1.3 Data Processing

The 20 CSV file were imported as individual data frames. The features can be categorized in 3 broad sub-categories details such as the Result of the match, Match Statistics and Betting Statistics etc. This includes the number of Home Team goals, Away Team goals, Match date, Half time goals, Match Referee, Home/Away Team Shots, Home/Away Team Shots on Targets, Home/Away Team Corners, Fouls Committed, Yellow and Red cards, Bet365 home win odds, Bet365 draw odds, Bet365 away win odds, Blue Square home win odds, Bet&Win away win odds etc.

Firstly, the dimensions of all the CSV files were checked and it was noticed that there were irregularities in the feature vectors of each data frame. Hence, it was observed that not all columns will be used for the training data. Once that was done, the files were checked for missing/null values and no missing values were observed for any particular column. We had noticed that the first 9 years did not include the betting statistics and hence the features were not consistent. Hence, for now, we have taken it as a standard to not use betting statistics as a feature in our prediction. In conclusion, 21 columns were picked out which included 20 features and one predicted label. Finally, we had concatenated all data frames which included 7640 data points (labels and features).

1.4 Feature Engineering

Firsty, it was important to convert the predicted variables which were denoted as 'Home', 'Away' and 'Draw'. Therefore, for classification to take place, it was important to encode these labels for predicated variable that were run for the entire dataset. The labels were denoted as 0,1,2 for Home Win, Away Win and Draw. Next, we plotted a correlation matrix to see the correlation of these features with the match result decision (FTR) . The features with high positive or negative correlation with FTR were selected and out of these features, the ones which were correlated with each other were dropped. The features that were selected as follows: HomeTeam, AwayTeam, HS, HST, AR, HC, AF, AY. We tried getting the final goal difference and the Average Home and Away goals scored but these will be implemented later due to dimensionality reduction and they were not included in the feature space. We had also separately integer-encoded the Home and Away teams which featured a total of 24 labels. This is to see whether the name/ Prescence of a certain club had influence on the predicted result

1.5 Implementation

Firstly, the Integer-encoded FTR labels were stored as predicted variables that were later divided using the Train-test split function using sci-kit learn. The train test split was divided such that we had kept 1000 test data points out of 7640 data points for cross validation. Finally, the classifiers we had used were KNN, Naïve Bayes and SVM which were implemented using Sci-kit libraries.

1.6 Performance Evaluation

In most of the research works, the accuracy of the prediction model has been taken as one of the common performance metrics while working on a model. Typically, the performance of the machine learning prediction algorithms measured by using some metrics based on the classification algorithm. The results were divided based on the inclusion or exclusion of Home Team and Away team Integer labels. The results were as follows:

a) With Home and Away Team Integer-labels:

   1) KNN- Accuracy: 0.384, Training Accuracy: 0.7375)

   2) Naïve Bayes Class: (0.464, 0.4733433734939759)

   3) SVM- Accuracy: 50.2%

b) Without Home and Away Team Integer-labels:

   1) KNN- (0.387, 0.7245481927710843)

   2) Naïve Bayes- (0.456, 0.4724397590361446)

   3) SVM- Accuracy: 51.300000000000004%

CODE

In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
```

In [4]:
```
y1 = pd.read_csv('2000-01.csv')
y2 = pd.read_csv('2001-02.csv')
y3 = pd.read_csv('2002-03.csv')
y4 = pd.read_csv('2003-04.csv')
y5 = pd.read_csv('2004-05.csv')
y6 = pd.read_csv('2005-06.csv')
y7 = pd.read_csv('2006-07.csv')
y8 = pd.read_csv('2007-08.csv')
y9 = pd.read_csv('2008-09.csv')
y10 = pd.read_csv('2009-10.csv')
y11 = pd.read_csv('2010-11.csv')
y12 = pd.read_csv('2011-12.csv')
y13 = pd.read_csv('2012-13.csv')
y14 = pd.read_csv('2013-14.csv')
y15 = pd.read_csv('2014-15.csv')
y16 = pd.read_csv('2015-16.csv')
y17 = pd.read_csv('2016-17.csv')
y18 = pd.read_csv('2017-18.csv')
y19 = pd.read_csv('2018-19.csv')
y20 = pd.read_csv('2019-20.csv')
```

In [5]:
```
y17
```

Out[5]:

	Div	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	...	BbAv<2.5	BbAH	BbAHh	BbMxAHH	BbAvAHH	BbMxAHA	BbAv
0	E0	13/08/16	Burnley	Swansea	0	1	A	0	0	D	...	1.61	32	-0.25	2.13	2.06	1.86	
1	E0	13/08/16	Crystal Palace	West Brom	0	1	A	0	0	D	...	1.52	33	-0.50	2.07	2.00	1.90	
2	E0	13/08/16	Everton	Tottenham	1	1	D	1	0	H	...	1.77	32	0.25	1.91	1.85	2.09	
3	E0	13/08/16	Hull	Leicester	2	1	H	1	0	H	...	1.67	31	0.25	2.35	2.26	2.03	
4	E0	13/08/16	Man City	Sunderland	2	1	H	1	0	H	...	2.48	34	-1.50	1.81	1.73	2.20	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
375	E0	21/05/17	Liverpool	Middlesbrough	3	0	H	1	0	H	...	2.73	18	-2.50	2.02	1.97	1.95	
376	E0	21/05/17	Man United	Crystal Palace	2	0	H	2	0	H	...	1.81	19	-0.25	2.19	2.11	1.85	
377	E0	21/05/17	Southampton	Stoke	0	1	A	0	0	D	...	2.01	18	-0.75	2.03	1.98	1.93	
378	E0	21/05/17	Swansea	West Brom	2	1	H	0	1	A	...	1.98	19	-0.50	2.11	2.06	1.86	
379	E0	21/05/17	Watford	Man City	0	5	A	0	4	A	...	2.80	18	1.75	1.99	1.94	2.00	

380 rows × 65 columns

```
raw_data_1 = y1[['Date', 'HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC'
print(raw_data_1.shape)
raw_data_2 = y2[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_2.shape)
raw_data_3 = y3[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_3.shape)
raw_data_4 = y4[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_4.shape)
raw_data_5 = y5[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_5.shape)
raw_data_6 = y6[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_6.shape)
raw_data_7 = y7[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_7.shape)
raw_data_8 = y8[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_8.shape)
raw_data_9 = y9[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_9.shape)
raw_data_10 = y10[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_10.shape)
raw_data_11 = y11[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_11.shape)
raw_data_12 = y12[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_12.shape)
raw_data_13 = y13[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_13.shape)
raw_data_14 = y14[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_14.shape)
raw_data_15 = y15[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_15.shape)
raw_data_16 = y16[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_16.shape)
raw_data_17 = y17[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_17.shape)
```

```
raw_data_18 = y18[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_18.shape)
raw_data_19 = y19[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_19.shape)
raw_data_20 = y20[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_20.shape)

#extracting raw data
```

```
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(160, 21)
(260, 21)
```

```
arr = [y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12,y13,y14,y15,y16,y17,y18,y19,y20]

for i in range(len(arr)):
    print(arr[i].shape)

for i in range(len(arr)):
    print("\n",i)
    print(arr[i].columns.values)
```

```
(380, 28)
(380, 28)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 71)
(380, 71)
(380, 71)
(380, 71)
(380, 65)
(380, 65)
(380, 62)
(380, 65)
(380, 65)
(160, 62)
(260, 106)
```

```
arr = [y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12,y13,y14,y15,y16,y17,y18,y19,y20]
a = []
for i in range (20):
    a.append(arr[i][['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','A

print(a)
```

```
[         Date       HomeTeam       AwayTeam  FTHG  FTAG FTR  HTHG  HTAG HTR  \
0    19/08/00       Charlton       Man City     4     0   H     2     0   H
1    19/08/00        Chelsea       West Ham     4     2   H     1     0   H
2    19/08/00       Coventry  Middlesbrough     1     3   A     1     1   D
3    19/08/00          Derby    Southampton     2     2   D     1     2   A
4    19/08/00          Leeds        Everton     2     0   H     2     0   H
..        ...            ...            ...   ...   ...  ..   ...   ...  ..
375  19/05/01       Man City        Chelsea     1     2   A     1     1   D
376  19/05/01  Middlesbrough       West Ham     2     1   H     2     1   H
377  19/05/01      Newcastle    Aston Villa     3     0   H     2     0   H
378  19/05/01    Southampton        Arsenal     3     2   H     0     1   A
379  19/05/01      Tottenham     Man United     3     1   H     1     1   D

     HS  ...  HST  AST  HF  AF  HC  AC  HY  AY  HR  AR
0    17  ...   14    4  13  12   6   6   1   2   0   0
1    17  ...   10    5  19  14   7   7   1   2   0   0
2     6  ...    3    9  15  21   8   4   5   3   1   0
3     6  ...    4    6  11  13   5   8   1   1   0   0
4    17  ...    8    6  21  20   6   4   1   3   0   0
```

```
playing_stat = a[0]
for j in range(0,20):
    playing_stat = pd.concat([playing_stat,a[j]],ignore_index=True)

playing_stat
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	Charlton	Man City	4	0	H	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	Chelsea	West Ham	4	2	H	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	Coventry	Middlesbrough	1	3	A	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	Derby	Southampton	2	2	D	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	Leeds	Everton	2	0	H	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	Norwich	Liverpool	0	1	A	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	Aston Villa	Tottenham	2	3	A	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	Arsenal	Newcastle	4	0	H	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	Chelsea	Man United	0	2	A	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	Man City	West Ham	2	0	H	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(playing_stat['FTR'])
print(integer_encoded)

playing_stat['FTR']= integer_encoded
playing_stat
```

```
[2 2 0 ... 2 0 2]
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	Charlton	Man City	4	0	2	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	Chelsea	West Ham	4	2	2	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	Coventry	Middlesbrough	1	3	0	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	Derby	Southampton	2	2	1	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	Leeds	Everton	2	0	2	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	Norwich	Liverpool	0	1	0	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	Aston Villa	Tottenham	2	3	0	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	Arsenal	Newcastle	4	0	2	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	Chelsea	Man United	0	2	0	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	Man City	West Ham	2	0	2	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(playing_stat['HomeTeam'])
print(integer_encoded)



playing_stat['HomeTeam']= integer_encoded

playing_stat
```

```
[11 12 13 ...  0 12 24]
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	11	Man City	4	0	2	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	12	West Ham	4	2	2	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	13	Middlesbrough	1	3	0	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	15	Southampton	2	2	1	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	21	Everton	2	0	2	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	29	Liverpool	0	1	0	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	1	Tottenham	2	3	0	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	0	Newcastle	4	0	2	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	12	Man United	0	2	0	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	24	West Ham	2	0	2	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
x22=pd.DataFrame(playing_stat['HomeTeam'])
x22['labels']=integer_encoded

x22['AwayTeam']=playing_stat['AwayTeam']
x22.head(20)
```

	HomeTeam	labels	AwayTeam
0	11	11	Man City
1	12	12	West Ham
2	13	13	Middlesbrough
3	15	15	Southampton
4	21	21	Everton
5	22	22	Aston Villa
6	23	23	Bradford
7	36	36	Arsenal
8	38	38	Ipswich
9	25	25	Newcastle
10	0	0	Liverpool
11	7	7	Chelsea
12	20	20	Man United
13	27	27	Tottenham
14	16	16	Charlton
15	24	24	Sunderland
16	28	28	Derby
17	34	34	Coventry

```
integer_encoded1 = label_encoder.fit_transform(playing_stat['AwayTeam'])
x22['label2']=integer_encoded1
x22

#THIS IS NOT USED IN CODE: THIS IS TO SHOW HOW ENCODING IS DONE
```

	HomeTeam	labels	AwayTeam	label2
0	11	11	Man City	24
1	12	12	West Ham	41
2	13	13	Middlesbrough	27
3	15	15	Southampton	34
4	21	21	Everton	16
...	...	...	...	...
7635	29	29	Liverpool	23
7636	1	1	Tottenham	38
7637	0	0	Newcastle	28
7638	12	12	Man United	25
7639	24	24	West Ham	41

7640 rows × 4 columns

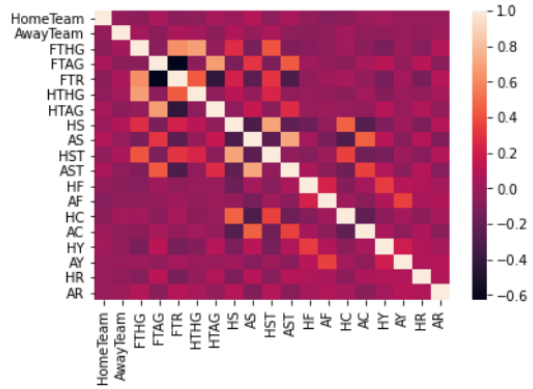
```
integer_encoded = label_encoder.fit_transform(playing_stat['AwayTeam'])
playing_stat['AwayTeam']= integer_encoded
playing_stat
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	11	24	4	0	2	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	12	41	4	2	2	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	13	27	1	3	0	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	15	34	2	2	1	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	21	16	2	0	2	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	29	23	0	1	0	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	1	38	2	3	0	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	0	28	4	0	2	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	12	25	0	2	0	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	24	41	2	0	2	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
sns.heatmap(playing_stat.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x19a59e49850>
```




```
y=playing_stat['FTR']
x=playing_stat[['HomeTeam','AwayTeam','HS', 'HST', 'AR', 'HC', 'AF','AY']]

x
```

	HomeTeam	AwayTeam	HS	HST	AR	HC	AF	AY
0	11	24	17	14	0	6	12	2
1	12	41	17	10	0	7	14	2
2	13	27	6	3	0	8	21	3
3	15	34	6	4	0	5	13	1
4	21	16	17	8	0	6	20	3
...	...	...	...	...	...	...	...	...

```
y=playing_stat['FTR']
x=playing_stat[['HomeTeam','AwayTeam','HS', 'HST', 'AR', 'HC', 'AF','AY']]

x
```

	HomeTeam	AwayTeam	HS	HST	AR	HC	AF	AY
0	11	24	17	14	0	6	12	2
1	12	41	17	10	0	7	14	2
2	13	27	6	3	0	8	21	3
3	15	34	6	4	0	5	13	1
4	21	16	17	8	0	6	20	3
...	...	...	...	...	...	...	...	...
7635	29	23	5	1	0	2	11	2
7636	1	38	18	4	0	12	10	0
7637	0	28	15	7	0	5	9	0
7638	12	25	17	1	0	9	11	3
7639	24	41	20	7	0	6	7	1

7640 rows × 8 columns

```
import sklearn
from sklearn.model_selection import train_test_split
# Shuffle and split the dataset into training and testing set.
train_X, test_X, train_Y, test_Y = train_test_split(x, y,
                                                    test_size = 1000,
                                                    random_state = 2,
                                                    stratify = y)

print(test_X)
```

```
      HomeTeam  AwayTeam  HS  HST  AR  HC  AF  AY
4700         5         1  17   12   0   6   7   2
846         17        20   7    3   0   3  17   3
191          7        36  11    3   0   9  17   0
4334        24         5  19   10   0   8  18   2
3452        24        12  13    7   1   8  10   1
...        ...       ...  ..  ...  ..  ..  ..  ..
4156        12        35  29   18   0   7  12   3
4099         3        12   8    4   0   3   9   1
3222        16         5  18   10   0   7  18   3
5445        41        17  23   12   0   6   8   2
6153        14        40  21    8   0   8  13   3

[1000 rows x 8 columns]
```

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics

def KNN_classifier(train_X,train_Y,test_X,test_Y):

    neigh = KNeighborsClassifier(n_neighbors=2)
    neigh.fit(train_X, train_Y)
    trainAccuracy = neigh.score(train_X,train_Y)
    y_pred = neigh.predict(test_X)
    accuracy = metrics.accuracy_score(test_Y,y_pred)
    return accuracy,trainAccuracy



def NaiveBayes_classifier(train_X,train_Y,test_X,test_Y):

    gnb = GaussianNB()
    gnb.fit(train_X, train_Y)
    trainAccuracy = gnb.score(train_X,train_Y)
    y_pred = gnb.predict(test_X)
    accuracy = metrics.accuracy_score(test_Y,y_pred)
    return accuracy,trainAccuracy

import sys, os
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.model_selection import train_test_split, GridSearchCV

def SVM(train_X,train_Y,test_X,test_Y):
    clf = svm.SVC(kernel='linear')
    clf.fit(train_X, train_Y)
    # Plot decision function on training and test data
    #plot_decision_function(train_X, train_Y, test_X, test_Y, clf)
    clf_predictions = clf.predict(test_X)
    print("Accuracy: {}%".format(clf.score(test_X, test_Y) * 100 ))
```

```
KNN_classifier(train_X,train_Y,test_X,test_Y)
```

```
(0.384, 0.7375)
```

```
from sklearn.naive_bayes import GaussianNB

NaiveBayes_classifier(train_X,train_Y,test_X,test_Y)
```

```
(0.464, 0.4733433734939759)
```

```
SVM(train_X,train_Y,test_X,test_Y)
```

```
Accuracy: 50.2%
```

```
x1= playing_stat[['HS', 'HST', 'AR', 'HC', 'AF','AY']]
import sklearn
from sklearn.model_selection import train_test_split
# Shuffle and split the dataset into training and testing set.
train_X, test_X, train_Y, test_Y = train_test_split(x1, y,
                                                    test_size = 1000,
                                                    random_state = 2,
                                                    stratify = y)
print(KNN_classifier(train_X,train_Y,test_X,test_Y))
print(NaiveBayes_classifier(train_X,train_Y,test_X,test_Y))
print(SVM(train_X,train_Y,test_X,test_Y))
```

```
(0.387, 0.7245481927710843)
(0.456, 0.4724397590361446)
Accuracy: 51.300000000000004%
None
```

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics

def KNN_classifier(train_X,train_Y,test_X,test_Y):

    neigh = KNeighborsClassifier(n_neighbors=2)
    neigh.fit(train_X, train_Y)
    trainAccuracy = neigh.score(train_X,train_Y)
    y_pred = neigh.predict(test_X)
    accuracy = metrics.accuracy_score(test_Y,y_pred)
    return accuracy,trainAccuracy



def NaiveBayes_classifier(train_X,train_Y,test_X,test_Y):

    gnb = GaussianNB()
    gnb.fit(train_X, train_Y)
    trainAccuracy = gnb.score(train_X,train_Y)
    y_pred = gnb.predict(test_X)
    accuracy = metrics.accuracy_score(test_Y,y_pred)
    return accuracy,trainAccuracy

import sys, os
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.model_selection import train_test_split, GridSearchCV

def SVM(train_X,train_Y,test_X,test_Y):
    clf = svm.SVC(kernel='linear')
    clf.fit(train_X, train_Y)
    # Plot decision function on training and test data
    #plot_decision_function(train_X, train_Y, test_X, test_Y, clf)
    clf_predictions = clf.predict(test_X)
    print("Accuracy: {}%".format(clf.score(test_X, test_Y) * 100 ))
```

```
import sklearn
from sklearn.model_selection import train_test_split
# Shuffle and split the dataset into training and testing set.
train_X, test_X, train_Y, test_Y = train_test_split(x, y,
                                                    test_size = 1000,
                                                    random_state = 2,
                                                    stratify = y)

print(test_X)
```

```
      HomeTeam  AwayTeam  HS  HST  AR  HC  AF  AY
4700         5         1  17   12   0   6   7   2
846         17        20   7    3   0   3  17   3
191          7        36  11    3   0   9  17   0
4334        24         5  19   10   0   8  18   2
3452        24        12  13    7   1   8  10   1
...        ...       ...  ..  ...  ..  ..  ..  ..
4156        12        35  29   18   0   7  12   3
4099         3        12   8    4   0   3   9   1
3222        16         5  18   10   0   7  18   3
5445        41        17  23   12   0   6   8   2
6153        14        40  21    8   0   8  13   3

[1000 rows x 8 columns]
```

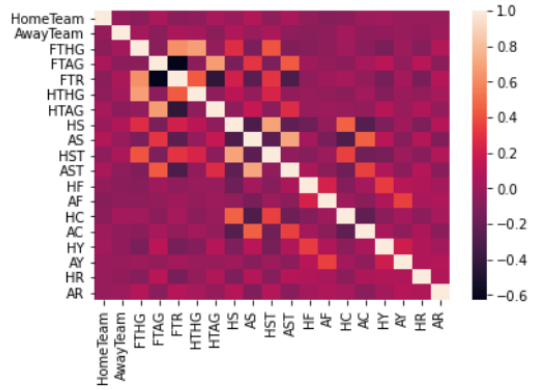
```
y=playing_stat['FTR']
x=playing_stat[['HomeTeam','AwayTeam','HS', 'HST', 'AR', 'HC', 'AF','AY']]

x
```

	HomeTeam	AwayTeam	HS	HST	AR	HC	AF	AY
0	11	24	17	14	0	6	12	2
1	12	41	17	10	0	7	14	2
2	13	27	6	3	0	8	21	3
3	15	34	6	4	0	5	13	1
4	21	16	17	8	0	6	20	3
...	...	...	...	...	...	...	...	...
7635	29	23	5	1	0	2	11	2
7636	1	38	18	4	0	12	10	0
7637	0	28	15	7	0	5	9	0
7638	12	25	17	1	0	9	11	3
7639	24	41	20	7	0	6	7	1

7640 rows × 8 columns

```
sns.heatmap(playing_stat.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x19a59e49850>
```




```
y=playing_stat['FTR']
x=playing_stat[['HomeTeam','AwayTeam','HS', 'HST', 'AR', 'HC', 'AF','AY']]

x
```

	HomeTeam	AwayTeam	HS	HST	AR	HC	AF	AY
0	11	24	17	14	0	6	12	2
1	12	41	17	10	0	7	14	2
2	13	27	6	3	0	8	21	3
3	15	34	6	4	0	5	13	1
4	21	16	17	8	0	6	20	3
...	...	...	...	...	...	...	...	...

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	11	24	4	0	2	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	12	41	4	2	2	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	13	27	1	3	0	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	15	34	2	2	1	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	21	16	2	0	2	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	29	23	0	1	0	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	1	38	2	3	0	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	0	28	4	0	2	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	12	25	0	2	0	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	24	41	2	0	2	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
integer_encoded1 = label_encoder.fit_transform(playing_stat['AwayTeam'])
x22['label2']=integer_encoded1
x22

#THIS IS NOT USED IN CODE: THIS IS TO SHOW HOW ENCODING IS DONE
```

	HomeTeam	labels	AwayTeam	label2
0	11	11	Man City	24
1	12	12	West Ham	41
2	13	13	Middlesbrough	27
3	15	15	Southampton	34
4	21	21	Everton	16
...	...	...	...	...
7635	29	29	Liverpool	23
7636	1	1	Tottenham	38
7637	0	0	Newcastle	28
7638	12	12	Man United	25
7639	24	24	West Ham	41

7640 rows × 4 columns

```
integer_encoded = label_encoder.fit_transform(playing_stat['AwayTeam'])
playing_stat['AwayTeam']= integer_encoded
playing_stat
```

```
x22=pd.DataFrame(playing_stat['HomeTeam'])
x22['labels']=integer_encoded

x22['AwayTeam']=playing_stat['AwayTeam']
x22.head(20)
```

	HomeTeam	labels	AwayTeam
0	11	11	Man City
1	12	12	West Ham
2	13	13	Middlesbrough
3	15	15	Southampton
4	21	21	Everton
5	22	22	Aston Villa
6	23	23	Bradford
7	36	36	Arsenal
8	38	38	Ipswich
9	25	25	Newcastle
10	0	0	Liverpool
11	7	7	Chelsea
12	20	20	Man United
13	27	27	Tottenham
14	16	16	Charlton
15	24	24	Sunderland
16	28	28	Derby
17	34	34	Coventry

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(playing_stat['HomeTeam'])
print(integer_encoded)



playing_stat['HomeTeam']= integer_encoded

playing_stat
```

```
[11 12 13 ...  0 12 24]
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	11	Man City	4	0	2	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	12	West Ham	4	2	2	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	13	Middlesbrough	1	3	0	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	15	Southampton	2	2	1	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	21	Everton	2	0	2	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	29	Liverpool	0	1	0	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	1	Tottenham	2	3	0	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	0	Newcastle	4	0	2	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	12	Man United	0	2	0	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	24	West Ham	2	0	2	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(playing_stat['FTR'])
print(integer_encoded)

playing_stat['FTR']= integer_encoded
playing_stat
```

```
[2 2 0 ... 2 0 2]
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	Charlton	Man City	4	0	2	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	Chelsea	West Ham	4	2	2	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	Coventry	Middlesbrough	1	3	0	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	Derby	Southampton	2	2	1	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	Leeds	Everton	2	0	2	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	Norwich	Liverpool	0	1	0	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	Aston Villa	Tottenham	2	3	0	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	Arsenal	Newcastle	4	0	2	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	Chelsea	Man United	0	2	0	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	Man City	West Ham	2	0	2	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
playing_stat = a[0]
for j in range(0,20):
    playing_stat = pd.concat([playing_stat,a[j]],ignore_index=True)

playing_stat
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	HS	...	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
0	19/08/00	Charlton	Man City	4	0	H	2	0	H	17	...	14	4	13	12	6	6	1	2	0	0
1	19/08/00	Chelsea	West Ham	4	2	H	1	0	H	17	...	10	5	19	14	7	7	1	2	0	0
2	19/08/00	Coventry	Middlesbrough	1	3	A	1	1	D	6	...	3	9	15	21	8	4	5	3	1	0
3	19/08/00	Derby	Southampton	2	2	D	1	2	A	6	...	4	6	11	13	5	8	1	1	0	0
4	19/08/00	Leeds	Everton	2	0	H	2	0	H	17	...	8	6	21	20	6	4	1	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7635	15/02/2020	Norwich	Liverpool	0	1	A	0	0	D	5	...	1	6	5	11	2	7	1	2	0	0
7636	16/02/2020	Aston Villa	Tottenham	2	3	A	1	2	A	18	...	4	10	12	10	12	7	2	0	0	0
7637	16/02/2020	Arsenal	Newcastle	4	0	H	0	0	D	15	...	7	2	15	9	5	6	2	0	0	0
7638	17/02/2020	Chelsea	Man United	0	2	A	0	1	A	17	...	1	3	11	11	9	8	4	3	0	0
7639	19/02/2020	Man City	West Ham	2	0	H	1	0	H	20	...	7	0	5	7	6	1	0	1	0	0

7640 rows × 21 columns

```
[         Date       HomeTeam       AwayTeam  FTHG  FTAG FTR  HTHG  HTAG HTR  \
0    19/08/00       Charlton       Man City     4     0   H     2     0   H
1    19/08/00        Chelsea       West Ham     4     2   H     1     0   H
2    19/08/00       Coventry  Middlesbrough     1     3   A     1     1   D
3    19/08/00          Derby    Southampton     2     2   D     1     2   A
4    19/08/00          Leeds        Everton     2     0   H     2     0   H
..        ...            ...            ...   ...   ... ..   ...   ... ..
375  19/05/01       Man City        Chelsea     1     2   A     1     1   D
376  19/05/01  Middlesbrough       West Ham     2     1   H     2     1   H
377  19/05/01      Newcastle    Aston Villa     3     0   H     2     0   H
378  19/05/01    Southampton        Arsenal     3     2   H     0     1   A
379  19/05/01      Tottenham     Man United     3     1   H     1     1   D

     HS  ...  HST  AST  HF  AF  HC  AC  HY  AY  HR  AR
0    17  ...   14    4  13  12   6   6   1   2   0   0
1    17  ...   10    5  19  14   7   7   1   2   0   0
2     6  ...    3    9  15  21   8   4   5   3   1   0
3     6  ...    4    6  11  13   5   8   1   1   0   0
4    17  ...    8    6  21  20   6   4   1   3   0   0
```

```
arr = [y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12,y13,y14,y15,y16,y17,y18,y19,y20]

for i in range(len(arr)):
    print(arr[i].shape)

for i in range(len(arr)):
    print("\n",i)
    print(arr[i].columns.values)
```

```
(380, 28)
(380, 28)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 23)
(380, 71)
(380, 71)
(380, 71)
(380, 71)
(380, 65)
(380, 65)
(380, 62)
(380, 65)
(380, 65)
(160, 62)
(260, 106)
```

```
arr = [y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12,y13,y14,y15,y16,y17,y18,y19,y20]
a = []
for i in range (20):
    a.append(arr[i][['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','A

print(a)
```

```
raw_data_18 = y18[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_18.shape)
raw_data_19 = y19[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_19.shape)
raw_data_20 = y20[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_20.shape)

#extracting raw data
```

```
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(380, 21)
(160, 21)
(260, 21)
```

```
raw_data_1 = y1[['Date', 'HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_1.shape)
raw_data_2 = y2[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_2.shape)
raw_data_3 = y3[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_3.shape)
raw_data_4 = y4[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_4.shape)
raw_data_5 = y5[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_5.shape)
raw_data_6 = y6[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_6.shape)
raw_data_7 = y7[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_7.shape)
raw_data_8 = y8[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_8.shape)
raw_data_9 = y9[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC',
print(raw_data_9.shape)
raw_data_10 = y10[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_10.shape)
raw_data_11 = y11[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_11.shape)
raw_data_12 = y12[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_12.shape)
raw_data_13 = y13[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_13.shape)
raw_data_14 = y14[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_14.shape)
raw_data_15 = y15[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_15.shape)
raw_data_16 = y16[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_16.shape)
raw_data_17 = y17[['Date','HomeTeam','AwayTeam','FTHG','FTAG','FTR','HTHG','HTAG','HTR','HS','AS','HST','AST','HF','AF','HC','AC
print(raw_data_17.shape)
```

In [5]:
```
y17
```

Out[5]:

	Div	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	...	BbAv<2.5	BbAH	BbAHh	BbMxAHH	BbAvAHH	BbMxAHA	BbAv
0	E0	13/08/16	Burnley	Swansea	0	1	A	0	0	D	...	1.61	32	-0.25	2.13	2.06	1.86	
1	E0	13/08/16	Crystal Palace	West Brom	0	1	A	0	0	D	...	1.52	33	-0.50	2.07	2.00	1.90	
2	E0	13/08/16	Everton	Tottenham	1	1	D	1	0	H	...	1.77	32	0.25	1.91	1.85	2.09	
3	E0	13/08/16	Hull	Leicester	2	1	H	1	0	H	...	1.67	31	0.25	2.35	2.26	2.03	
4	E0	13/08/16	Man City	Sunderland	2	1	H	1	0	H	...	2.48	34	-1.50	1.81	1.73	2.20	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
375	E0	21/05/17	Liverpool	Middlesbrough	3	0	H	1	0	H	...	2.73	18	-2.50	2.02	1.97	1.95	
376	E0	21/05/17	Man United	Crystal Palace	2	0	H	2	0	H	...	1.81	19	-0.25	2.19	2.11	1.85	
377	E0	21/05/17	Southampton	Stoke	0	1	A	0	0	D	...	2.01	18	-0.75	2.03	1.98	1.93	
378	E0	21/05/17	Swansea	West Brom	2	1	H	0	1	A	...	1.98	19	-0.50	2.11	2.06	1.86	
379	E0	21/05/17	Watford	Man City	0	5	A	0	4	A	...	2.80	18	1.75	1.99	1.94	2.00	

380 rows × 65 columns

In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
```

In [4]:
```
y1 = pd.read_csv('2000-01.csv')
y2 = pd.read_csv('2001-02.csv')
y3 = pd.read_csv('2002-03.csv')
y4 = pd.read_csv('2003-04.csv')
y5 = pd.read_csv('2004-05.csv')
y6 = pd.read_csv('2005-06.csv')
y7 = pd.read_csv('2006-07.csv')
y8 = pd.read_csv('2007-08.csv')
y9 = pd.read_csv('2008-09.csv')
y10 = pd.read_csv('2009-10.csv')
y11 = pd.read_csv('2010-11.csv')
y12 = pd.read_csv('2011-12.csv')
y13 = pd.read_csv('2012-13.csv')
y14 = pd.read_csv('2013-14.csv')
y15 = pd.read_csv('2014-15.csv')
y16 = pd.read_csv('2015-16.csv')
y17 = pd.read_csv('2016-17.csv')
y18 = pd.read_csv('2017-18.csv')
y19 = pd.read_csv('2018-19.csv')
y20 = pd.read_csv('2019-20.csv')
```

```
KNN_classifier(train_X,train_Y,test_X,test_Y)
```

```
(0.384, 0.7375)
```

```
from sklearn.naive_bayes import GaussianNB

NaiveBayes_classifier(train_X,train_Y,test_X,test_Y)
```

```
(0.464, 0.4733433734939759)
```

```
SVM(train_X,train_Y,test_X,test_Y)
```

```
Accuracy: 50.2%
```

```
x1= playing_stat[['HS', 'HST', 'AR', 'HC', 'AF','AY']]
import sklearn
from sklearn.model_selection import train_test_split
# Shuffle and split the dataset into training and testing set.
train_X, test_X, train_Y, test_Y = train_test_split(x1, y,
                                                    test_size = 1000,
                                                    random_state = 2,
                                                    stratify = y)
print(KNN_classifier(train_X,train_Y,test_X,test_Y))
print(NaiveBayes_classifier(train_X,train_Y,test_X,test_Y))
print(SVM(train_X,train_Y,test_X,test_Y))
```

```
(0.387, 0.7245481927710843)
(0.456, 0.4724397590361446)
Accuracy: 51.300000000000004%
None
```