“CRICKET TEAM PREDICTION USING MACHINE LEARNING”

PREPARED BY

Kunal Sharma (2018BTECHCSE109)

FACULTY SUPERVISOR

DR. ALOK KUMAR

Department of Engineering (IET)
JK Lakshmipat University Jaipur

JUNE 2021

ACKNOWLEDGEMENT

Firstly, I would like to thank our course supervisor **Dr. Alok Kumar**. His constant advice and guidance played a vital role in making this report and project. He always gave me his suggestions that were crucial in making this report as flawless as possible. He also allowed me to approach him freely right from the very beginning of this work until the completion of my project. His encouragement, guidance, and suggestions provided me with necessary insight into the problem and constructed the way for the meaningful ending of this project on time. I have no hesitation to say that without his constant support and advice from time to time, I would not be able to finish work appropriately

Also, I want to extend my thankfulness to my parents for their constant support during this pandemic when I was working from home.

Project source code : https://github.com/kunaal07/DREAM-XI

Sincerely yours,

Kunal Sharma

ABSTRACT

Player selection is a crucial part of every sport, and cricket is no exception. A variety of factors influence the players' performance. The team management and captain select eleven players from the full roster for each match. After a study of numerous factors and player results, the best eleven players are chosen. Each batter contributes by scoring runs, while each bowler contributes by taking wickets and allowing the least amount of runs possible.

This research aims to predict team success based on historical player performance. Individual player performances, as well as their contribution to the team, such as best batting performance among available batsmen, best bowling performance among available bowlers, and best allrounder performance, will be highly useful in determining the eleven players. To create the problem's prediction models, we employed the like Random Forest classifier, Naive Bayes, KNN, Logistic regression, Support Vector Machine (SVM) AND Decision tree classifier. The Random Forest classifier was shown to be the most reliable for the challenges presented.

CONTENTS

FIGURE TABLE

CHAPTER 1- INTRODUCTION:

Increasing the number of Sports following not only our in our Country but globally, Formation of a good team for any sport is vital to its eventual success. Team selection in most sports is a subjective issue involving commonly accepted notions to form a good team. Selection of players for a cricket team is a complex task which can be viewed as a constrained multi-objective optimization and a multiple criteria decision-making problem. In cricket team formation, batting strength and bowling strength of a team are the major factors affecting its performance and an optimum trade-off needs to be reached in the formation of a good and successful team. it is important for us to select teams that can win us match or trophy. Considering Cricket, selecting teams these days from vast data having numerous statistical data and large number of rows and columns is one of the biggest problems and will require much more human resource if done physically. So, it is important for us to make this process quick. There are total 105 nations recognized by International Cricket Council (ICC) playing cricket and over millions have represented their country in domestic/international matches/leagues. With the introduction to new technical era and availability of fast and efficient algorithms for data analysis it is now quite easy to predict anything.

After completing the dataset, data preprocessing was done for training the modal by considering different features so that we get maximum accuracy as possible. All the classification algorithms like Random Forest classification, Naive Bayes, KNN, Logistic regression, Support Vector Machine (SVM) AND Decision tree classification have been tested for prediction. The algorithm with better accuracy will be considered as the best algorithm for achieving the goal of this project.

In my case I am predicting the playing 11 squad by using data from 2005- 2019.

1.1 Problem Statement

In this project, we need to analyses the data to predict the team using machine learning algorithms. The Exploratory Data Analysis and visualization of the insight are very important for the selection and present the results. We will be required to clean the data as well as perform feature engineering to improve our results.

1.2 Goal

Maximum part of our current work is focused majorly in two directions:

- Cleaning data by making it worth using by dropping some columns and encoding some.
- Predicting 11-member cricket squad by giving some features as an inputs and calculating their prediction accuracy.

1.3 Objective

The objective of this project is to predict a playing 11 squad of cricket considering data of more than 80+ nations and their players in national/international matches or leagues. With the help of machine learning techniques, we will predict 11 players according to the features.

CHAPTER 2- LITERATURE REVIEW:

The research paper that we have read, and the work done till date in field of predicting players is basically regional and no such notable work has been done so far. They have prepared a dataset of some regional matches or teams; on the basis of that data they are predicting good players, but they are still not been able to figure out playing 11 using features. Also, some fantasy apps like dream11, fantasy cricket allows user to create their playing 11.

Muthuswamy and lam [1] predicted Indian bowler's success against 7 international teams playing cricket and predicted nations against which they frequently played. They also predicted how many runs bowler is likely to give and wickets a bowler is going to take.

WikramaSinghe [2] predicted batsmen overall success in test cricket.

Barr and Kantor have [3] created a framework for determining performance of batsmen in limited overs.

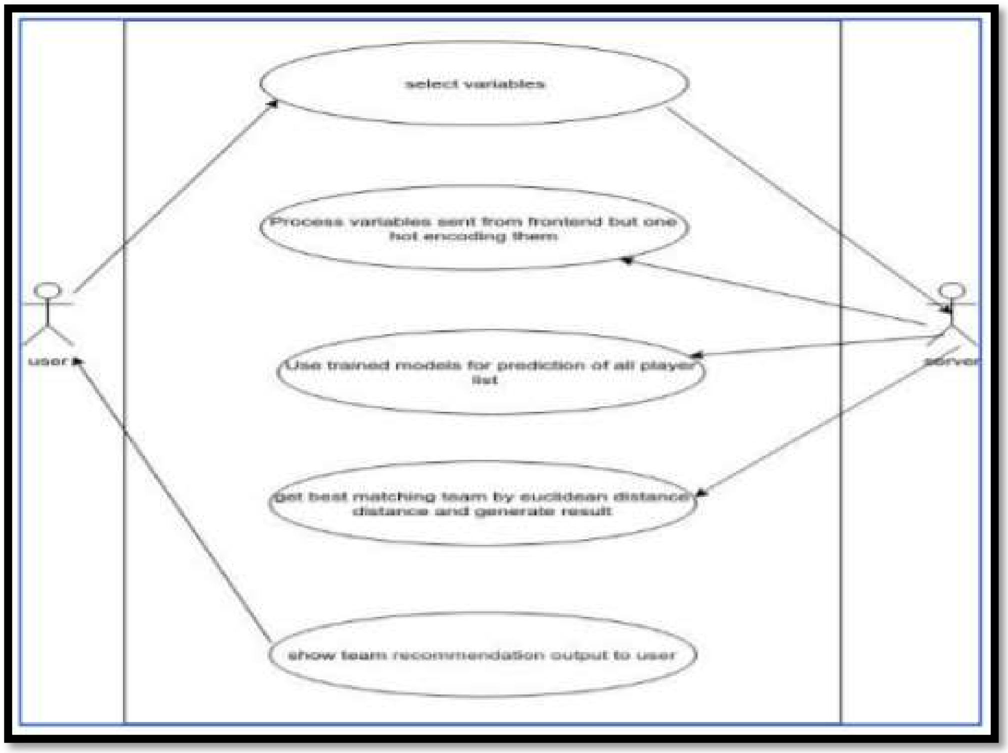

select variables
Process variables sent from frontend but one hot encoding them
user
server
Use trained models for prediction of all player list
get best matching team by euclidean distance distance and generate result
show team recommendation output to user

Fig 2.1 Use Case Diagram

CHAPTER 3 - TECHNOLOGY USED:

3.1 Software Details

- Anaconda Distribution (v5.1)
- Python (3.6.5)
- Packages Used:

  1. Pandas (0.22.1)
  2. Numpy (1.14.2)
  3. Sklearn (Tool for Machine learning)
  4. Matplotlib
  5. Seaborn

3.2 Hardware Details

- **CPU:** Intel Core 2 Quad CPU Q6600 @ 2.40GHz or greater.
- **System architecture:** 64-bit x86, 32-bit x86 with Windows or Linux.
- **Operating system:** Windows 7 or newer, 64-bit macOS 10.
- **RAM:** 4 GB or greater.

CHAPTER 4- DESCRIPTION OF DATA:

This dataset contains complete information about All cricket events whether or national or international matches/leagues/ICC Events . Dataset prepared from the span of 2005-2019. There are many factors that can be analysed from this dataset.

We have two datasets(initially) namely:-

1. Match.csv (1950 rows*25 columns)
2. Player_performance .csv(1906 rows*440 columns)
   Combined data:- (1950 rows* 465 columns)

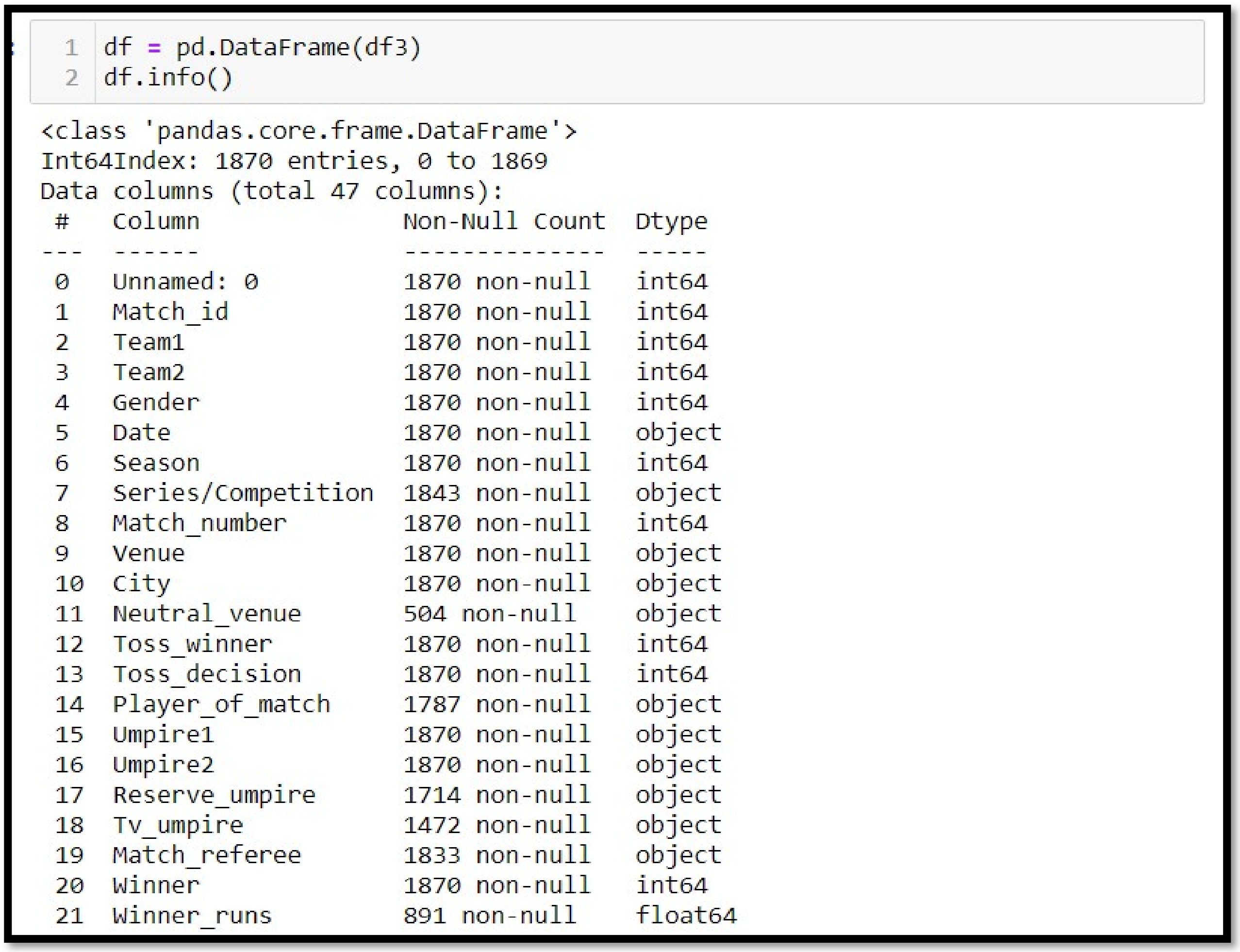
After cleaning of combined data, the size of datasets reduced to:-

1. Combined data:- (1870 rows*35 Columns)

CHAPTER 5- EXPLORATORY DATA ANALYSIS

With the use of summary statistics and graphical representations, exploratory data analysis refers to the crucial process of doing first investigations on data to uncover patterns, spot anomalies, test hypotheses, and check assumptions.

- To begin, I loaded the data collection and imported the relevant libraries (in this case, pandas, numpy, matplotlib, and seaborn).
- To get a closer look at the data, I used the pandas library's ".head()" function, which gives the first five observations in the data set. Similarly, ".tail()" retrieves the data set's last five observations.
- Using ".shape," I calculated the total number of rows and columns in the data collection.
- Using df.info()
  - Only float and integer values are present in the data.
  - There are no null/missing values in any variable column.

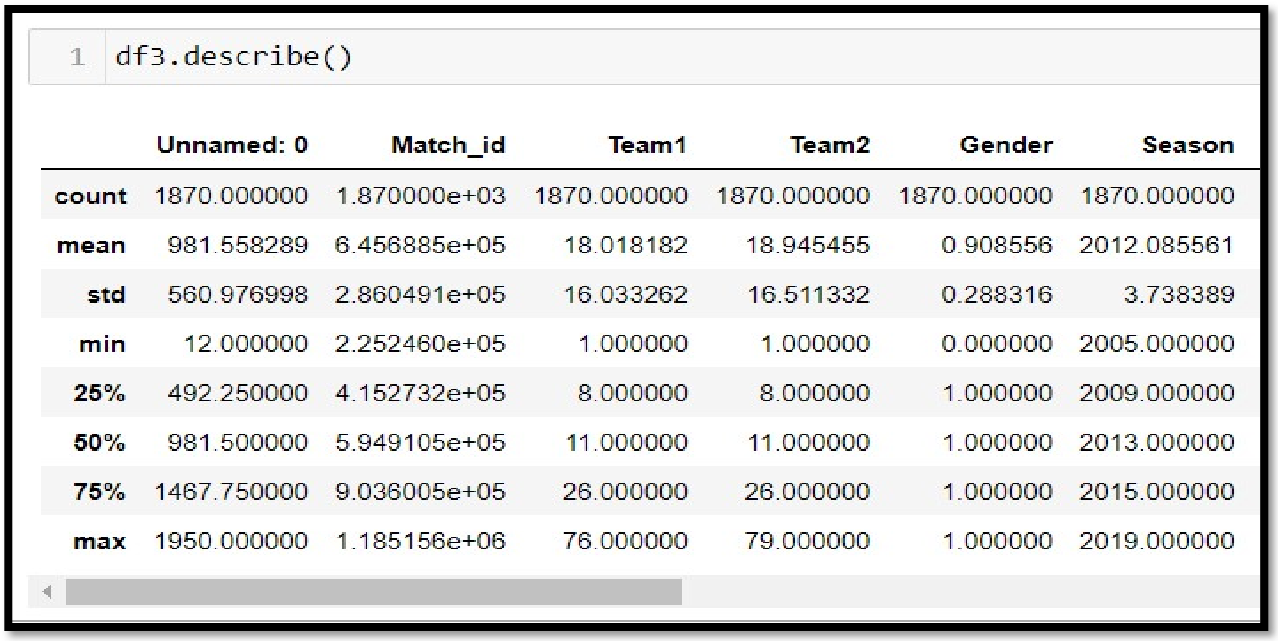


```
df = pd.DataFrame(df3)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1870 entries, 0 to 1869
Data columns (total 47 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Unnamed: 0           1870 non-null   int64
 1   Match_id             1870 non-null   int64
 2   Team1                1870 non-null   int64
 3   Team2                1870 non-null   int64
 4   Gender               1870 non-null   int64
 5   Date                 1870 non-null   object
 6   Season               1870 non-null   int64
 7   Series/Competition   1843 non-null   object
 8   Match_number         1870 non-null   int64
 9   Venue                1870 non-null   object
 10  City                 1870 non-null   object
 11  Neutral_venue        504 non-null    object
 12  Toss_winner          1870 non-null   int64
 13  Toss_decision        1870 non-null   int64
 14  Player_of_match      1787 non-null   object
 15  Umpire1              1870 non-null   object
 16  Umpire2              1870 non-null   object
 17  Reserve_umpire       1714 non-null   object
 18  Tv_umpire            1472 non-null   object
 19  Match_referee        1833 non-null   object
 20  Winner               1870 non-null   int64
 21  Winner_runs          891 non-null    float64
```

Fig 5.1 df3.info()

- The pandas describe() method is extremely useful for obtaining various summary statistics. The count, mean, standard deviation, minimum and maximum values, and quantiles of the data are returned by this function.

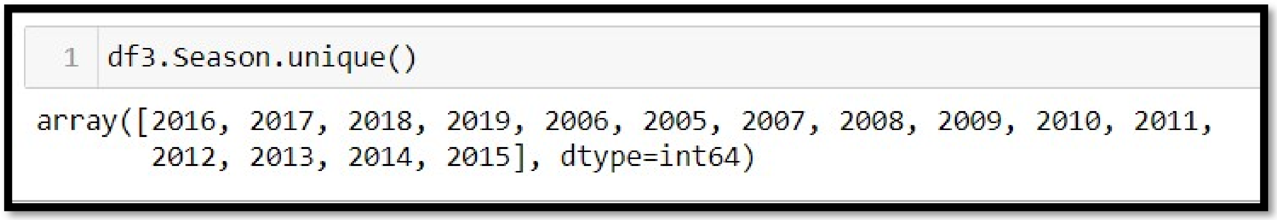


```
df3.describe()
```

	Unnamed: 0	Match_id	Team1	Team2	Gender	Season
count	1870.000000	1.870000e+03	1870.000000	1870.000000	1870.000000	1870.000000
mean	981.558289	6.456885e+05	18.018182	18.945455	0.908556	2012.085561
std	560.976998	2.860491e+05	16.033262	16.511332	0.288316	3.738389
min	12.000000	2.252460e+05	1.000000	1.000000	0.000000	2005.000000
25%	492.250000	4.152732e+05	8.000000	8.000000	1.000000	2009.000000
50%	981.500000	5.949105e+05	11.000000	11.000000	1.000000	2013.000000
75%	1467.750000	9.036005e+05	26.000000	26.000000	1.000000	2015.000000
max	1950.000000	1.185156e+06	76.000000	79.000000	1.000000	2019.000000

Fig 5.2 df3.describe()

- The nature of the target variable/dependent variable is discrete and categorical.
    - df3.Season.unique()

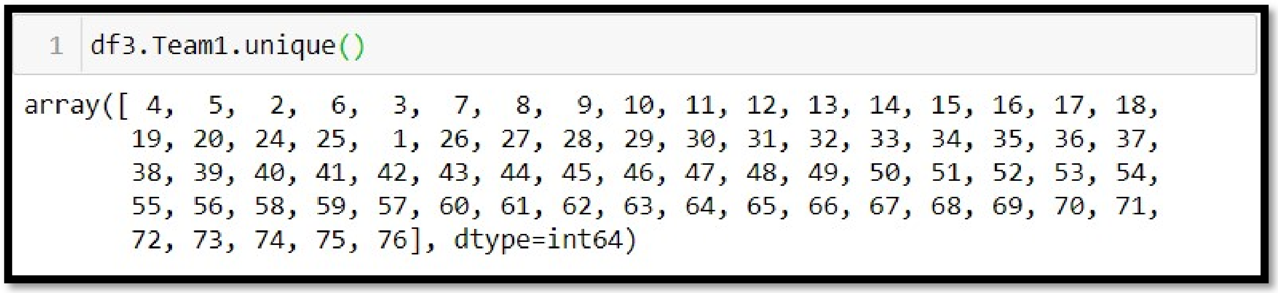


```
df3.Season.unique()
```

```
array([2016, 2017, 2018, 2019, 2006, 2005, 2007, 2008, 2009, 2010, 2011,
       2012, 2013, 2014, 2015], dtype=int64)
```

Fig 5.3 df3.Season.unique()

- df3.Team1.unique() (encoded teams)

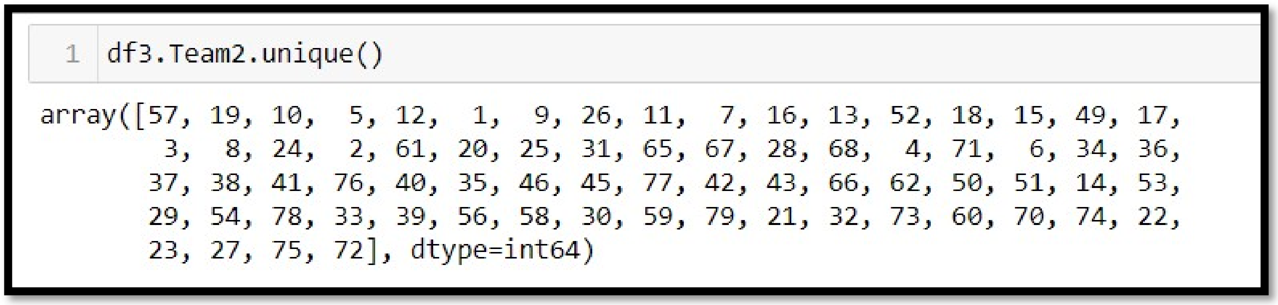


```
df3.Team1.unique()
```

```
array([ 4,  5,  2,  6,  3,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19, 20, 24, 25,  1, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
       38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54,
       55, 56, 58, 59, 57, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71,
       72, 73, 74, 75, 76], dtype=int64)
```

Fig 5.4 df3.team1.unique()

- df3.Team2.unique() (encoded teams)

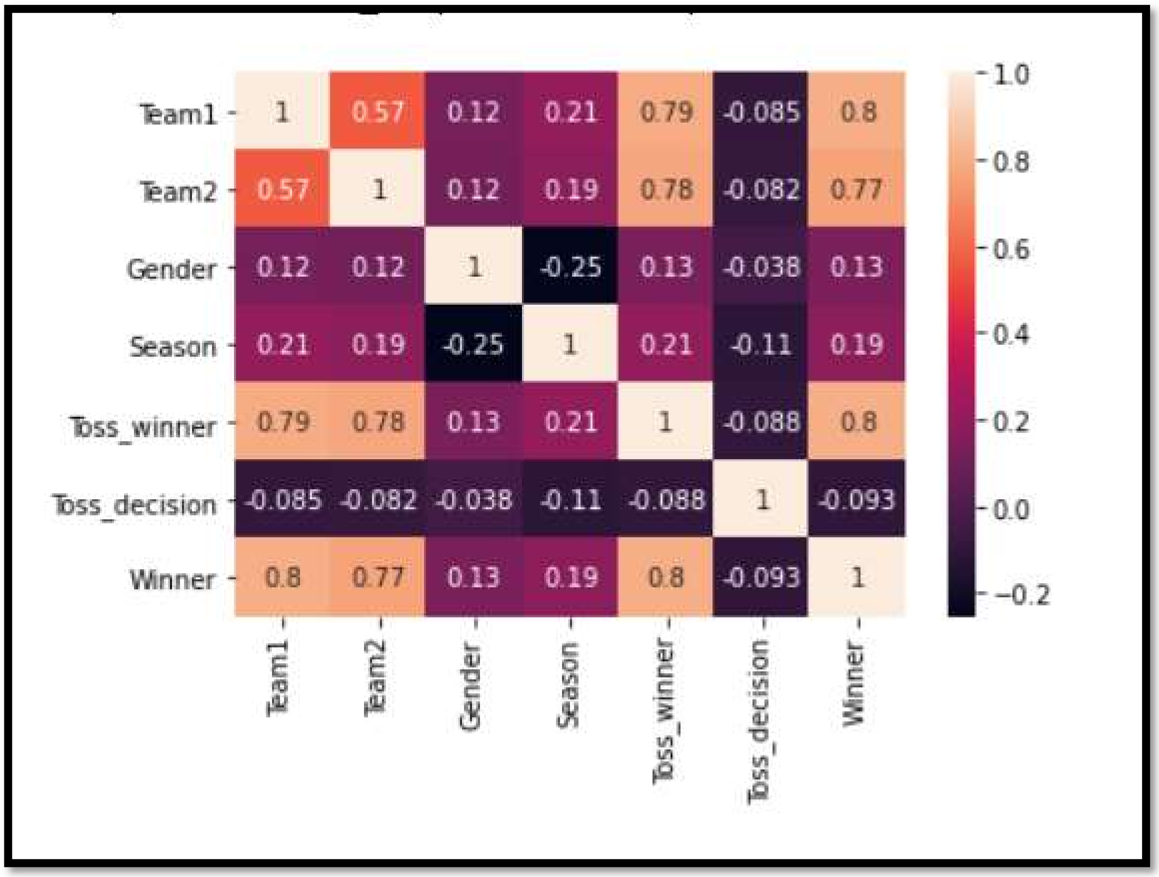


```
df3.Team2.unique()
```

```
array([57, 19, 10,  5, 12,  1,  9, 26, 11,  7, 16, 13, 52, 18, 15, 49, 17,
        3,  8, 24,  2, 61, 20, 25, 31, 65, 67, 28, 68,  4, 71,  6, 34, 36,
       37, 38, 41, 76, 40, 35, 46, 45, 77, 42, 43, 66, 62, 50, 51, 14, 53,
       29, 54, 78, 33, 39, 56, 58, 30, 59, 79, 21, 32, 73, 60, 70, 74, 22,
       23, 27, 75, 72], dtype=int64)
```

Fig 5.5 df3.Team2.unique()

- Heat map visualisation
  - Positive correlation is represented by dark tones, whereas negative correlation is represented by lighter shades.
  - If you set annot=True, you'll obtain values in grid-cells that show how characteristics are connected to one another.



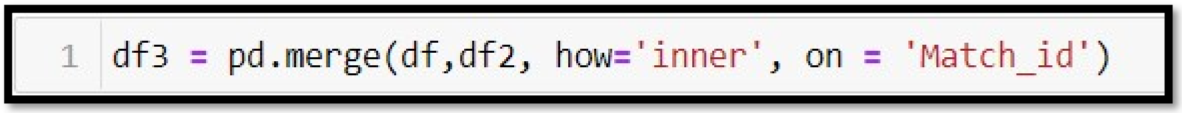
	Team1	Team2	Gender	Season	Toss_winner	Toss_decision	Winner
Team1	1	0.57	0.12	0.21	0.79	-0.085	0.8
Team2	0.57	1	0.12	0.19	0.78	-0.082	0.77
Gender	0.12	0.12	1	-0.25	0.13	-0.038	0.13
Season	0.21	0.19	-0.25	1	0.21	-0.11	0.19
Toss_winner	0.79	0.78	0.13	0.21	1	-0.088	0.8
Toss_decision	-0.085	-0.082	-0.038	-0.11	-0.088	1	-0.093
Winner	0.8	0.77	0.13	0.19	0.8	-0.093	1

Fig 5.6 Heat map

CHAPTER 6- FEATURE ENGINEERING:

The most important art in machine learning is feature engineering, which makes the difference between a good model and a terrible model.

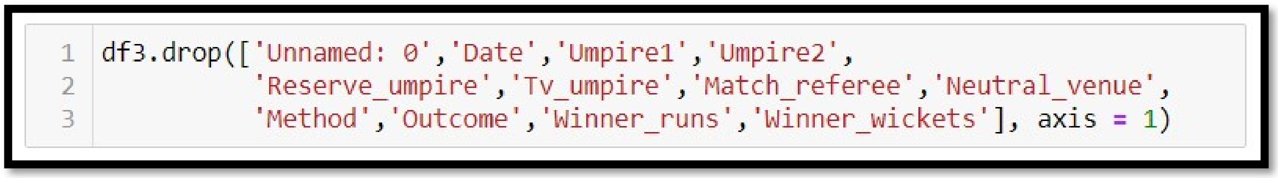
- We were given with two data frames namely match.csv (df) and player_performance.csv(df2) and combined them to new data frame df3 .



```python
df3 = pd.merge(df,df2, how='inner', on = 'Match_id')
```

Fig 6.1 Combining dataframes

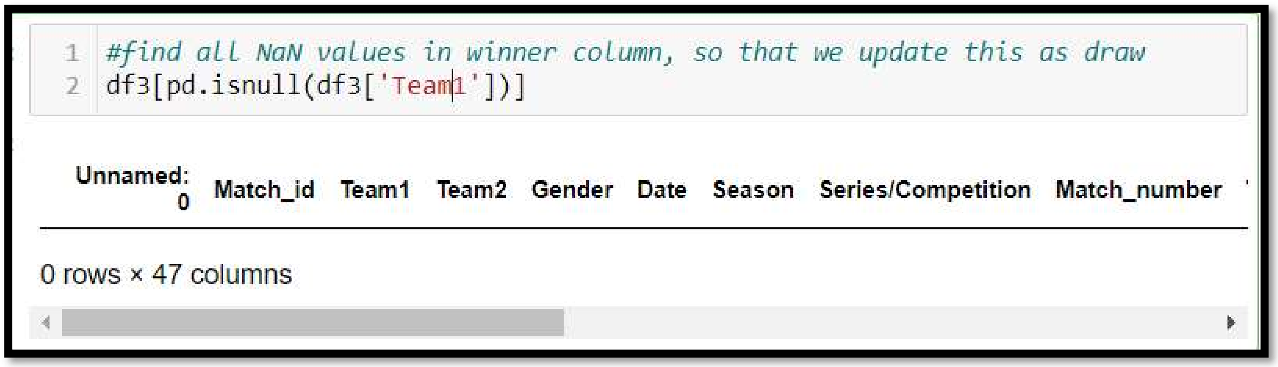
- Then we dropped all the columns that are not required in df3.



```python
df3.drop(['Unnamed: 0','Date','Umpire1','Umpire2',
          'Reserve_umpire','Tv_umpire','Match_referee','Neutral_venue',
          'Method','Outcome','Winner_runs','Winner_wickets'], axis = 1)
```

Fig 6.2 Dropping not useful columns

- Checking null values in Team1, Team2, Toss winner, City, Venue, Winner.

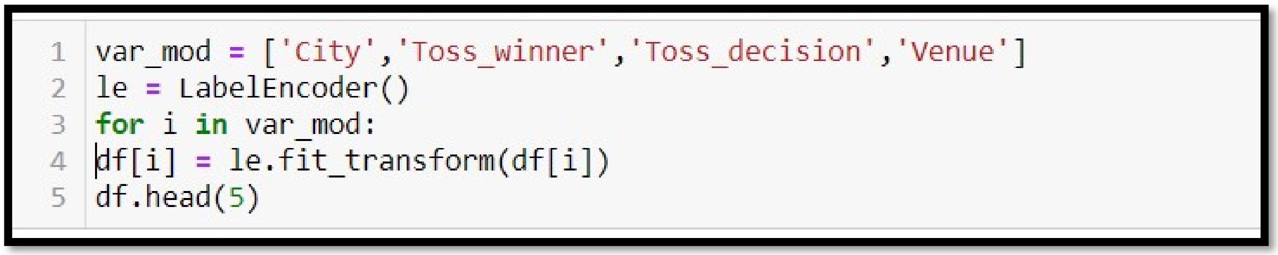


```python
#find all NaN values in winner column, so that we update this as draw
df3[pd.isnull(df3['Team1'])]
```

```
Unnamed:
       0  Match_id  Team1  Team2  Gender  Date  Season  Series/Competition  Match_number

0 rows × 47 columns
```

Fig 6.3 Checking null values

- Converting categorical to numerical dataset.



```python
var_mod = ['City','Toss_winner','Toss_decision','Venue']
le = LabelEncoder()
for i in var_mod:
df[i] = le.fit_transform(df[i])
df.head(5)
```

Fig 6.4 Converting data frame to numerical

<u>CHAPTER 7- MODEL SELECTION:</u>

Unlike currently available techniques, the suggested system focuses on projecting the best playing 11 rather than just a single player's performance. The Random Forest Algorithm is used in the system, which has been shown to be the best and most adaptive for this model. The player's performance must not only be predicted, but a choice must be made if the player is an excellent candidate for inclusion in the team based on previous records and other factors. Random forests, also known as random decision forests, are a learning method for classification, regression, and other procedures that work by constructing a network of decision trees during the training and performance mode of individual trees during class (classification) or mean prediction (regression). Random forests address decision trees' tendency to overfit their training range.
Random forest is made up of many characters decision trees that work together as an ensemble. Each tree in the random forest makes a category forecast, and the elegance with the majority becomes the prediction of our model.

<u>CHAPTER 8- IMPLEMENTATION:</u>

The implementation of this project is divided into the following steps –

8.1. Data collection

We were given both the datasets, so no data collection is done as such.

8.2. Data Preprocessing

We were given with two datasets namely match.csv and player_performance.csv and combined them, after combining we drop some columns like Batting order, Inning batted, Bat Bowled, Bat Dismissed, Bat runs, 4s, 6s, ball faced, dots, batting team, inning bowled, over, runs, wicket, maiden, extra,etc.
Also we used encoding for 6 columns:-

- **<u>Encoded values of team1/team2/toss_winner/winner</u>**
  {'Australia' : 1, 'Scotland' : 2, 'Papua New Guinea' : 3, 'Netherlands' : 4,'Kenya' : 5, 'Zimbabwe' : 6, 'New Zealand' : 7, 'England' : 8, 'South Africa' : 9,'India' : 10,'Pakistan' : 11,'Bangladesh' : 12,'Adelaide Strikers' : 13,'Perth Scorchers' : 14,'Sydney Thunder' : 15, 'Brisbane Heat' : 16,'Sydney Sixers' : 17, 'Melbourne Stars' : 18, 'Hong Kong' : 19, 'West Indies' : 20,'Vanuatu' : 21, 'Maldives' : 22, 'Saudi Arabia' : 23,'Ireland' : 24, 'Afghanistan' : 25, 'Sri Lanka' : 26, 'Malaysia' : 27, 'Denmark' : 28,'Bermuda' : 29, 'United States of America'

: 30, 'Italy' : 31, 'Japan' : 32,'United Arab Emirates' : 33,'Thailand' : 34,'Rising Pune Supergiant' : 35, 'Mumbai Indians' : 36,'Kolkata Knight Riders' : 37,'Kings XI Punjab' : 38,'Delhi Daredevils' : 39,'Royal Challengers Bangalore' : 40, 'Sunrisers Hyderabad' : 41, 'Nigeria' : 42, 'Ghana' : 43, 'Botswana' : 44,'Sierra Leone' : 45,'Tanzania' : 46,'Qatar' : 47,'Bahrain' : 48, 'Melbourne Renegades' : 49, 'Rajasthan Royals' : 50, 'Chennai Super Kings' : 51,'Hobart Hurricanes' : 52, 'Delhi Capitals' : 53, 'Canada' : 54, 'Africa XI' : 55,'Deccan Chargers' : 56, 'Nepal' : 57, 'Argentina' : 58, 'Cayman Islands' : 59,'Pune Warriors' : 60, 'Singapore' : 61, 'Bhutan' : 62, 'Samoa' : 63, 'Belgium' : 64, 'Oman' : 65,'Kuwait' : 66, 'Jersey' : 67, 'China' : 68, 'South Korea' : 69, 'Uganda' : 70, 'Namibia' : 71,'Fiji' : 72, 'Norway' : 73, 'Guernsey' : 74, 'Suriname' : 75,'Gujarat Lions' : 76,'Zambia' : 77,'Asia XI' : 78, 'Kochi Tuskers Kerala' : 79}

- **Encoded value of gender**
  {'Female' : 0, 'Male' : 1}
- **Encoded value of toss_Decision**
  {'Field' : 0, 'Bat' : 1}

8.3 Feature selection

Features selection is done which can be used to build the model. The attributes used for feature selection are X coordinate and Y coordinate. On X we used Team1, Team2, Gender, Season , Toss winner, Toss decision, Winner as attributes and on Y coordinate, we use Player1, Player1, Player2, Player3, Player4, Player5, Player6, Player7, Player8, Player9, Player10, Player11, as an attribute.

8.4. Building and Training Model

After feature selection, features and players attributes are used for training. The dataset is divided into pairs of x train, y train, and test, y test for 11 times for eacy player. The algorithms model is imported from sklearn. Building models is done using models. Fit (xtrain, ytrain). We applied models of Random foresetand compared it with the applied algos of classifier Naive Bayes, Decision tree classifier, Support vector machine classifier, , K-Neighbors classifiers, Logistic regression classifier.

8.5. Prediction

After the model is built using the above process, a prediction is done using model. Predict(xtest). The accuracy is calculated using accuracy score imported from metrics – metrics, accuracy_score (ytest, predicted) .

8.6. Visualization

Using the matplotlib library from sklearn. Analysis of the dataset is done by plotting various graphs.

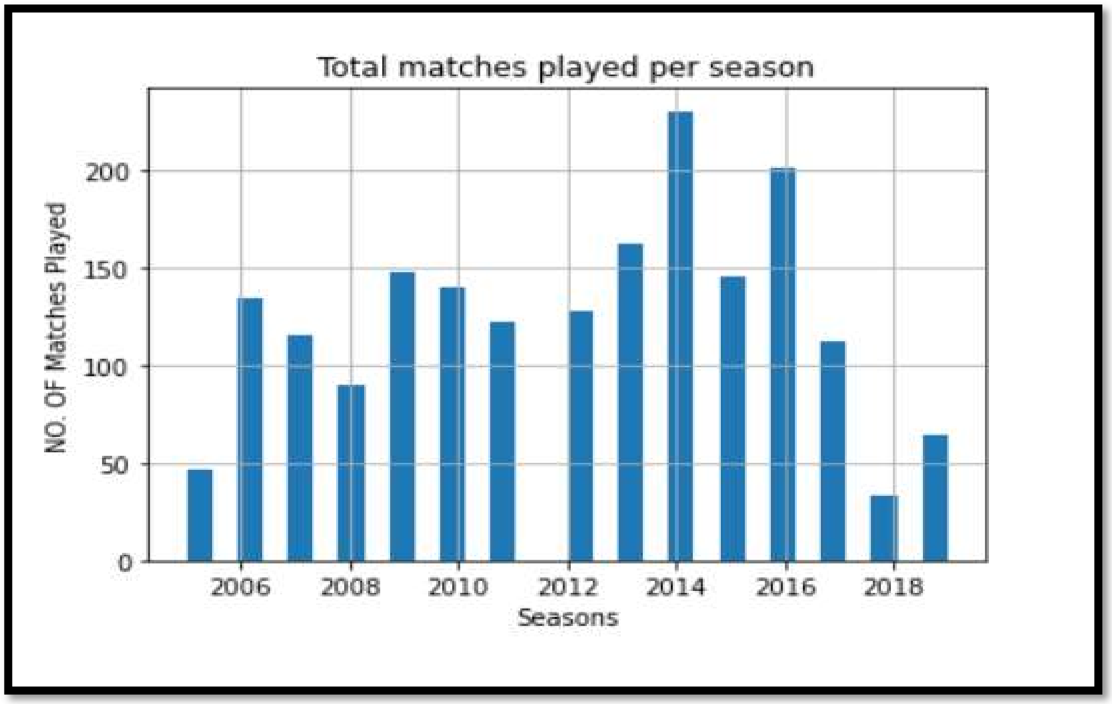


Fig 8.1 Total Matches played per season

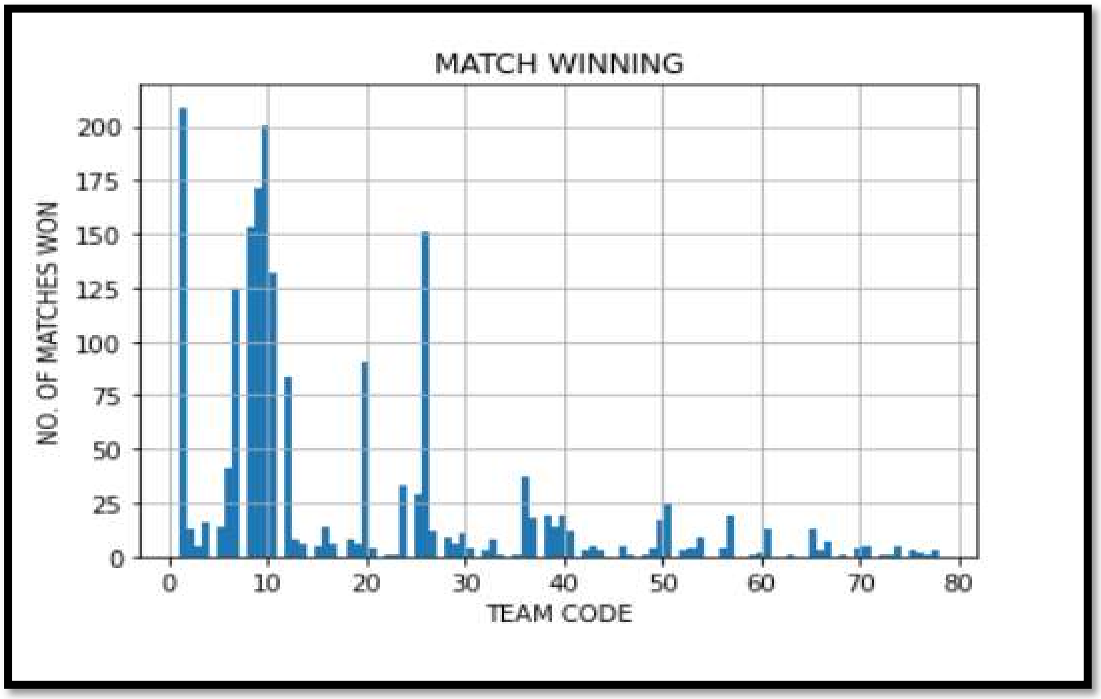


Fig 8.2 Total Matches won by each team (encoded team number)

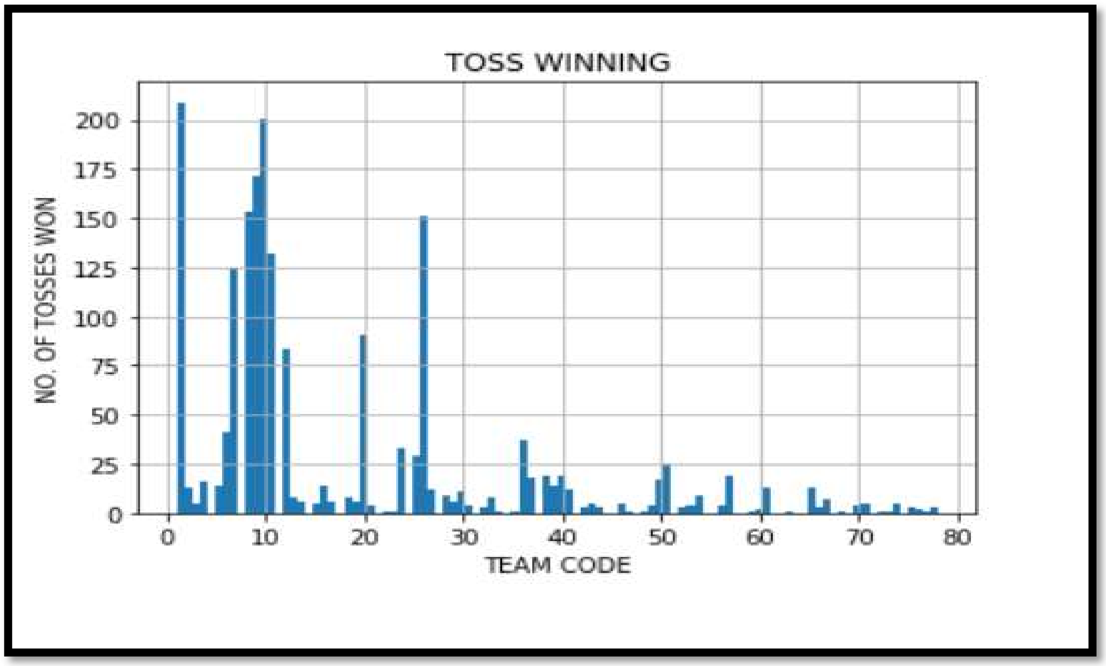
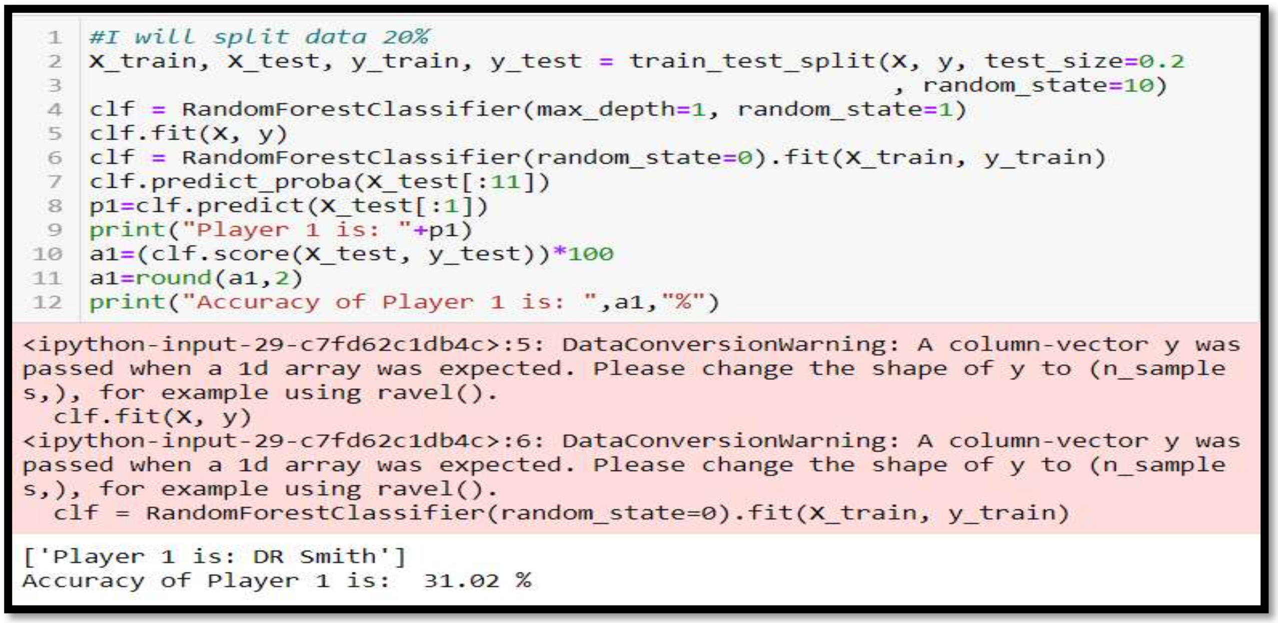


Fig 8.3 Total tosses won by each team (encoded team number)

CHAPTER 9- PREDICTION:

After running model 11 times for 11 different players, The system gives a team of 11 players based on the features given as input by us.

- PLAYER 1 PREDICTION:



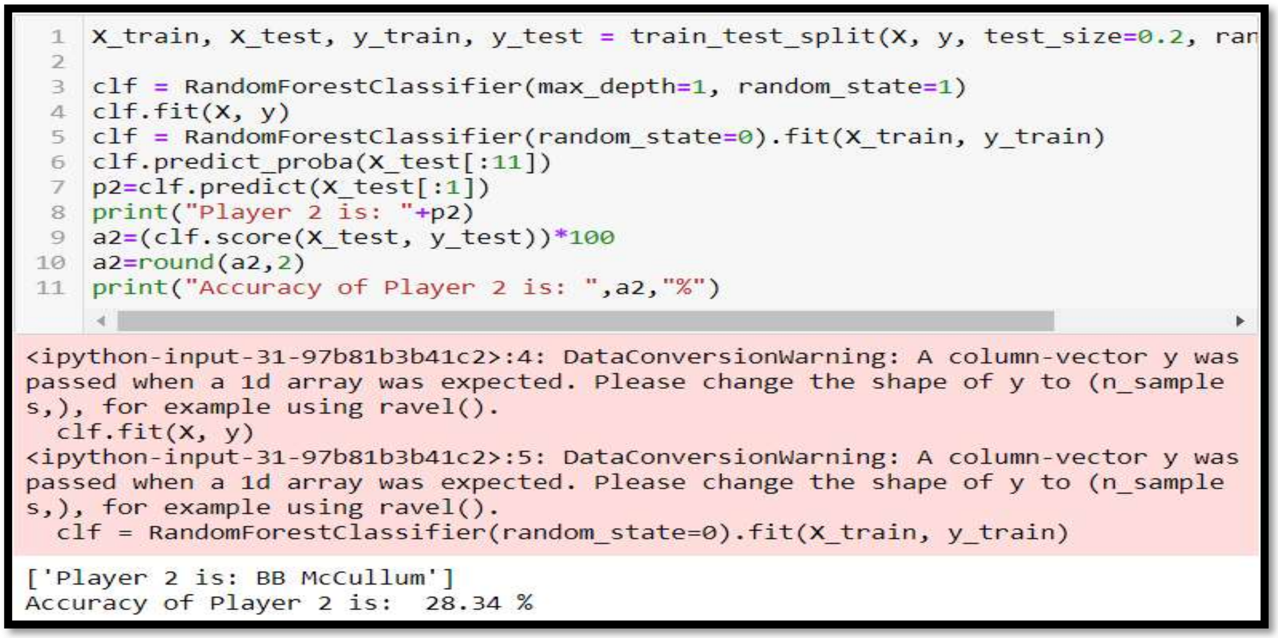
```
#I will split data 20%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                     , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p1=clf.predict(X_test[:1])
print("Player 1 is: "+p1)
a1=(clf.score(X_test, y_test))*100
a1=round(a1,2)
print("Accuracy of Player 1 is: ",a1,"%")
```

```
<ipython-input-29-c7fd62c1db4c>:5: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-29-c7fd62c1db4c>:6: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 1 is: DR Smith']
Accuracy of Player 1 is:  31.02 %
```

Fig 9.1 Player 1 prediction

- PLAYER 2 PREDICTION:



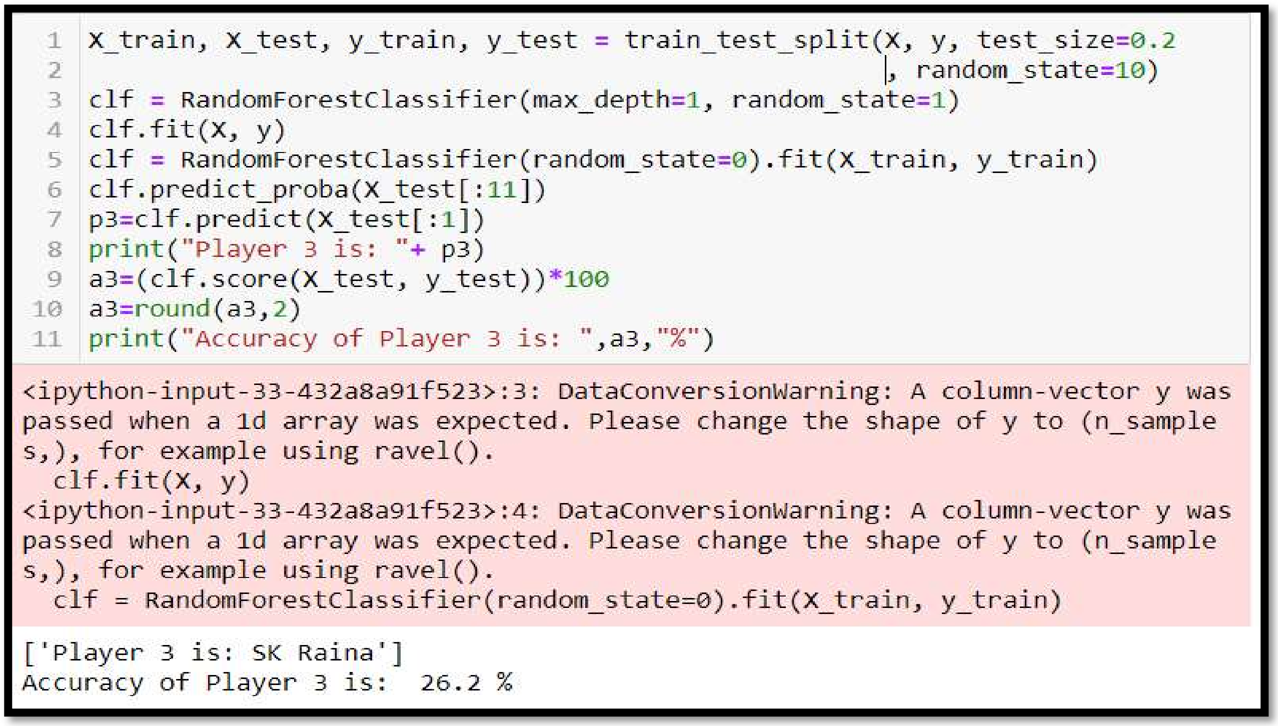
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran

clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p2=clf.predict(X_test[:1])
print("Player 2 is: "+p2)
a2=(clf.score(X_test, y_test))*100
a2=round(a2,2)
print("Accuracy of Player 2 is: ",a2,"%")
```

```
<ipython-input-31-97b81b3b41c2>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-31-97b81b3b41c2>:5: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 2 is: BB McCullum']
Accuracy of Player 2 is:  28.34 %
```

Fig 9.2 Player 2 prediction

- PLAYER 3 PREDICTION:



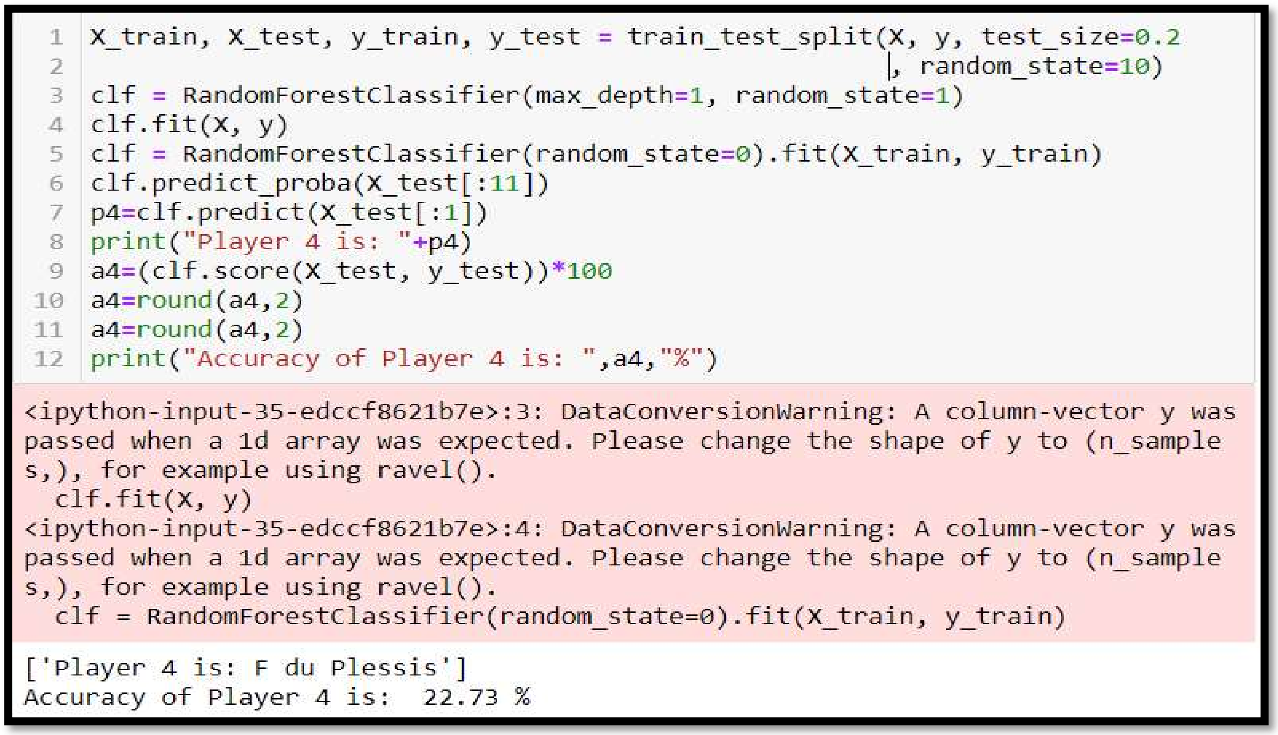
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                    , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p3=clf.predict(X_test[:1])
print("Player 3 is: "+ p3)
a3=(clf.score(X_test, y_test))*100
a3=round(a3,2)
print("Accuracy of Player 3 is: ",a3,"%")
```

```
<ipython-input-33-432a8a91f523>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-33-432a8a91f523>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 3 is: SK Raina']
Accuracy of Player 3 is:  26.2 %
```

Fig 9.3 Player 3 prediction

- PLAYER 4 PREDICTION:



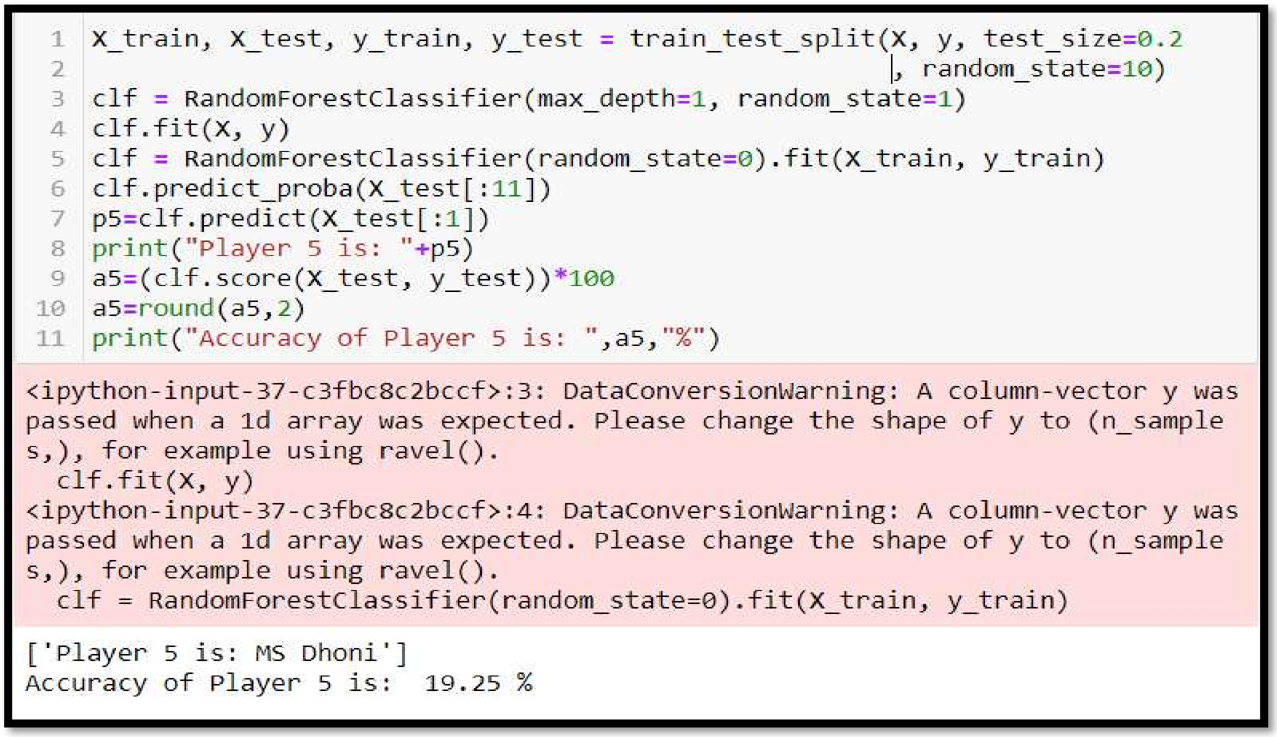
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                    , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p4=clf.predict(X_test[:1])
print("Player 4 is: "+p4)
a4=(clf.score(X_test, y_test))*100
a4=round(a4,2)
a4=round(a4,2)
print("Accuracy of Player 4 is: ",a4,"%")
```

```
<ipython-input-35-edccf8621b7e>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-35-edccf8621b7e>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 4 is: F du Plessis']
Accuracy of Player 4 is:  22.73 %
```

Fig 9.4 Player 4 prediction

- PLAYER 5 PREDICTION:



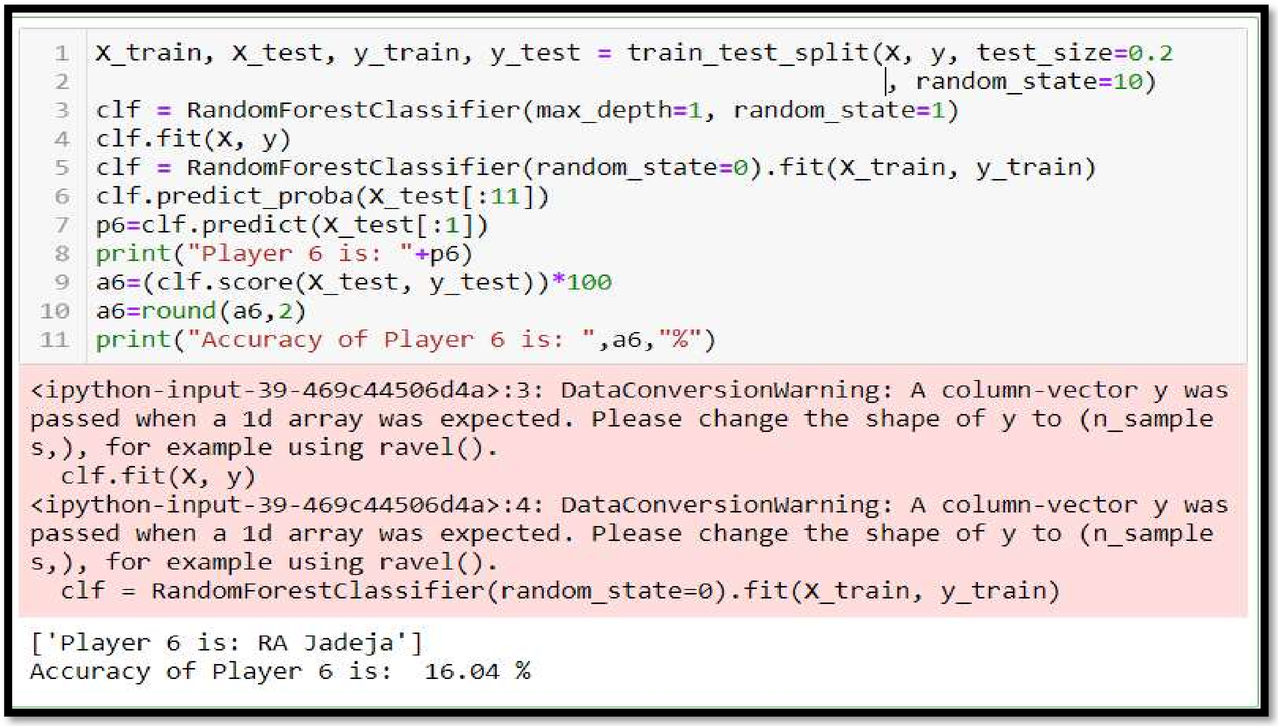
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                    , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p5=clf.predict(X_test[:1])
print("Player 5 is: "+p5)
a5=(clf.score(X_test, y_test))*100
a5=round(a5,2)
print("Accuracy of Player 5 is: ",a5,"%")
```

```
<ipython-input-37-c3fbc8c2bccf>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-37-c3fbc8c2bccf>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 5 is: MS Dhoni']
Accuracy of Player 5 is:  19.25 %
```

Fig 9.5 Player 5 prediction

- PLAYER 6 PREDICTION:



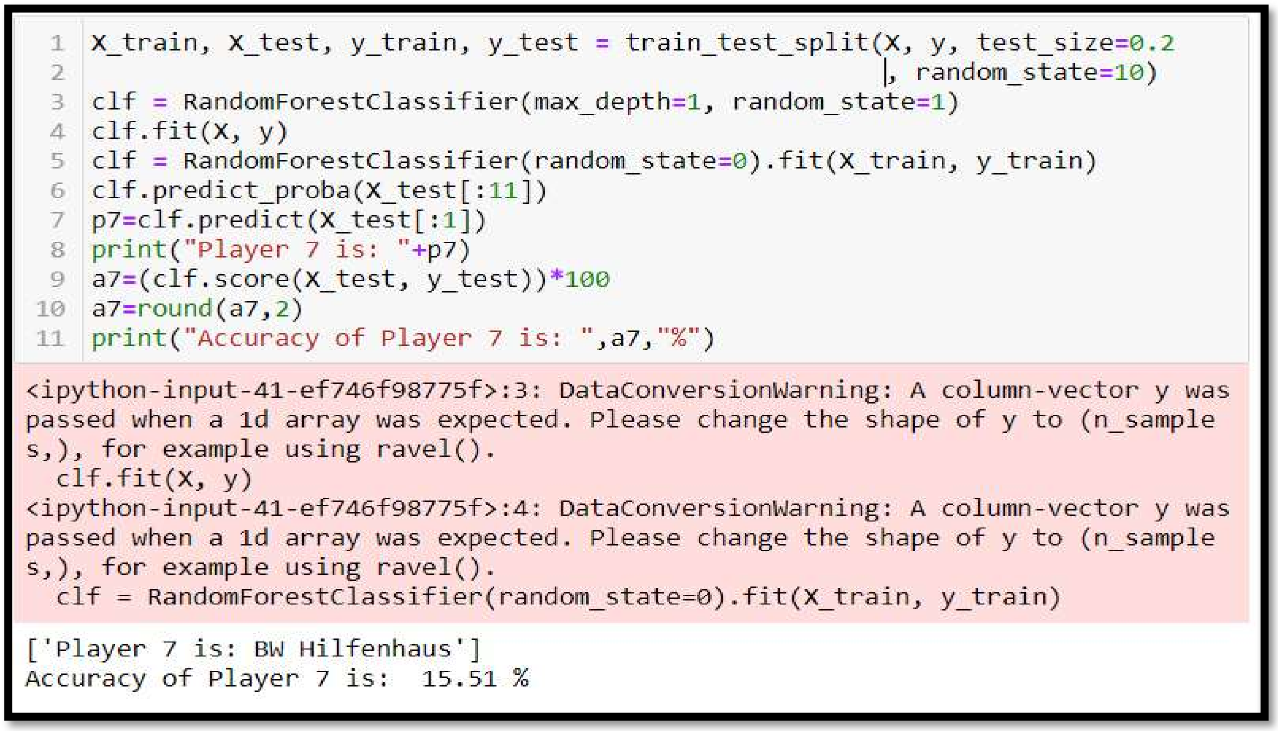
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                    , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p6=clf.predict(X_test[:1])
print("Player 6 is: "+p6)
a6=(clf.score(X_test, y_test))*100
a6=round(a6,2)
print("Accuracy of Player 6 is: ",a6,"%")
```

```
<ipython-input-39-469c44506d4a>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-39-469c44506d4a>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 6 is: RA Jadeja']
Accuracy of Player 6 is:  16.04 %
```

Fig 9.6 Player 6 prediction

- PLAYER 7 PREDICTION:



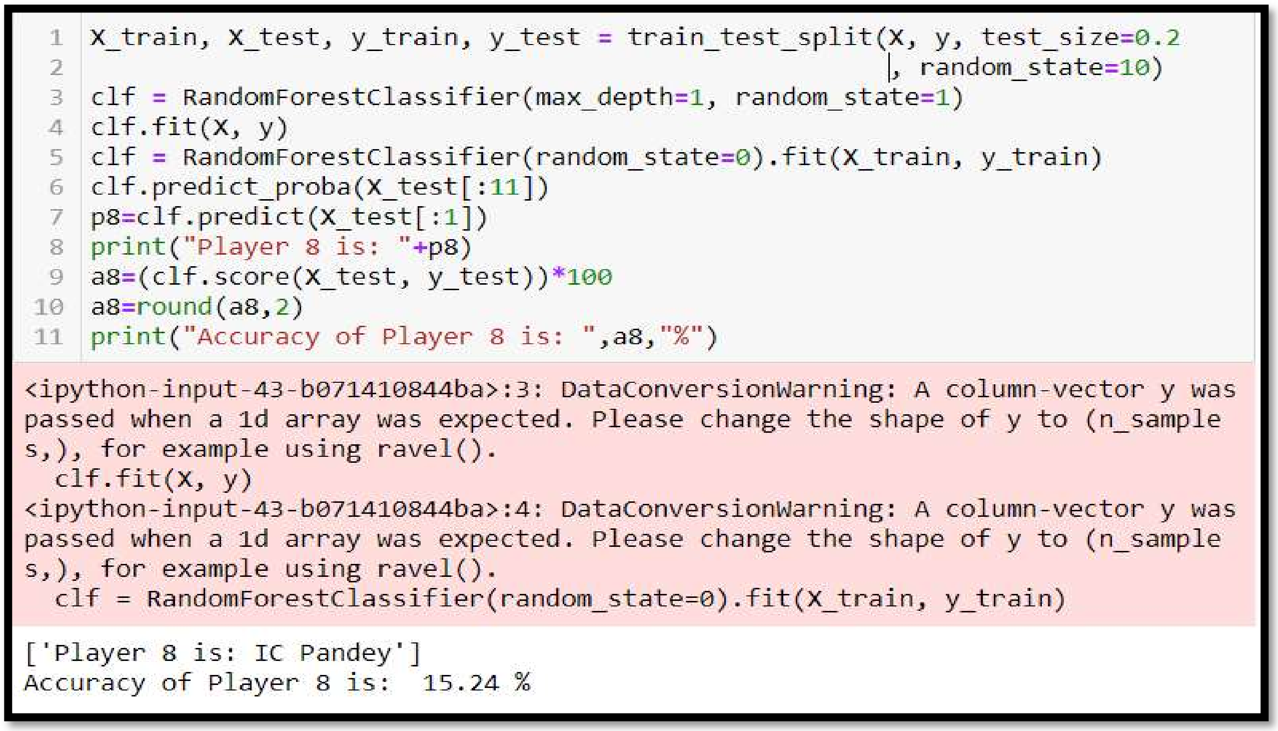
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                     , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p7=clf.predict(X_test[:1])
print("Player 7 is: "+p7)
a7=(clf.score(X_test, y_test))*100
a7=round(a7,2)
print("Accuracy of Player 7 is: ",a7,"%")
```

```
<ipython-input-41-ef746f98775f>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-41-ef746f98775f>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 7 is: BW Hilfenhaus']
Accuracy of Player 7 is:  15.51 %
```

Fig 9.7 Player 7 prediction

- PLAYER 8 PREDICTION:



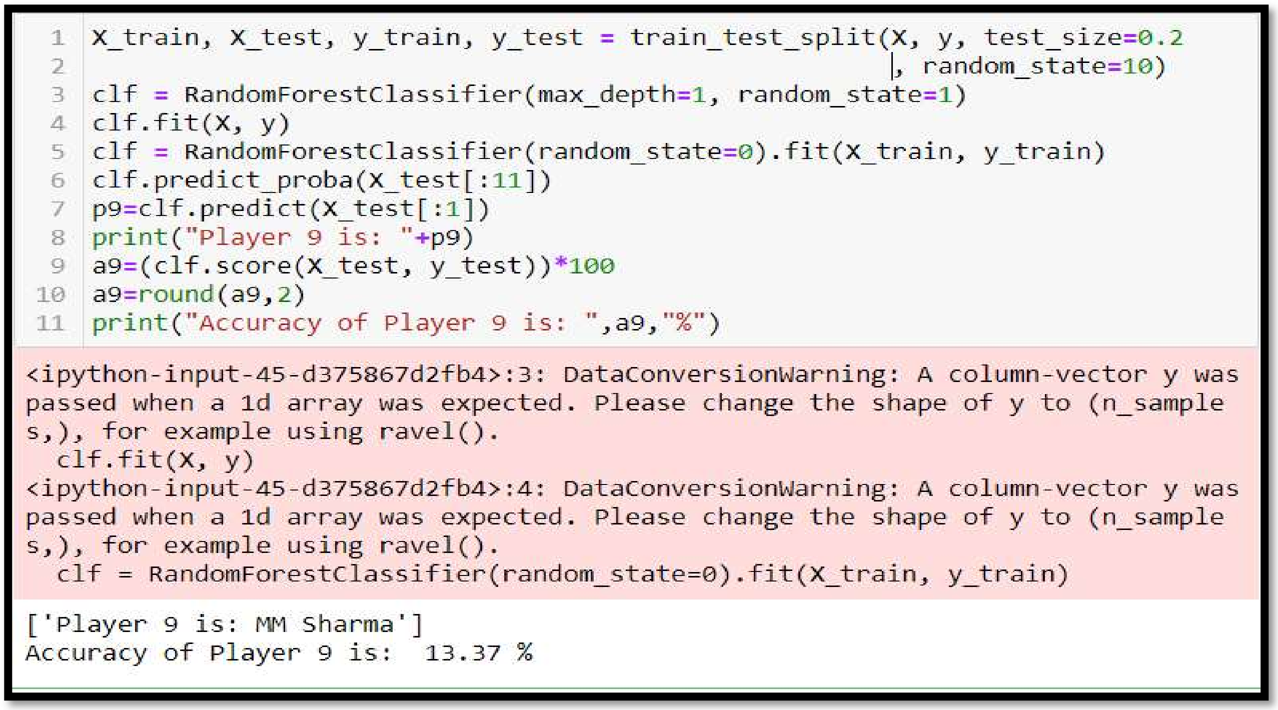
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                     , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p8=clf.predict(X_test[:1])
print("Player 8 is: "+p8)
a8=(clf.score(X_test, y_test))*100
a8=round(a8,2)
print("Accuracy of Player 8 is: ",a8,"%")
```

```
<ipython-input-43-b071410844ba>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-43-b071410844ba>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 8 is: IC Pandey']
Accuracy of Player 8 is:  15.24 %
```

Fig 9.8 Player 8 prediction

- PLAYER 9 PREDICTION:



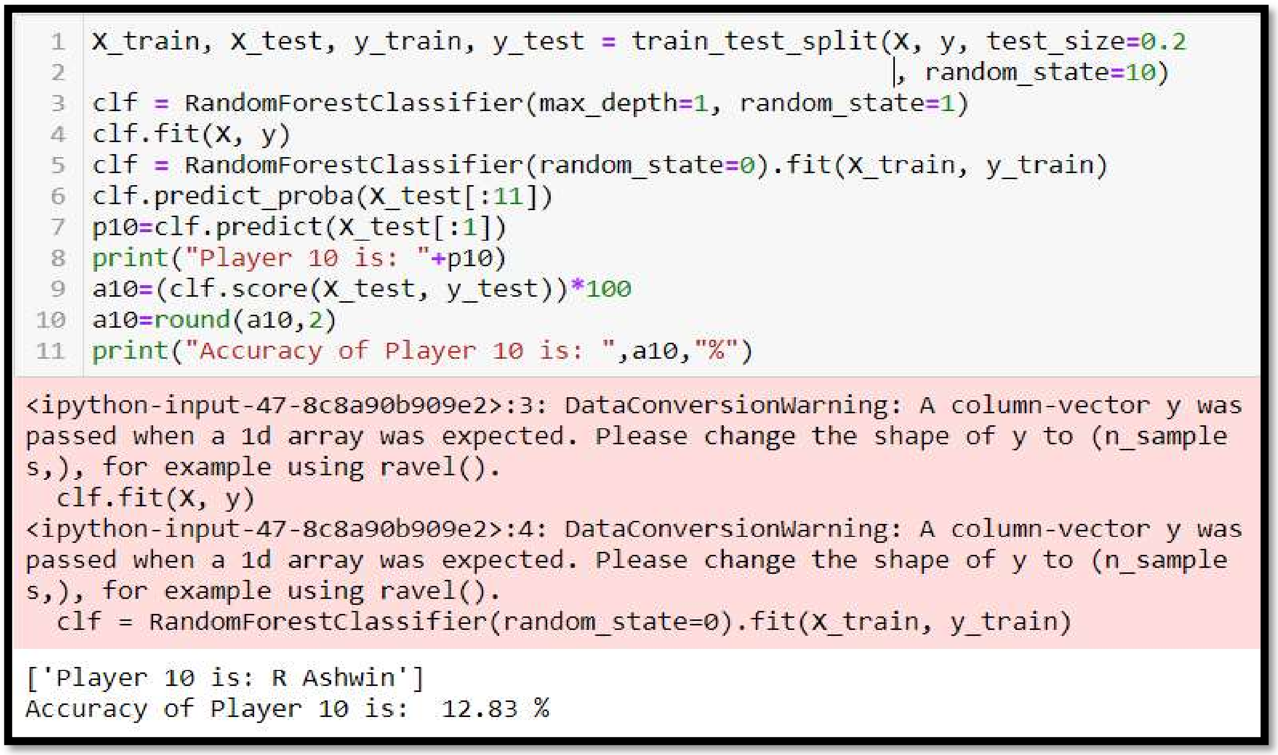
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                     , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p9=clf.predict(X_test[:1])
print("Player 9 is: "+p9)
a9=(clf.score(X_test, y_test))*100
a9=round(a9,2)
print("Accuracy of Player 9 is: ",a9,"%")
```

```
<ipython-input-45-d375867d2fb4>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-45-d375867d2fb4>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 9 is: MM Sharma']
Accuracy of Player 9 is:  13.37 %
```

Fig 9.9 Player 9 prediction

- PLAYER 10 PREDICTION:



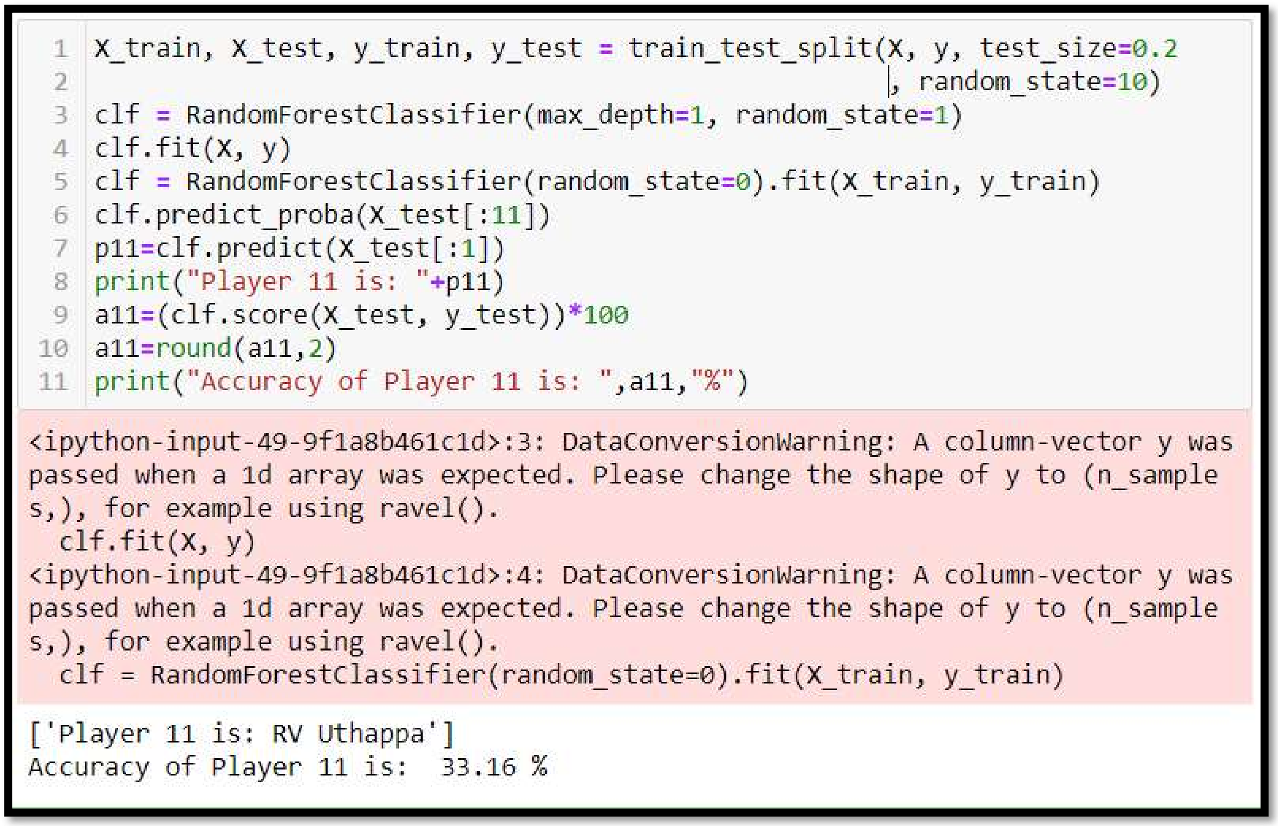
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                     , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p10=clf.predict(X_test[:1])
print("Player 10 is: "+p10)
a10=(clf.score(X_test, y_test))*100
a10=round(a10,2)
print("Accuracy of Player 10 is: ",a10,"%")
```

```
<ipython-input-47-8c8a90b909e2>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-47-8c8a90b909e2>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 10 is: R Ashwin']
Accuracy of Player 10 is:  12.83 %
```

Fig 9.10 Player 10 prediction

- PLAYER 11 PREDICTION:



```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
                                                     , random_state=10)
clf = RandomForestClassifier(max_depth=1, random_state=1)
clf.fit(X, y)
clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
clf.predict_proba(X_test[:11])
p11=clf.predict(X_test[:1])
print("Player 11 is: "+p11)
a11=(clf.score(X_test, y_test))*100
a11=round(a11,2)
print("Accuracy of Player 11 is: ",a11,"%")
```

```
<ipython-input-49-9f1a8b461c1d>:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf.fit(X, y)
<ipython-input-49-9f1a8b461c1d>:4: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  clf = RandomForestClassifier(random_state=0).fit(X_train, y_train)
```

```
['Player 11 is: RV Uthappa']
Accuracy of Player 11 is:  33.16 %
```

Fig 9.11 Player 11 prediction

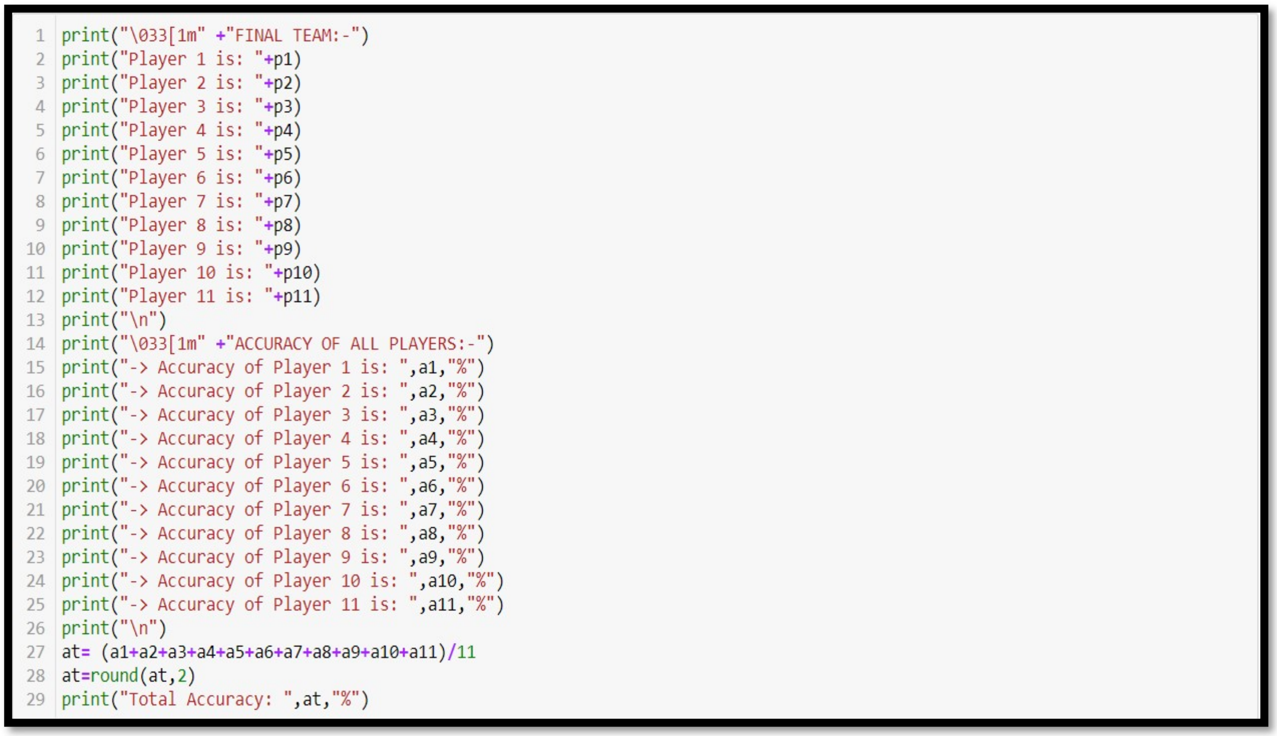
CHAPTER 10- CONCLUSION:

After implementing the algorithms, this section covers the result of accuracy comparison of Random Forest classification, Naive Bayes, KNN, Logistic regression, Support Vector Machine (SVM) AND Decision tree. All these algorithms come under the classification machine learning algorithm category. The algorithm was run to predict 11 cricket players.

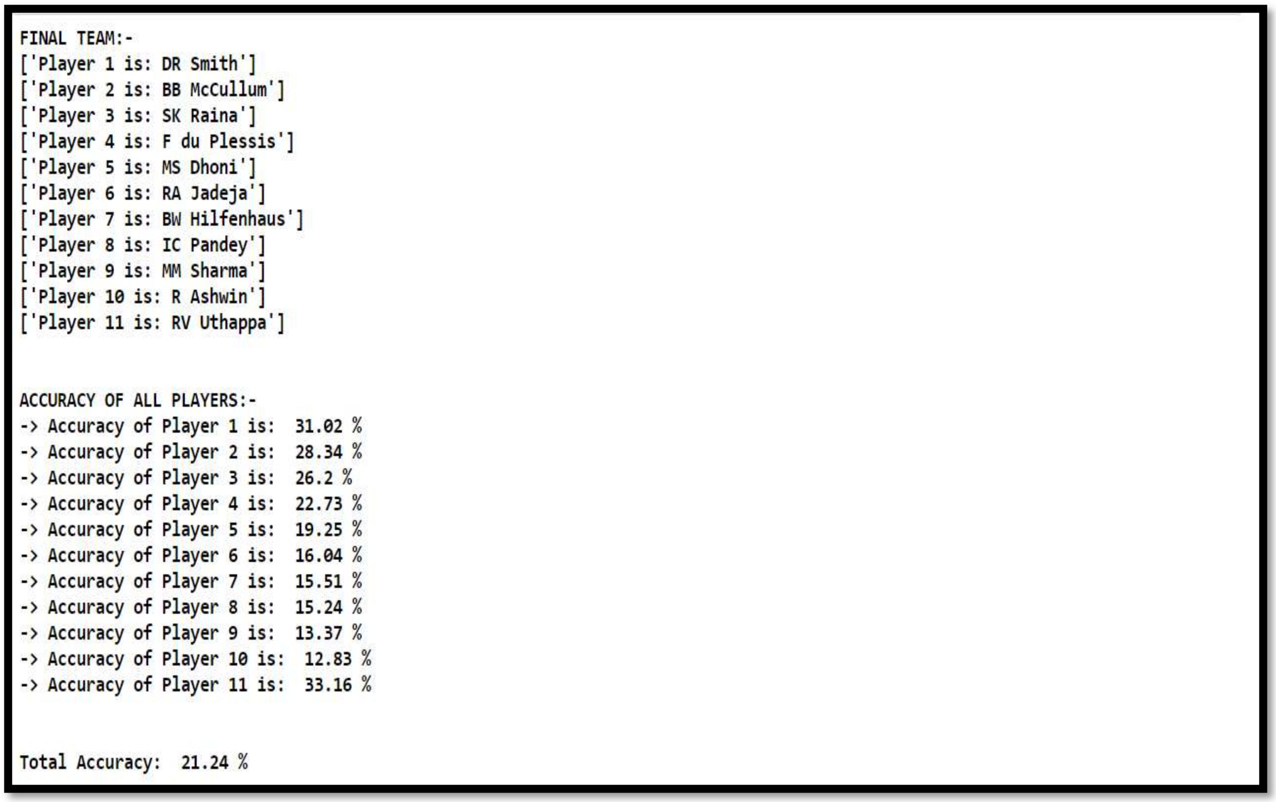
The results obtained by following the various processes that come under machine learning. Data preprocessing includes dropping some rows in which cases come under more than 10 rows and also Converting string variables into numerical. After completing the data processing work, the dataset was divided into testing and training sets.

Combined teams predicted is:-

FINAL TEAM:-	ACCURACY OF ALL PLAYERS:-
['Player 1 is: DR Smith']	**-> Accuracy of Player 1 is: 31.02 %**
['Player 2 is: BB McCullum']	**-> Accuracy of Player 2 is: 28.34 %**
['Player 3 is: SK Raina']	**-> Accuracy of Player 3 is: 26.2 %**
['Player 4 is: F du Plessis']	**-> Accuracy of Player 4 is: 22.73 %**
['Player 5 is: MS Dhoni']	**-> Accuracy of Player 5 is: 19.25 %**
['Player 6 is: RA Jadeja']	**-> Accuracy of Player 6 is: 16.04 %**
['Player 7 is: BW Hilfenhaus']	**-> Accuracy of Player 7 is: 15.51 %**
['Player 8 is: IC Pandey']	**-> Accuracy of Player 8 is: 15.24 %**
['Player 9 is: MM Sharma']	**-> Accuracy of Player 9 is: 13.37 %**
['Player 10 is: R Ashwin']	**-> Accuracy of Player 10 is: 12.83 %**
['Player 11 is: RV Uthappa']	**-> Accuracy of Player 11 is: 33.16 %**
Total Accuracy: 21.24 %	



```
print("\033[1m" +"FINAL TEAM:-")
print("Player 1 is: "+p1)
print("Player 2 is: "+p2)
print("Player 3 is: "+p3)
print("Player 4 is: "+p4)
print("Player 5 is: "+p5)
print("Player 6 is: "+p6)
print("Player 7 is: "+p7)
print("Player 8 is: "+p8)
print("Player 9 is: "+p9)
print("Player 10 is: "+p10)
print("Player 11 is: "+p11)
print("\n")
print("\033[1m" +"ACCURACY OF ALL PLAYERS:-")
print("-> Accuracy of Player 1 is: ",a1,"%")
print("-> Accuracy of Player 2 is: ",a2,"%")
print("-> Accuracy of Player 3 is: ",a3,"%")
print("-> Accuracy of Player 4 is: ",a4,"%")
print("-> Accuracy of Player 5 is: ",a5,"%")
print("-> Accuracy of Player 6 is: ",a6,"%")
print("-> Accuracy of Player 7 is: ",a7,"%")
print("-> Accuracy of Player 8 is: ",a8,"%")
print("-> Accuracy of Player 9 is: ",a9,"%")
print("-> Accuracy of Player 10 is: ",a10,"%")
print("-> Accuracy of Player 11 is: ",a11,"%")
print("\n")
at= (a1+a2+a3+a4+a5+a6+a7+a8+a9+a10+a11)/11
at=round(at,2)
print("Total Accuracy: ",at,"%")
```



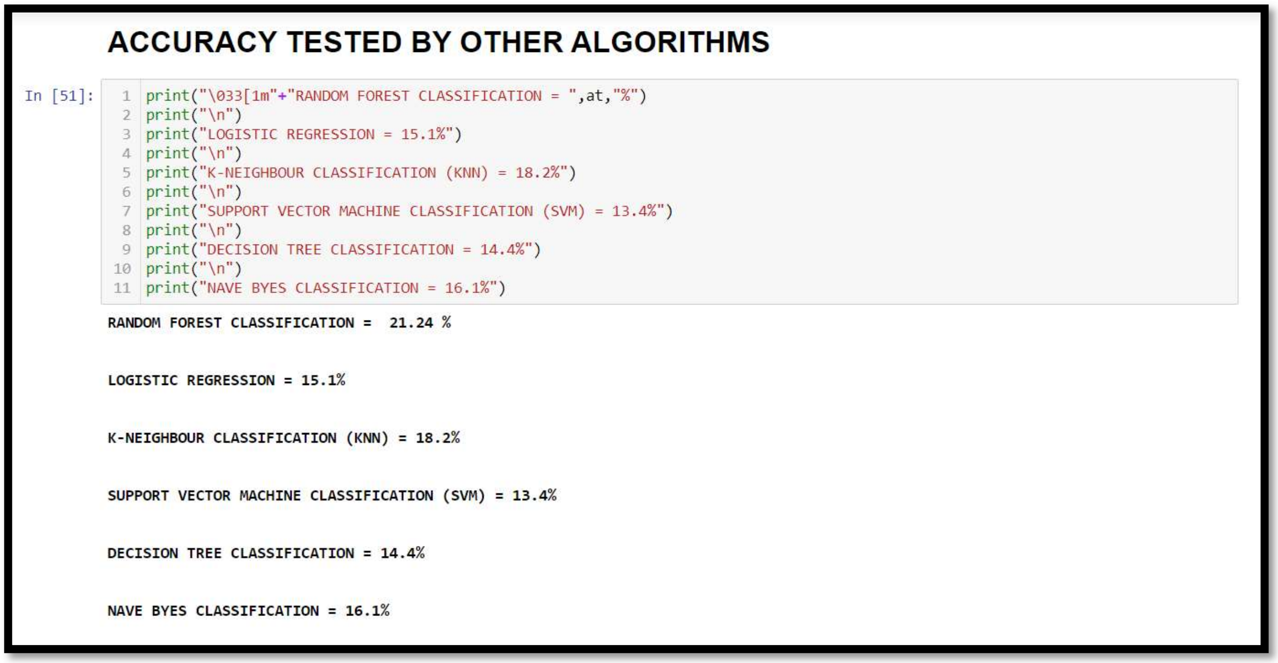
```
FINAL TEAM:-
['Player 1 is: DR Smith']
['Player 2 is: BB McCullum']
['Player 3 is: SK Raina']
['Player 4 is: F du Plessis']
['Player 5 is: MS Dhoni']
['Player 6 is: RA Jadeja']
['Player 7 is: BW Hilfenhaus']
['Player 8 is: IC Pandey']
['Player 9 is: MM Sharma']
['Player 10 is: R Ashwin']
['Player 11 is: RV Uthappa']


ACCURACY OF ALL PLAYERS:-
-> Accuracy of Player 1 is:  31.02 %
-> Accuracy of Player 2 is:  28.34 %
-> Accuracy of Player 3 is:  26.2 %
-> Accuracy of Player 4 is:  22.73 %
-> Accuracy of Player 5 is:  19.25 %
-> Accuracy of Player 6 is:  16.04 %
-> Accuracy of Player 7 is:  15.51 %
-> Accuracy of Player 8 is:  15.24 %
-> Accuracy of Player 9 is:  13.37 %
-> Accuracy of Player 10 is:  12.83 %
-> Accuracy of Player 11 is:  33.16 %


Total Accuracy:  21.24 %
```

Fig 10.1 Combined team with average accuracy

The implementation of the algorithm and models is trained according to the classification algorithms. The result of the implementation of algorithms is mentioned in the below table. Sklearn is used for calculating the accuracy by using the function “score_accuracy”. The accuracy of all the algorithms is mentioned in the below table.

An algorithm which has the greatest precision value and also generated the greatest accuracy score among the 7 algorithms considered as the best suitable algorithm for completing the objective of this project.

Please note that the source code contains only prediction by random forest ad all other algorithms were applied during hit and trial for finding algorithm with highest accuracy.



ACCURACY TESTED BY OTHER ALGORITHMS

```
In [51]:
print("\033[1m"+"RANDOM FOREST CLASSIFICATION = ",at,"%")
print("\n")
print("LOGISTIC REGRESSION = 15.1%")
print("\n")
print("K-NEIGHBOUR CLASSIFICATION (KNN) = 18.2%")
print("\n")
print("SUPPORT VECTOR MACHINE CLASSIFICATION (SVM) = 13.4%")
print("\n")
print("DECISION TREE CLASSIFICATION = 14.4%")
print("\n")
print("NAVE BYES CLASSIFICATION = 16.1%")
```

```
RANDOM FOREST CLASSIFICATION =  21.24 %


LOGISTIC REGRESSION = 15.1%


K-NEIGHBOUR CLASSIFICATION (KNN) = 18.2%


SUPPORT VECTOR MACHINE CLASSIFICATION (SVM) = 13.4%


DECISION TREE CLASSIFICATION = 14.4%


NAVE BYES CLASSIFICATION = 16.1%
```

Fig 10.2 Accuracy comparison by all models

CHAPTER 11- FUTURE SCOPE:

1. Data, in this case, is mixed for e.g. , test and limited over statistics are mixed up , so we can classify them and can choose playing 11 for each format.

2. Accuracy of team is quite low which can be improved further by more data cleaning and encoding.

3. We can also predict playing XI for each country.

CHAPTER 12- REFERENCES:

[1] S. Muthuswamy and S. S. Lam, "Bowler Performance Prediction for One-day International Cricket Using Neural Networks", Proceedings of IIE Annual Conference and Expo 2008, pp.1391-1395/

[2] I. P. Wickramasinghe, "Predicting the performance of batsmen in test cricket", Journal of Human Sport & Exercise, vol. 9, no. 4, pp. 744-751, May 2014.

[3] G. D. I. Barr and B. S. Kantor, "A Criterion for Comparing and Selecting Batsmen in Limited Overs Cricket", Operational Research Society, vol. 55, no. 12, pp. 1266-1274, December 2004.