Title

Modelling Football as a Markov Process.

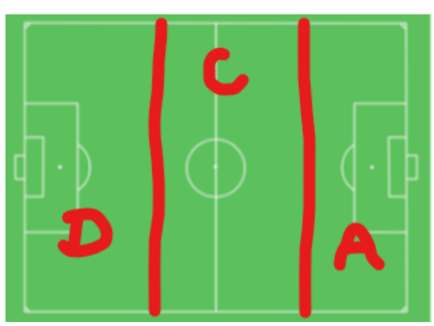
Abstract

This innovative project aims to model the Football Game Set Pieces (i.e. Throw Ins, Free Kicks, Goal Kicks and Corners) using Markov Theory. The aim is to construct a Markov Chain depicting flow of game between transition states connected through simple passes.

Then by Regression Analysis on a various range of covariates we approximate the single step Transition Probabilities of such a process from one state to another. The basic objective is to show that even in dynamic sports such as Football we can use Stochastic Processes (here the concept of Markov Chains) to predict the outcome of certain Set Pieces and the probability of going into that Set Pieces in any game, we consider the factor of Team, Timeline of the game and current score of the game.

Introduction

The problem we are trying to address is that Can we model the outcomes of a set piece in football as a Markov Chain and what factors are prone to alter the transition probabilities and how can we integrate these into the modeling, some of the definitions and assumptions are that we include Passes to as the connecting factor between various States in a Markov Chain, and we divide the field into three locations namely, Attack(A), Defence(D) and Central/Safe Zone(C). One other assumptions that the ability of team to move from one transition states to next does not change with time and hence it is Memoryless Markov Chains as the ability to form an Attacking Pass in 60 min and in 30 min would be the same, the only factors affecting will be the score, time spent from beginning and odds of winning of a team, in other words we can say that probability of ball being played from state X to state Y depends solely on the current state X.



Literature Studied

1. **Markov Theory**

This project uses the theory of Markov Processes and Markov Chains, we define it as a Stochastic Process which is a Markov Chain if

$$P(X(t_{n+1}) = i_{n+1} | X(t_n) = i_n, X(t_{n-1}) = i_{n-1}, ..., X(t_0) = i_0) \\ = P(X(t_{n+1}) = i_{n+1} | X(t_n) = i_n) \quad \forall n \wedge i$$

Where i is any state within a given Markov Chain, $i \in \Omega = \{1, ..., N\}$

Which is the probability space containing all possible states within a given Markov Process, here the process does not consider the "historic jumps" between states, this core element constitutes Markov Process property namely the Markov Property and stochastic process is memoryless.

We now use the concepts of Transition Probabilities of Time independent homogeneous Markov Chain.

$$p_{ij} = P(X_n = j | X_{n-1} = i) \quad i, j \in \Omega$$

And we can represent these single step transition using a Matrix, P defined

$$\boldsymbol{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1N} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2N} \\ p_{31} & p_{32} & p_{33} & \cdots & p_{3N} \\ \vdots & \vdots & \vdots & \ddots & \\ p_{N1} & p_{N2} & p_{N3} & \cdots & p_{NN} \end{bmatrix} \qquad \sum_{i=1}^{N} p_{ri} = 1 \quad \forall r \in \Omega$$

1.1 Chapman Kolmogrov

These are the identities relating the joint probability distribution of different sets of coordinates on a Stochastic Process, we use it for finding transition matrix after a certain (say n) amounts of jumps as

$$a) \qquad p_{ij}^{(m+n)} = \sum_{k \in \Omega} p_{ik}^{(m)} p_{kj}^{(n)}$$

$$b) \qquad \boldsymbol{P}^{(m+n)} = \boldsymbol{P}^m \boldsymbol{P}^n$$

$$c) \qquad \boldsymbol{P}^{(n)} = \boldsymbol{P}^n$$

$$d) \qquad \boldsymbol{p}^{(n)} = \boldsymbol{p}^{(0)} \boldsymbol{P}^{(n)} = \boldsymbol{p}^{(0)} \boldsymbol{P}^n$$

1.2 Absorbing States and Irreducibility

We will also use the concept of Absorbing States, the state in which we can enter and then never leave, hence for any absorbing states the probability of absorption is 1 over long time considerations, hence in our case the goal will eventually be scored after we have iterated over and over.

All the states we are considering are communicating and the Markov Chain is Irreducible, all of the states are Transient and we can either create a DTMC or a CTMC depending on the requirement of when to measure the state of the game.

2. Regression Analysis

For approximating the probabilities, we use the regression analysis, in which our data points or values will be estimated by considering relations between different parameters.

2.1 The Logit Regression

The Logistic Regression model is used when we consider probabilistic variables, and we do not use any other regression model.

$$y_i = \frac{e^{x_i\beta}}{1+e^{x_i\beta}} = p(x_i\beta), \qquad i = 1, \ldots, n$$

Yi is the observation of the dependent stochastic random variable and n is the number of observations, x are the covariates and beta is coefficients.

The observational data we are going to use will be dummy variables which will be variables with value either zero or one. Where we consider 1 when there is a possible jump option otherwise not.

Then we try to use the **Log-Likelihood Function** try to maximise the estimation of Y from the equation,

$$ln(L) = \sum_{i=1}^{n} ln[(2d_i - 1)p(x_k\hat{\beta}) + 1 - d_i]$$

There are many other tests like ***Likelihood-Ratio Test, Wald-Test, AIC-Test and Goodness of Fit Test***, we can take use of them to improve upon the present model and for achieving better idea of the probability of jumping from one state to another.

Problem Undertaken

The major problem here is to find and model the football set pieces in terms of Markov Chains, and then use Regression on obtained Probabilities to get better results on the Model.

Computations

1. **Setting Up Markovian Model**

We assume the memoryless-ness property of the states we can set up the Transition Matrix, now since here we can say one more thing that the field-location will have impact on certain states, Goal Kicks and Corners will be location independent but Free Kicks, Passes and Throw Ins will be Field Location dependent. Hence we make these 3 states into 9 corresponding states dividing each of them on A, C, D bases, there are a total 11 states.

$$\begin{bmatrix}
p_{p_cp_c}(\theta) & p_{p_cp_a}(\theta) & p_{p_cp_d}(\theta) & p_{p_ct_c}(\theta) & p_{p_ct_a}(\theta) & p_{p_ct_d}(\theta) & p_{p_cf_c}(\theta) & p_{p_cf_a}(\theta) & p_{p_cf_d}(\theta) & p_{p_cg}(\theta) & p_{p_cc}(\theta) \\
p_{p_ap_c}(\theta) & p_{p_ap_a}(\theta) & p_{p_ap_a}(\theta) & p_{p_at_c}(\theta) & p_{p_at_a}(\theta) & p_{p_at_d}(\theta) & p_{p_af_c}(\theta) & p_{p_af_a}(\theta) & p_{p_af_d}(\theta) & p_{p_ag}(\theta) & p_{p_ac}(\theta) \\
p_{p_dp_c}(\theta) & p_{p_dp_a}(\theta) & p_{p_dp_a}(\theta) & p_{p_dt_c}(\theta) & p_{p_dt_a}(\theta) & p_{p_dt_d}(\theta) & p_{p_df_c}(\theta) & p_{p_df_a}(\theta) & p_{p_df_d}(\theta) & p_{p_dg}(\theta) & p_{p_dc}(\theta) \\
p_{t_cp_c}(\theta) & p_{t_cp_a}(\theta) & p_{t_cp_a}(\theta) & p_{t_ct_c}(\theta) & p_{t_ct_a}(\theta) & p_{t_ct_d}(\theta) & p_{t_cf_c}(\theta) & p_{t_cf_a}(\theta) & p_{t_cf_d}(\theta) & p_{t_cg}(\theta) & p_{t_cc}(\theta) \\
p_{t_ap_c}(\theta) & p_{t_ap_a}(\theta) & p_{t_ap_a}(\theta) & p_{t_at_c}(\theta) & p_{t_at_a}(\theta) & p_{t_at_d}(\theta) & p_{t_af_c}(\theta) & p_{t_af_a}(\theta) & p_{t_af_d}(\theta) & p_{t_ag}(\theta) & p_{t_ac}(\theta) \\
p_{t_dp_c}(\theta) & p_{t_dp_a}(\theta) & p_{t_dp_a}(\theta) & p_{t_dt_c}(\theta) & p_{t_dt_a}(\theta) & p_{t_dt_d}(\theta) & p_{t_df_c}(\theta) & p_{t_df_a}(\theta) & p_{t_df_d}(\theta) & p_{t_dg}(\theta) & p_{t_dc}(\theta) \\
p_{f_cp_c}(\theta) & p_{f_cp_a}(\theta) & p_{f_cp_a}(\theta) & p_{f_ct_c}(\theta) & p_{f_ct_a}(\theta) & p_{f_ct_d}(\theta) & p_{f_cf_c}(\theta) & p_{f_cf_a}(\theta) & p_{f_cf_d}(\theta) & p_{f_cg}(\theta) & p_{f_cc}(\theta) \\
p_{f_ap_c}(\theta) & p_{f_ap_a}(\theta) & p_{f_ap_a}(\theta) & p_{f_at_c}(\theta) & p_{f_at_a}(\theta) & p_{f_at_d}(\theta) & p_{f_af_c}(\theta) & p_{f_af_a}(\theta) & p_{f_af_d}(\theta) & p_{f_ag}(\theta) & p_{f_ac}(\theta) \\
p_{f_dp_c}(\theta) & p_{f_dp_a}(\theta) & p_{f_dp_a}(\theta) & p_{f_dt_c}(\theta) & p_{f_dt_a}(\theta) & p_{f_dt_d}(\theta) & p_{f_df_c}(\theta) & p_{f_df_a}(\theta) & p_{f_df_d}(\theta) & p_{f_dg}(\theta) & p_{f_dc}(\theta) \\
p_{gp_c}(\theta) & p_{gp_a}(\theta) & p_{gp_a}(\theta) & p_{gt_c}(\theta) & p_{gt_a}(\theta) & p_{gt_d}(\theta) & p_{gf_c}(\theta) & p_{gf_a}(\theta) & p_{gf_d}(\theta) & p_{gg}(\theta) & p_{gc}(\theta) \\
p_{cp_c}(\theta) & p_{cp_a}(\theta) & p_{cp_a}(\theta) & p_{ct_c}(\theta) & p_{ct_a}(\theta) & p_{ct_d}(\theta) & p_{cf_c}(\theta) & p_{cf_a}(\theta) & p_{cf_d}(\theta) & p_{cg}(\theta) & p_{cc}(\theta)
\end{bmatrix}$$

Here Pc is the pass in central zone, Pa is an offensive pass, Pd is a defensive pass, thus the probability space will be

$$\Omega = \{p_c, p_a, p_d, t_c, t_a, t_d, f_c, f_a, f_d, g, c\}$$

Theta in the Transition Matrix is the parameter vector containing induce modifications of transition probabilities, which will be

1. Time has an impact on transition probabilities (depends game is at 30 min or 90 min)
2. The score-line affects the player's ability to transition.
3. The odds of winning (determines which team is stronger) also impacts the game, assume odds for Team 1 and 2 are a, b then:-

$$c = \frac{a}{(a+b)} \qquad d = \frac{b}{(a+b)}$$

and Theta(odds)

$$\theta_{odds} = \left|\frac{1}{c} - \frac{1}{d}\right|$$

$$\theta = \{\theta_{time}, \theta_{score}, \theta_{odds}\}$$

2. Data Collection

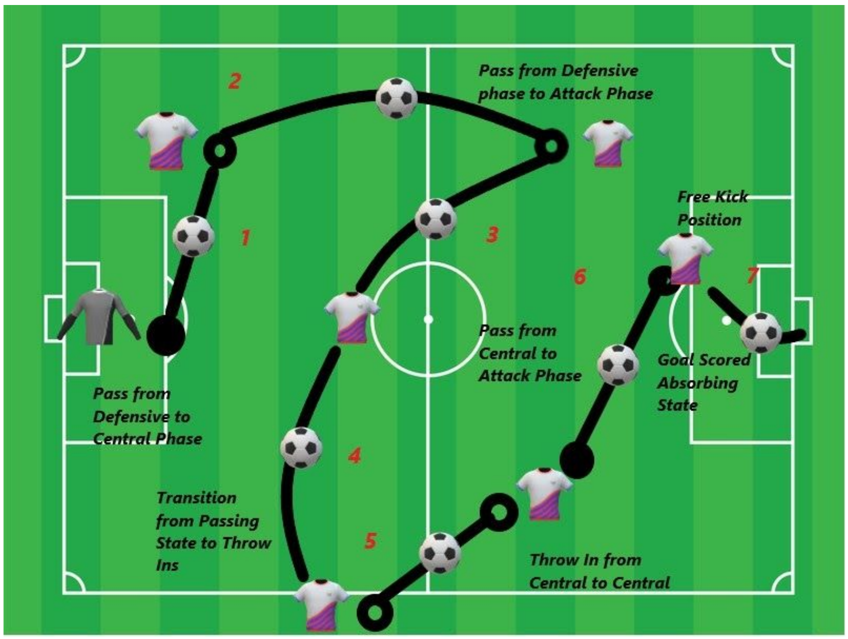
We have collected the data from this free Github Repository and from this Kaggle Dataset. The data includes the match winning odds, positional movements of players on the fields, goals scored and time, whether the game was away or home.

3. Estimating the Transition Probabilities

3.1 Counting Procedure

We divide the data by the state that preceded them and then count the different transitions, we obtain the among Nij, transition from state i to state j, then divide the total jumps out of that state i and get the transition probability as

$$P(X(t_{n+1}) = j | X(t_n) = i) = \frac{n_{ij}}{\sum_{k \in \Omega} n_{ik}}$$



Here we can see one possible Markov Chain being formed.
Defend->Central->Attack->Defend->ThrowIn->Central->Attack->Free Kick->Goal

3.2 Logit Regression

For data preprocessing we use .csv file containing information about movements of players and balls, we use the following variable to be regressed upon

$$y = \frac{e^{\beta + x_1\theta_{time} + x_2\theta_{score} + x_3\theta_{odds}}}{1 + e^{\beta + x_1\theta_{time} + x_2\theta_{score} + x_3\theta_{odds}}}$$

Due to big amount of data to be regressed and interpreted we transform our given Markov Chain such that

1. We regress all transition probabilities with our initial Logit Model that contains all of the covariates (having parameter vector Theta), we do Wald Test on every covariate to identify transitions where we can reject Ho within 90% confidence interval.
2. Then we perform the VIF-test to remove multicollinearity.

3. Further we perform the log-likelihood test and then Goodness of Fit test to interpret the described data.

3.3 Considering the Final Matrix (The Transition Matrix)

There will be many models because of consideration of one covariate over the other, these models are.

$$M1: \quad P(X(t_{n+1}) = j | X(t_n) = i) = \frac{n_{ij}}{\sum_{k\in\Omega} n_{ik}}\left(1 + \frac{1 - P_{row_i}}{P_{row_i}}\right)$$

$$M2: \quad y = \frac{e^{\beta + x_1\theta_{time} + x_2\theta_{score} + x_3\theta_{odds}}}{1 + e^{\beta + x_1\theta_{time} + x_2\theta_{score} + x_3\theta_{odds}}}\left(1 + \frac{1 - P_{row_i}}{P_{row_i}}\right)$$

M1: The Counting Model
M2: The Logit Regression with Theta Covariates

$$M3: \quad y = \frac{e^{\beta + x_1\theta_{time} + x_2\theta_{score}}}{1 + e^{\beta + x_1\theta_{time}}}\left(1 + \frac{1 - P_{row_i}}{P_{row_i}}\right)$$

$$M4: \quad y = \frac{e^{\beta + x_1\theta_{time} + x_2\theta_{odds}}}{1 + e^{\beta + x_1\theta_{score}}}\left(1 + \frac{1 - P_{row_i}}{P_{row_i}}\right)$$

$$M6: \quad y = \frac{e^{\beta + x_1\theta_{time}}}{1 + e^{\beta + x_1\theta_{time}}}\left(1 + \frac{1 - P_{row_i}}{P_{row_i}}\right)$$

$$M7: \quad y = \frac{e^{\beta + x_1\theta_{score}}}{1 + e^{\beta + x_1\theta_{score}}}\left(1 + \frac{1 - P_{row_i}}{P_{row_i}}\right)$$

$$M8: \quad y = \frac{e^{\beta + x_1\theta_{odds}}}{1 + e^{\beta + x_1\theta_{odds}}}\left(1 + \frac{1 - P_{row_i}}{P_{row_i}}\right)$$

M3: Theta Time and Theta Score are covariates
M4: Theta Time and Thera Odds are covariates
M5: Theta Score and Theta Odds
M6: Theta Time being scole variate
M7: Theta score as covariate
M8: Theta Odds as Covariate.

4. Writing the Python Code for Markov Chain and Logit Regression.

First we will try to carry out Logit Regression and then after obtaining the dependent probability relation we will construct the Markov Chain to show our results.

```
import numpy as np
import pandas as pd
import os

cwd = os.getcwd()
print(cwd)
files = os.listdir(cwd)
print(files)
```

```python
#Reading .csv files containing information about season 11-12 of EPL
all_raw_data_12 = pd.read_csv(r'C:\Users\kushk\Desktop\Projects For Semester 5\2011-12.csv')
raw_data_12 =
all_raw_data_12[['HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR',

'HS','AS','HST','AST','HF','AF','HC','AC','HY','AY','HR','AR']]
print(raw_data_12.shape)
print(raw_data_12.head(), raw_data_12.tail())
print(raw_data_12[['FTR']])

playing_stat = pd.concat([raw_data_12], ignore_index = True)
seasons = [raw_data_12]
print(playing_stat.head())
number = playing_stat.shape[0]
print(number)
#We consider the data for this season in general.

#Feature Extraction for using Logit Regression.
table = pd.DataFrame(columns = ('Team','HGS','AGS','HAS','AAS',
                                'HGC','AGC','HDS','ADS','FTAG','FTHG'))
avg_home_scored = playing_stat.FTHG.sum()/number
avg_away_scored = playing_stat.FTAG.sum()/number
avg_home_conceded = avg_away_scored
avg_away_conceded = avg_home_scored

res_home = playing_stat.groupby('HomeTeam')
res_away = playing_stat.groupby('AwayTeam')
all_teams_list = list(res_home.groups.keys())
print("All Teams List\n", all_teams_list)

table.Team = list(res_home.groups.keys())
table.HGS = res_home.FTHG.sum().values
table.HGC = res_home.FTAG.sum().values
table.AGS = res_away.FTAG.sum().values
table.AGC = res_away.FTHG.sum().values

table.HAS = (table.HGS / 19.0) / avg_home_scored
table.AAS = (table.AGS / 19.0) / avg_away_scored
table.HDS = (table.HGC / 19.0) / avg_home_conceded
table.ADS = (table.AGC / 19.0) / avg_away_conceded

feature_table = playing_stat.iloc[:,:23]
feature_table = feature_table[['HomeTeam','AwayTeam','FTR','HST','AST','HC','AC']]

#Home Attacking Strength(HAS), Home Defensive Strength(HDS), Away Attacking Strength(AAS), Away Defensive Strength(ADS)
f_HAS = []
```

```
f_HDS = []
f_AAS = []
f_ADS = []
for index,row in feature_table.iterrows():
    f_HAS.append(table[table['Team'] == row['HomeTeam']]['HAS'].values[0])
    f_HDS.append(table[table['Team'] == row['HomeTeam']]['HDS'].values[0])
    f_AAS.append(table[table['Team'] == row['AwayTeam']]['AAS'].values[0])
    f_ADS.append(table[table['Team'] == row['AwayTeam']]['ADS'].values[0])

feature_table['HAS'] = f_HAS
feature_table['HDS'] = f_HDS
feature_table['AAS'] = f_AAS
feature_table['ADS'] = f_ADS
print(feature_table)
#This data gets us the Home and Away, Attacking and Defending Strength
#Which we will use to calculate probabilities.

n_matches = len(playing_stat)
average_home_goals = sum(playing_stat['FTHG'])/n_matches
average_away_goals = sum(playing_stat['FTAG'])/n_matches
average_home_points = (3*sum(playing_stat['FTR'] == 'H') +
sum(playing_stat['FTR'] == 'D'))/n_matches
average_away_points = (3*sum(playing_stat['FTR'] == 'A') +
sum(playing_stat['FTR'] == 'D'))/n_matches
print("Aveage Home Goals",average_home_goals)
print("Average Away Goals",average_away_goals)
print("Average Home Points",average_home_points)
print("Average Away Points",average_away_points)

#Since all the parameters like
#Forward Passing, Free-Kicks impact performance
x_train_home = raw_data_12[['FTHG']]
y_train_home = raw_data_12[['FTR']]

from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(x_train_home, y_train_home)

x_test = raw_data_12[['FTAG']]
y_pred = classifier.predict(x_test)
print(y_pred.shape)
print(y_pred)


## Now representing the data as markov chain.
import matplotlib.pyplot as plt
import numpy as np
```

```
class MarkovChain():
    def __init__(self, transition_prob):
        self.transition_prob = transition_prob
        self.states = list(transition_prob.keys())

    def next_state(self, current_state):
        p = [self.transition_prob[current_state][next_state] for next_state in
self.states]
        p = np.array(p)
        p/=p.sum() #Normalise
        return np.random.choice(
            self.states, p = p , replace = False)

    def generate_states(self, current_state, number = 5):
        future_states = []
        for i in range(number):
            future_states.append(self.next_state(current_state))
        return future_states


transition_prob = {'X0':{'X0': 0.732,'X1': 0.165,'X2':0.016 ,'X3':0.029
,'X4':0.0130 ,'X5':0.004 ,'X6':0.019 ,'X7': 0.006,'X8': 0.0006,'X9':
0.0075,'X10': 0.0029},
                   'X1':{'X0': 0.308,'X1': 0.4659,'X2':0.1152 ,'X3':0.0184
,'X4':0.0143 ,'X5':0.0109 ,'X6': 0.0157,'X7':0.0072 ,'X8':0.0023 ,'X9':0.0258
,'X10': 0.0119},
                   'X2':{'X0': 0.2579,'X1': 0.2063,'X2': 0.2440,'X3':0.0203
,'X4':0.0179 ,'X5':0.0428 ,'X6':0.0242 ,'X7':0.0027 ,'X8': 0.0027,'X9':
0.0811,'X10': 0.0786},
                   'X3':{'X0': 0.7379,'X1':0.1827 ,'X2':0.0132 ,'X3':0.0183
,'X4': 0.0105,'X5':0.0039 ,'X6':0.0202 ,'X7':0.0085 ,'X8':0 ,'X9':0.0027
,'X10':0.0015},
                   'X4':{'X0': 0.2841,'X1':0.5255 ,'X2': 0.1021,'X3':0.0154
,'X4':0.0126 ,'X5':0.0147 ,'X6':0.0147 ,'X7':0.0084 ,'X8':0.0042 ,'X9':0.0140
,'X10':0.0042},
                   'X5':{'X0': 0.1118,'X1':0.3291 ,'X2':0.4284 ,'X3':
0.0136,'X4':0.0136 ,'X5':0.0240 ,'X6':0.0261 ,'X7':0.0021
,'X8':0.0073,'X9':0.0188 ,'X10':0.0219},
                   'X6':{'X0': 0.6070,'X1': 0.2279,'X2':0.0564 ,'X3':0.0244
,'X4':0.0120 ,'X5':0.0099 ,'X6':0.0259 ,'X7': 0.0049,'X8':0.0019 ,'X9':0.0209
,'X10':0.0069  },
                   'X7':{'X0':0.2541 ,'X1':0.3398 ,'X2':0.1684 ,'X3':0.0135
,'X4':0.0181 ,'X5':0.0270 ,'X6': 0.0225,'X7': 0.0060,'X8': 0,'X9':0.0962
,'X10':0.0300},
                   'X8':{'X0':0.2538 ,'X1': 0.1769,'X2':0.1461 ,'X3':0.0153
,'X4':0.0153 ,'X5':0.0461 ,'X6': 0.0153,'X7': 0,'X8': 0,'X9':0.1307
,'X10':0.1461},
```

```
                   'X9':{'X0': 0.6359,'X1': 0.2495,'X2':0.0074 ,'X3':0.0358
,'X4':0.0189 ,'X5': 0.0019,'X6': 0.0303,'X7':0.0134 ,'X8':0.0004 ,'X9':0.0044
,'X10': 0.0009},
                   'X10':{'X0': 0.2137,'X1':0.1882 ,'X2':0.2832 ,'X3':0.0229
,'X4':0.0245 ,'X5': 0.0254,'X6':0.0432 ,'X7':0.0025 ,'X8':0.0016 ,'X9':0.1153
,'X10': 0.0585}}

football_game = MarkovChain(transition_prob = transition_prob)
print("The probability of moving from state of Defending to free kick in
Attack",football_game.transition_prob['X2']['X7'])
print("The probability of moving from state of Central to Attacking Free
Kick",football_game.transition_prob['X0']['X7'])
print("The probability of moving from state of Attacking to Corner",
football_game.transition_prob['X2']['X10'])
print("The probability of moving from state of Defending Free Kick to Goal
Kick", football_game.transition_prob['X8']['X9'])
print("The probability of moving from Throw In Central to Attacking Pass",
football_game.transition_prob['X3']['X2'])
print("The next possible jump from Attacking Throw:",
football_game.next_state(current_state = 'X7'))
print("The next possible jump from Corner:",
football_game.next_state(current_state = 'X10'))
print("\nCreating Random Markov Chain, this is one instance of what Markov
Chain could look like if we start from Goal Kick\n")
print(football_game.generate_states(current_state = 'X9', number = 10))
print("\nAnother Possible Markov Chain from a Defensive Pass could be\n")
print(football_game.generate_states(current_state = 'X2', number = 10))
```

Results and Conclusions

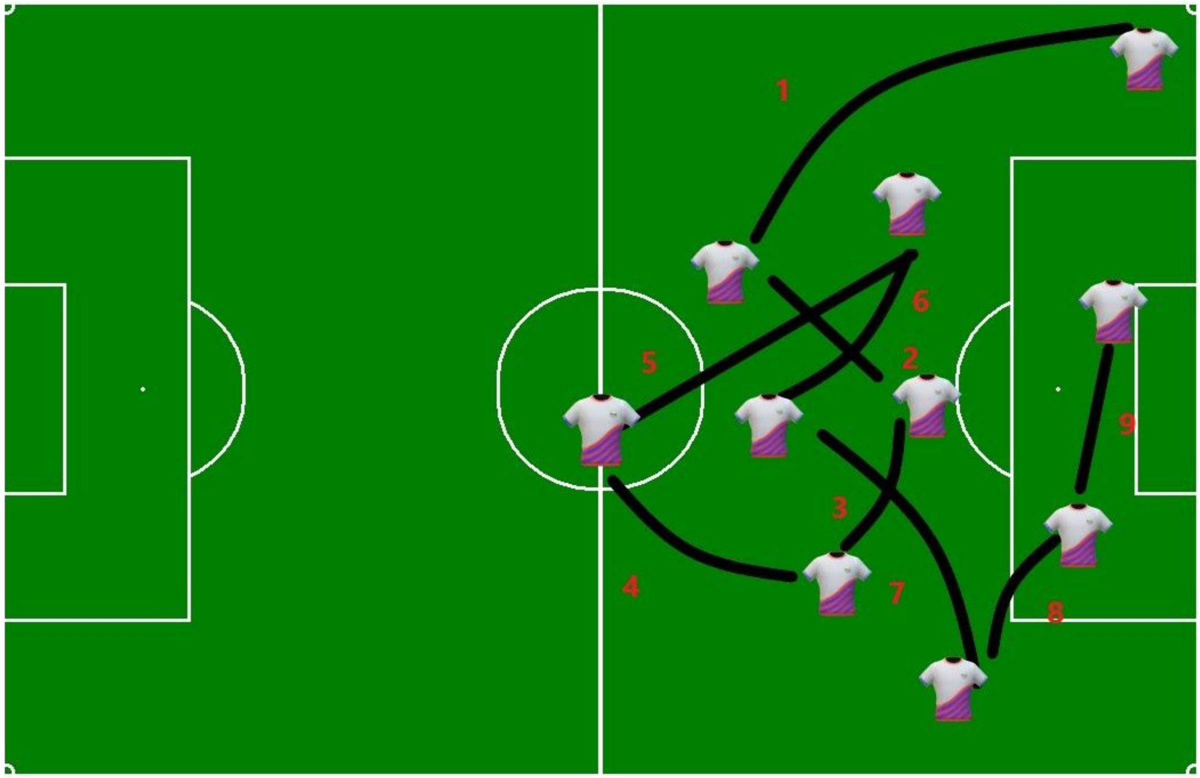
1. We have first collected the data from Github and Kaggle
2. Then we have arranged the data and extracted its important features.
3. Then we observed some of the important parameters of the dataset.
4. Then we have used Logit Regression to obtain a fitting model which is then iterated to fit perfectly by maximising the log-likelihood function.
5. Finally we have them construct a Markov Chain of the field and analyse how a game progresses.

File Edit Shell Debug Options Window Help

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win3
2
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/kushk/Desktop/Projects For Semester 5/LogitRegressionSP.py =
C:\Users\kushk\Desktop\Projects For Semester 5
['2011-12.csv', 'ff.png', 'FootballMarkov1.jpg', 'LogitRegress.py', 'LogitRegressionSP.py', '
LogitRegressMultiple.py', 'OS_E-BOOT_IEEEAccess.pdf', 'Regression_Analysis_Project.pdf', 'Reg
ression_Analysis_Project.pptx', 'SIR.py', 'SIR2.py', 'SPGuidliens.txt', 'Untitled.jpg']

The Data Shape is (380, 20)

This is how the data is given     HomeTeam     AwayTeam  FTHG  FTAG FTR  HTHG  ...  HC AC  HY
 AY  HR  AR
0  Blackburn       Wolves     1      2   A     1  ...  12  6   4   2   0   0
1     Fulham  Aston Villa     0      0   D     0  ...   2  3   2   4   0   0
2  Liverpool   Sunderland     1      1   D     1  ...   6  3   4   4   0   0
3  Newcastle      Arsenal     0      0   D     0  ...   2  5   3   5   0   1
4        QPR       Bolton     0      4   A     0  ...   3  2   1   2   1   0

[5 rows x 20 columns]         HomeTeam     AwayTeam  FTHG  FTAG FTR  HTHG  ...  HC AC  HY  AY
HR  AR
375  Sunderland  Man United     0      1   A     0  ...   1  9   3   3   0   0
376     Swansea   Liverpool     1      0   H     0  ...   6  5   1   1   0   0
377   Tottenham      Fulham     2      0   H     1  ...   9  3   0   2   0   0
378   West Brom     Arsenal     2      3   A     2  ...   9  5   0   1   0   0
379       Wigan      Wolves     3      2   H     2  ...   9  2   1   2   0   0

[5 rows x 20 columns]
```

```
File  Edit  Shell  Debug  Options  Window  Help
378    A
379    H

[380 rows x 1 columns]

Average Goals Scored Home and Away, Conceded Home and Away are 1.5894736842105264 1.215789473
6842105 1.2157894736842105 1.5894736842105264

All Teams List
 ['Arsenal', 'Aston Villa', 'Blackburn', 'Bolton', 'Chelsea', 'Everton', 'Fulham', 'Liverpool
', 'Man City', 'Man United', 'Newcastle', 'Norwich', 'QPR', 'Stoke', 'Sunderland', 'Swansea',
'Tottenham', 'West Brom', 'Wigan', 'Wolves']

This is the data about the Team Strength in variouos condition, this is one of the features o
f Logit Regression.        HomeTeam     AwayTeam FTR  HST  ...       HAS       HDS       AAS
     ADS
0      Blackburn       Wolves   A    8  ...  0.860927  1.428571  0.909091  1.291391
1         Fulham  Aston Villa   D    9  ...  1.192053  1.125541  0.735931  0.927152
2      Liverpool   Sunderland   D    4  ...  0.794702  0.692641  0.822511  0.960265
3      Newcastle      Arsenal   D    1  ...  0.960265  0.735931  1.515152  1.059603
4            QPR       Bolton   A    7  ...  0.794702  1.082251  0.995671  1.258278
..           ...          ...  ..  ...  ...       ...       ...       ...       ...
375   Sunderland   Man United   A    5  ...  0.860927  0.735931  1.601732  0.463576
376      Swansea    Liverpool   H    8  ...  0.894040  0.779221  0.995671  0.794702
377    Tottenham       Fulham   H    9  ...  1.291391  0.735931  0.519481  0.827815
378    West Brom      Arsenal   A    8  ...  0.695364  0.952381  1.515152  1.059603
379        Wigan       Wolves   H   10  ...  0.728477  1.168831  0.909091  1.291391

[380 rows x 11 columns]
For Logit Regression we are going to use The Average Odds, Forward Passing and other features
as our independent variables and we try to predict the game winning probabilities and also wh
at state occurs next from a given state for any team.
```

```
Considering for all of the matches to be played these are the probabilities of either H,A Team winning th
e Game
(380,)
['H' 'A' 'A' 'A' 'H' 'A' 'A' 'H' 'A' 'H' 'A' 'A' 'A' 'A' 'A' 'H' 'A' 'A'
 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'H' 'A' 'H' 'A' 'A' 'H' 'H' 'A' 'A' 'H' 'H'
 'A' 'A' 'A' 'A' 'H' 'H' 'A' 'A' 'H' 'H' 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'A'
 'A' 'A' 'H' 'A' 'A' 'A' 'H' 'H' 'A' 'H' 'H' 'H' 'A' 'A' 'A' 'A' 'A' 'A'
 'A' 'A' 'A' 'H' 'A' 'H' 'A' 'H' 'H' 'A' 'A' 'H' 'A' 'H' 'H' 'H' 'A' 'H'
 'A' 'A' 'H' 'H' 'A' 'H' 'H' 'A' 'H' 'A' 'H' 'A' 'A' 'A' 'A' 'H' 'A' 'H'
 'A' 'A' 'A' 'H' 'H' 'A' 'A' 'A' 'H' 'H' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'H'
 'H' 'A' 'A' 'A' 'H' 'A' 'H' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'H' 'A' 'A' 'H'
 'A' 'H' 'A' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'H' 'H' 'A' 'H' 'A' 'H' 'H' 'H'
 'A' 'H' 'A' 'H' 'H' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'H' 'A' 'A'
 'A' 'A' 'H' 'H' 'A' 'H' 'A' 'A' 'A' 'H' 'H' 'A' 'H' 'H' 'A' 'A' 'H' 'H'
 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'H' 'A' 'H' 'A' 'A' 'A' 'H' 'A' 'A' 'H'
 'A' 'H' 'H' 'H' 'A' 'A' 'A' 'A' 'H' 'H' 'H' 'A' 'A' 'A' 'A' 'A' 'A' 'H'
 'A' 'H' 'A' 'H' 'A' 'A' 'H' 'H' 'A' 'A' 'A' 'H' 'H' 'A' 'A' 'H' 'A' 'A'
 'H' 'A' 'A' 'A' 'H' 'H' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'H' 'A' 'A' 'H' 'A'
 'A' 'A' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'A' 'H' 'A' 'A' 'H' 'A' 'A' 'A' 'H'
 'A' 'A' 'A' 'A' 'H' 'A' 'A' 'A' 'H' 'H' 'A' 'H' 'A' 'A' 'H' 'A' 'A' 'H'
 'A' 'A' 'H' 'H' 'H' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'H'
 'H' 'A' 'A' 'A' 'H' 'H' 'A' 'A' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'A' 'A' 'A'
 'A' 'H' 'H' 'H' 'A' 'H' 'A' 'H' 'H' 'A' 'A' 'A' 'A' 'A' 'A' 'A' 'H' 'H'
 'H' 'A' 'H' 'A' 'A' 'H' 'A' 'A' 'A' 'A' 'A' 'A' 'H' 'A' 'H' 'A' 'A' 'A'
 'H' 'H']

Using Logit Regression and then maximising Log-Likelihood Function we have achieved the Single Step Trans
ition Matrix as follows
These are some of the specific movement probabilities

The probabililty of moving from state of Defending to free kick in Attack 0.0027
The probability of moving from state of Central to Attacking Free Kick 0.006
The probability of moving from state of Attacking to Corner 0.0786
The probability of moving from state of Defending Free Kick to Goal Kick 0.1307
The probability of moving from Throw In Central to Attacking Pass 0.0132
The next possible jumpt from Atacking Throw: X0
The next possible jumpt from Corner: X2

Creating Random Markov Chain, this is one instance of what Markov Chain could look like if we start from
Goal Kick

['X0', 'X1', 'X0', 'X1', 'X0', 'X1', 'X1', 'X0', 'X1', 'X6']

Another Possible Markov Chain from a Defensive Pass could be

['X9', 'X0', 'X10', 'X9', 'X1', 'X0', 'X2', 'X1', 'X0', 'X2']
```



This is the resulting Markov Chain after 10 discrete time intervals from Corner Kick, this is one instance of Markov chain generated.

$$\begin{bmatrix}
0.7323 & 0.1656 & 0.0176 & 0.0291 & 0.0130 & 0.0041 & 0.0196 & 0.0064 & 0.0006 & 0.0075 & 0.0029 \\
0.3088 & 0.4659 & 0.1152 & 0.0184 & 0.0143 & 0.0109 & 0.0157 & 0.0072 & 0.0023 & 0.0258 & 0.0119 \\
0.2579 & 0.2063 & 0.2440 & 0.0203 & 0.0179 & 0.0428 & 0.0242 & 0.0027 & 0.0027 & 0.0811 & 0.0786 \\
0.7379 & 0.1827 & 0.0132 & 0.0183 & 0.0105 & 0.0039 & 0.0202 & 0.0085 & 0 & 0.0027 & 0.0015 \\
0.2841 & 0.5255 & 0.1021 & 0.0154 & 0.0126 & 0.0147 & 0.0147 & 0.0084 & 0.0042 & 0.0140 & 0.0042 \\
0.1118 & 0.3291 & 0.4284 & 0.0136 & 0.0136 & 0.0240 & 0.0261 & 0.0021 & 0.0073 & 0.0188 & 0.0219 \\
0.6070 & 0.2279 & 0.0564 & 0.0244 & 0.0120 & 0.0099 & 0.0259 & 0.0049 & 0.0019 & 0.0209 & 0.0069 \\
0.2541 & 0.3398 & 0.1684 & 0.0135 & 0.0180 & 0.0270 & 0.0225 & 0.0060 & 0 & 0.0962 & 0.0300 \\
0.2538 & 0.1769 & 0.1461 & 0.0153 & 0.0153 & 0.0461 & 0.0153 & 0 & 0 & 0.1307 & 0.1461 \\
0.6359 & 0.2495 & 0.0074 & 0.0358 & 0.0189 & 0.0019 & 0.0303 & 0.0134 & 0.0004 & 0.0044 & 0.0009 \\
0.2137 & 0.1882 & 0.2832 & 0.0229 & 0.0245 & 0.0254 & 0.0432 & 0.0025 & 0.0016 & 0.1153 & 0.0585
\end{bmatrix}$$

This is the resulting Transition Matrix (Single Step)

This is the Log-Likelihood Ratios.

Trans.	p-value	Trans.	p-value
$p_{p_s p_s}$	2.280811e-24	$p_{p_s p_a}$	1.351248e-11
$p_{p_s p_d}$	1.017026e-04	$p_{p_s t_s}$	1.023651e-07
$p_{p_s t_a}$	8.217796e-06	$p_{p_s f_s}$	1.133169e-05
$p_{p_s g}$	**5.736316e-02**	$p_{p_s c}$	1.503139e-02
$p_{p_a p_s}$	1.008443e-07	$p_{p_a p_a}$	5.180665e-04
$p_{p_a p_d}$	7.926483e-03	$p_{p_a f_a}$	1.393922e-02
$p_{p_a g}$	6.395749e-03	$p_{p_d p_d}$	**8.59601e-02**
$p_{p_d f_a}$	7.163972e-03	$p_{t_s p_s}$	1.067479e-02

Trans.	p-value	Trans.	p-value
$p_{t_s p_a}$	2.754313e-02	$p_{t_s g}$	2.422576e-02
$p_{t_a p_d}$	1.483587e-02	$p_{t_d p_d}$	4.727346e-02
$p_{t_d f_a}$	2.288513e-02	$p_{f_s p_s}$	1.283991e-03
$p_{f_s p_d}$	8.894682e-04	$p_{f_s t_s}$	2.290607e-02
$p_{f_s g}$	2.620068e-03	$p_{f_a p_a}$	3.347832e-03
$p_{f_a p_d}$	1.867370e-03	$p_{f_d t_d}$	**7.72366e-02**
$p_{g p_s}$	9.398821e-03	$p_{g p_a}$	3.886137e-05
p_{gc}	**5.85898e-02**	$p_{c t_a}$	3.243490e-02

Goodness of Fit Test Pseudo R2 values

All values are under 1% confidence interval, hence the logistic regression model that we have used is predicting the Markov Chain Correctly.

The Theta(odds) Factor

These are the weights of corresponding factors in Logit Line

Trans.	$\theta odds$-Estimate
$p_{p_s p_d}$	-0.01992
$p_{t_a p_d}$	-0.05956
$p_{t_d p_d}$	0.03276
$p_{f_s p_d}$	-0.09389
$p_{f_a p_d}$	-0.09242
$p_{f_d t_d}$	0.13822

<u>Conclusion</u>:-

We have used the Logit Regression model to fit data of set pieces in Football to predict the chain (Markov) of events that can happen in a game, and constructed Markov Chain from it. The state transition matrix we obtained gives us the one step transition probabilities of movement from one state to another.

Reference:-

1. Kaggle Dataset on Regression Analysis
2. Premier League
3. Goodness of Fit Test
4. Logistic Regression for Betting
5. Output of NFL game Thesis
6. Predictive analysis and modelling football results using Machine learning approach for English Premier League Research Paper
7. Machine learning in men's professional football Research Paper
8. A Markovian model for association football possession and its Outcomes.
9. Learning State Representation and Markov Models in Football Analytics