values[0]
    away_goals_avg = foot_model.predict(pd.DataFrame(data={'team': awayTeam,
                                            'opponent': homeTeam,'home':0},
                                        index=[1])).values[0]
    team_prediction = [[poisson.pmf(i, team_avg) for i in range(0, max_goals+1)] for
team_avg in [home_goals_avg, away_goals_avg]]
    return(np.outer(np.array(team_prediction[0]), np.array(team_prediction[1])))


def model_match_simulation(params_dict, homeTeam, awayTeam, max_goals=10):
    team_avgs = calculate_means(params_dict, homeTeam, awayTeam)
    team_prediction = [[poisson.pmf(i, team_avg) for i in range(0, max_goals+1)] for
team_avg in team_avgs]
    output_matrix = np.outer(np.array(team_prediction[0]), np.array(team_prediction[1]))
    correction_matrix = np.array([[roh_calc(home_goals, away_goals, team_avgs[0],
                                            team_avgs[1], params_dict['roh']) for away_goals in range(2)]
                                  for home_goals in range(2)])
    output_matrix[:2,:2] = output_matrix[:2,:2] * correction_matrix
    return output_matrix
```

```
if __name__ == '__main__':


    params = solve_parameters(data1718)

    cheWat = match_simulation(poisson_model, 'Chelsea', 'Watford', max_goals=10)

    cheWat_m = model_match_simulation(params, 'Chelsea', 'Watford', max_goals=10)

    print("Chelsea Win")

    print('; '.join("{0}: {1:.5f}".format(model, prob) for model,prob in zip(["poisson", "model"],
list(map(lambda x:np.sum(np.tril(x, -1)), [cheWat, cheWat_m])))))

    print("Watford Win")

    print('; '.join("{0}: {1:.5f}".format(model, prob) for model,prob in zip(["poisson", "model"],
list(map(lambda x:np.sum(np.triu(x, 1)), [cheWat, cheWat_m])))))

    print("Draw")

    print('; '.join("{0}: {1:.5f}".format(model, prob) for model,prob in zip(["poisson", "model"],
list(map(lambda x:np.sum(np.diag(x)), [cheWat, cheWat_m])))))
```

Data Sample

The first 10 rows of the dataset are provided below as well as the links of the datasets used in the models.

Model 1

http://www.football-data.co.uk/mmz4281/1819/E0.csv

Model 2

http://www.football-data.co.uk/mmz4281/1718/E0.csv

Div	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR
E0	11/08/2017	Arsenal	Leicester	4	3	H	2	2	D
E0	12/08/2017	Brighton	Man City	0	2	A	0	0	D
E0	12/08/2017	Chelsea	Burnley	2	3	A	0	3	A
E0	12/08/2017	Crystal Palac	Huddersfield	0	3	A	0	2	A
E0	12/08/2017	Everton	Stoke	1	0	H	1	0	H
E0	12/08/2017	Southamptor	Swansea	0	0	D	0	0	D
E0	12/08/2017	Watford	Liverpool	3	3	D	2	1	H
E0	12/08/2017	West Brom	Bournemout	1	0	H	1	0	H
E0	13/08/2017	Man United	West Ham	4	0	H	1	0	H

Referee	HS	AS	HST	AST	HF	AF	HC	AC	HY
M Dean	27	6	10	3	9	12	9	4	0
M Oliver	6	14	2	4	6	9	3	10	0
C Pawson	19	10	6	5	16	11	8	5	3
J Moss	14	8	4	6	7	19	12	9	1
N Swarbrick	9	9	4	1	13	10	6	7	1
M Jones	29	4	2	0	10	13	13	0	2
A Taylor	9	14	4	5	14	8	3	3	0
R Madley	16	9	6	2	15	3	8	2	3
M Atkinson	22	9	6	1	19	7	11	1	2

AY	HR	AR	B365H	B365D	B365A	BWH	BWD	BWA	IWH
1	0	0	1.53	4.5	6.5	1.5	4.6	6.75	1.47
2	0	0	11	5.5	1.33	11	5.25	1.3	8
3	2	0	1.25	6.5	15	1.22	6.5	12.5	1.22
3	0	0	1.83	3.6	5	1.8	3.5	4.75	1.85
1	0	0	1.7	3.8	5.75	1.7	3.6	5.5	1.7
1	0	0	1.62	4	6.5	1.57	4	6	1.65
3	0	0	6	4.2	1.62	6	4.2	1.55	5.5
1	0	0	2.4	3.3	3.3	2.4	3.2	3.1	2.3
2	0	0	1.3	5.75	12	1.28	5.5	11	1.33

IWD	IWA	LBH	LBD	LBA	PSH	PSD	PSA	WHH	WHD
4.5	6.5	1.44	4.4	6.5	1.53	4.55	6.85	1.53	4.2
5.3	1.35	10	5	1.3	10.95	5.55	1.34	10	4.8
6.2	13.5	1.25	5.75	15	1.26	6.3	15.25	1.25	5.5
3.5	4.3	1.8	3.4	4.6	1.83	3.58	5.11	1.8	3.3
3.7	5	1.67	3.6	5.25	1.7	3.83	5.81	1.7	3.5
3.8	5.3	1.6	3.7	6	1.64	3.94	6.35	1.62	3.6
4	1.6	5.8	4	1.57	5.74	4.29	1.63	5.5	3.8
3.3	3.15	2.4	3.1	3	2.46	3.25	3.26	2.5	3.1
5.3	8.7	1.33	5	10	1.33	5.68	10.85	1.3	5

WHA	VCH	VCD	VCA	Bb1X2	BbMxH	BbAvH	BbMxD	BbAvD
6	1.53	4.5	6.5	41	1.55	1.51	4.6	4.43
1.33	10	5.5	1.33	40	11.5	10.1	5.6	5.25
13	1.25	6.25	15	41	1.27	1.24	6.55	6.06
5	1.83	3.6	5	41	1.86	1.81	3.65	3.5
5.5	1.7	3.8	5.75	40	1.71	1.69	3.85	3.69
6	1.65	4	5.5	41	1.66	1.61	4.05	3.84
1.62	6	4	1.65	41	6.5	5.75	4.3	4.06
3	2.5	3.3	3.13	41	2.5	2.43	3.3	3.19
11	1.3	5.5	11.5	40	1.35	1.31	5.75	5.29

BbMxA	BbAvA	BbOU	BbMx>2.5	BbAv>2.5	BbMx<2.5	BbAv<2.5	BbAH
6.89	6.44	37	1.65	1.61	2.43	2.32	21
1.36	1.32	35	1.7	1.63	2.4	2.27	20
15.5	13.67	36	1.71	1.66	2.33	2.23	20
5.11	4.82	36	2.19	2.11	1.79	1.72	18
6	5.5	35	2.17	2.08	1.8	1.76	19
6.5	5.98	36	2.17	2.08	1.8	1.75	19
1.65	1.6	37	1.89	1.82	2.08	1.99	21
3.3	3.11	36	2.25	2.16	1.75	1.7	22
13	10.74	35	1.76	1.7	2.25	2.16	19

BbAHh	BbMxAHH	BbAvAHH	BbMxAHA	BbAvAHA	PSCH	PSCD	PSCA
-1	1.91	1.85	2.1	2.02	1.49	4.73	7.25
1.5	1.95	1.91	2.01	1.96	11.75	6.15	1.29
-1.75	2.03	1.97	1.95	1.9	1.33	5.4	12.25
-0.75	2.1	2.05	1.86	1.83	1.79	3.56	5.51
-0.75	1.94	1.9	2.01	1.98	1.82	3.49	5.42
-0.75	1.83	1.78	2.16	2.1	1.56	4.25	6.85
1	1.9	1.84	2.13	2.04	6.88	4.27	1.56
-0.25	2.12	2.08	1.85	1.81	2.65	3.21	3.02
-1.5	2.01	1.96	1.95	1.92	1.31	5.79	12.01