IPL Data Analysis and Visualization for Team Selection and Profit Strategy

Abstract- **Day by day, the role of data science and machine learning in cricket is increasing due to the large amount of data generated from a single player on a whole line. The field of data science is the intensive study of data to extract insights and knowledge from the data and apply the acquired knowledge and actionable insights. We use these available data and statistics to predict things like the team's first-innings score and the probability of winning the second team, etc. In this paper, we will work on Indian Premier League (IPL) Data Analysis 2022 and Data Visualization for the duration (2008-2020) using Python. This application modules are followed by preprocessing, data analysis, and visualization, and finally create a model that predicts the team's overall score and the probability of winning. When building models, we use python algorithms such as Numpy for Scientific Computing, Pandas for Data Analysis, and finally Matplotlib and Seaborn for Data Visualization. Finally, this paper helps the trainers and owners of the team during the auction to select an emerging player to win the matches in the entire season and to win the trophy and to get profit to the owners by the profit strategy done in this analysis report.**

Keywords: Data Analysis, Data Cleaning, Indian Premier League, Prediction, Pandas, Numpy, Matplotlib, Seaborn.

I.INTRODUCTION

Python is a division of artificial intelligence, where real-time problem statements can be solved in the real world. This method is purely done using Python programming using library files such as umpy, Pandas, Matplotlib, Seaborn and depends on learning data from previous data and predicting the result accordingly. Python methods benefit from the use of knowledge acquisition and mathematical models.

Leema Nelson
Department of Computer Science and Engineering,
Chitkara University Institute of Engineering & technology,
Chitkara University,
Punjab, India
leema.nelson@chitkara.edu.in

Today, the demand for cricket is growing rapidly, with many people focusing on data analysis and data prediction through various digital technologies. Predicting and Analyzing the Indian Premier League (IPL) data through Python play a key role in the selection of players. The players are chosen based on various factors.

The cricket board and the mentor to decide the team selection for the Indian Premier League and the captain also has a main role in choosing the team squad on seeing the average scores of the team players against the opponent team players in the previous matches played. So, this paper is based on the victory of individual players of the team based on the median details of the matches played before. The mentor decides on the greatest batting and the finest bowling performances, and the analysis of all-rounder performances. Finally, based on these analyses, 15 players are selected for the Indian Premier League. This conveys the prediction and analysis through python algorithms that are being used in this research project. Thus, these algorithms forecast the performer's moderate score efficiently. The results of the study show that the prediction of the entire team is convincing and realistic.

II.EXISTING SYSTEM

In the existing technique, there is an algorithm to compute the projected score and a winning predictor based on the win percentage of a squad and ballots. These strategies won't give precise outcomes because they are based on projections and perceptions based on a certain instance. Consider computing the tossed score using the traditional algorithm [1]. Projected score = Current Run Rate * Overs remaining in an Innings [2].

The moderate accurateness obtained by the following the above algorithm is very less. Trainers can include good all-rounder performers to improve squad outcomes [3]. Several pictorial illustrations and visual methods were employed to analyze the batting and bowling performances of the cricketers [4]. Then For the computation and interpretation of the charts, the work examined the batsman and Bowler's history from the 2008 season [5]. Nominated were 12 bowlers and 12 batsmen who bowled at least 15 overs and took at least 5 wickets, and batsmen who faced at least 20 overs and had at least 5 completed innings [6].

The analysis shows that Indian bowlers performed well, with Indians ranking among the top ten bowlers in all four seasons (2009, 2011, 2015, and 2019) [7]. This algorithm-based approach is used to huddle the players by position and grade the players' performance [8]. Based on performance, players from the 2008 IPL season were selected for creating rank [9]. Based on their bowling and batting performances, players were divided into groups [10]. They assess the performance of individual players from each team [11]. They used Python algorithms to simulate batsmen and bowlers' execution based on past and current career data [12]. The experiment is carried out by using a bowling

average, strike rate, and economy, all of which are referred to as the Combined Bowling Rate [13].

A statistical model is developed to estimate a player's value by taking into account various statistics of batsmen, bowlers, and all-rounders [14]. The work is done to evaluate bowlers' performance [15]. They attempted to develop a systematic analytical regression model to select better auction participants [16]. In this paper, a multiple criteria decision-making evolutionary method was used to optimise a squad's batting and bowling stability and to find group mates. An algorithm is used to evaluate each player's performance [17].

III. Proposed System

In our proposed system, we have used python libraries like Pandas, NumPy, Matplotlib, and Seaborn for Data Analyzing and Data Visualization. Firstly, we will clean the given set of incompetent match data into a functional form and eliminate the redundancies from the dataset like inappropriate team titles, teams with distinct spellings by the strategy of Data Cleaning after that, we will use the above-mentioned software to explore various fields of match by acquiring the statistics from which we can analyze some useful and fascinating outcomes.

A. Tools and Methodology

The Indian Premier League is one of the most popular T20 leagues in the world, with millions of fans worldwide. From 2008 to 2020, approximately 816 matches were played. There is a massive amount of data that includes ball-by--by-ball acuity for each team played, match innings, the date the match occurred, venue, toss, overs, wickets, boundaries, extras, winner, umpires with match location, and all other necessary details. Jupyter, an Open License and Scientific Python Development Environment that is integrated with Visual Studio Code (VSC) version 1.64.0, was used to conduct data analysis and plotting for visualisation. Jupyter has a large collection of scientific packages for in-depth analysis, as well as the ability to perform data exploration. Jupyter is used to perform Ethical Data Analysis and Visualization using NumPy, Pandas, Matplotlib, and Seaborn. These Python library packages aid in basic and modern visualisation. This paper proposes a system development approach that employs NumPy and Pandas for data analysis, as well as Seaborn and Matplotlib for performer visualisation.

B. Data Collection

The exploratory data has been gathered from various website sources such as www.kaggle.com,www.data.world/datasets/ipl, dataset has information pertaining to all matches played from 2008 to 2020. It has 17 attributes and 816 entities. Each row corresponds to the information of a unique match. The vital data attributes included are the city, date, venue, best performer of the match, team played, winning the toss, decision made on winning the toss, the team which won the match, final result, super , umpire's details of the match.

C. Importing IPL Dataset

In importing the dataset, we have our data in the form of excel which has the matches which were played from 2008 till 2020. It is shown in Fig. 1. First of all, Pandas is a software library used in Python for Data Manipulation and Data Analysis. The Panda is furthermore a free software that was released under the three-clause BSD license. We have used Panda library for importing the details of the matches played from 2008 to 2020 into the Jupyter Platform under Visual Studio Code (VSC) for analyzing the data and it is also used for printing the analysed data using the head function in Python. In this, the pandas will use the read.csv command along with the pathway of the dataset to the Jupyter platform which we have already connected to Jupyter Server and using it in the Visual Studio Code (VSC) platform.

index	id	city	date	player…
0	335982	Bangalore	18-04-20…	BB McCul…
1	335983	Chandiga…	19-04-20…	MEK Huss…
2	335984	Delhi	19-04-20…	MF Mahar…
3	335985	Mumbai	20-04-20…	MV Bouch…
4	335986	Kolkata	20-04-20…	DJ Hussey
5	335987	Jaipur	21-04-20…	SR Watson
6	335988	Hyderabad	22-04-20…	V Sehwag
7	335989	Chennai	23-04-20…	ML Hayden
8	335990	Hyderabad	24-04-20…	YK Pathan
9	335991	Chandiga…	25-04-20…	KC Sanga…
10	335992	Bangalore	26-04-20…	SR Watson
11	335993	Chennai	26-04-20…	JDP Oram

Fig. 1. Importing IPL Data from Raw Data Set

D. Checking Dataset for NULL Values

Before starting the process of IPL Data Analysis, we need to check the columns, rows, and NULL values present in the dataset so that we can conclude the analyzing phase by removing the NULL Values present in it. For this analyzing phase, we use the Panda "match_data.isnull().sum()" function, where the "isnull" is used to check the number of NULL Values in a column and the "sum" is used to add the total number of NULL Values present in a column. It is shown in Fig. 2.

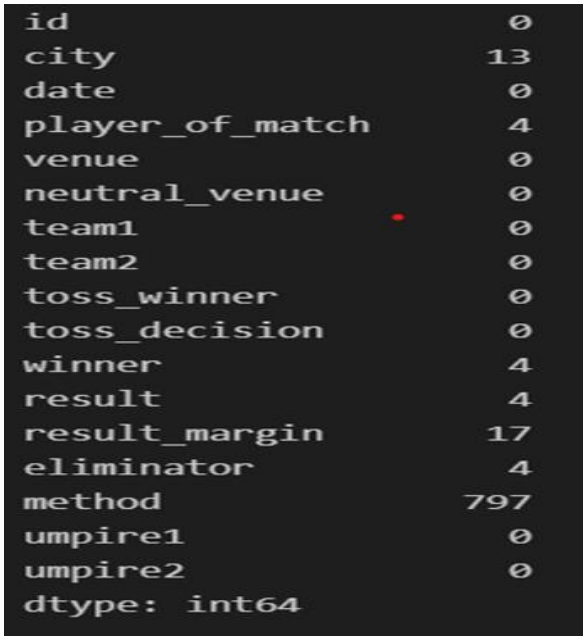


Fig. 2. Checking NULL Values from the Data

IV. Analysis And Visualization

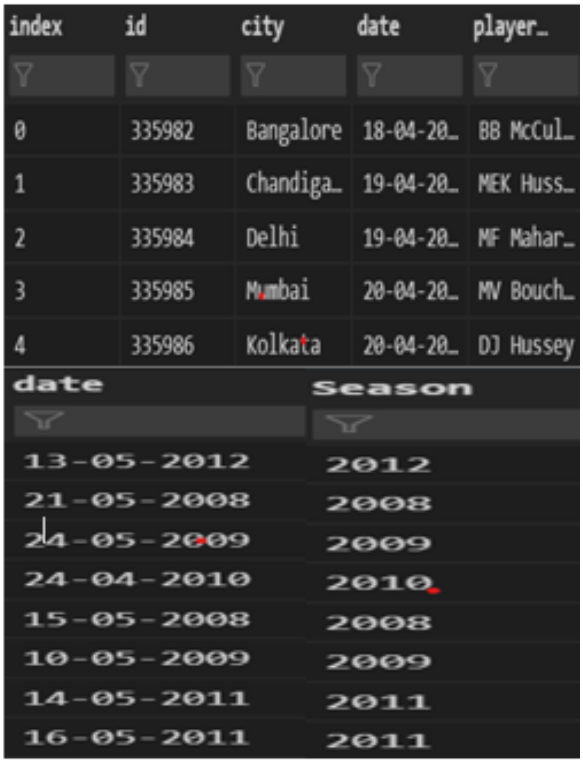
With all basic understanding of the attributes present in Python, we will start the project of analyzing the IPL Dataset and visualize it in various forms such as bar chart, pie chart, bar graph, and line graph. In this, we can also extract particular data from the dataset and create a new dataset with the specific columns and rows we require.

1. List of Extracted Columns

In the given data set, we use the panda "match_data.columns" command to extract the particular column names from the entire data set and visualize them in the form of object data types [18]. Data set collects from https://www.kaggle.com/rishpande/indian-premier-league-ipl-data-visualization.

2. Extraction of Season Based on Date Column

For this, we will be creating a new column named "Season" in the excel data set. We use the "DatetimeIndex" array, which is a part of Pandas for separating the year from the date column. It is a process similar to the data extraction and visualizing it as the outcome. We have extracted this year from the date column and named it Season because we will use it in the process for visualizing purposes. It is shown in Fig. 3.



index	id	city	date	player…
0	335982	Bangalore	18-04-20…	BB McCul…
1	335983	Chandiga…	19-04-20…	MEK Huss…
2	335984	Delhi	19-04-20…	MF Mahar…
3	335985	Mumbai	20-04-20…	MV Bouch…
4	335986	Kolkata	20-04-20…	DJ Hussey

date	Season
13-05-2012	2012
21-05-2008	2008
24-05-2009	2009
24-04-2010	2010
15-05-2008	2008
10-05-2009	2009
14-05-2011	2011
16-05-2011	2011

Fig. 3. Extract Seasons based on Date

3. Analyzing the Total Matches Played in Each Particular Season

Here we have created a new variable named "match_season_data", where we have used the Panda "group by" library and we have also grouped the season columns using the unique id given to each row of the data set. After this analyzing phase, we have used the "count" function to count the total number of matches played in every season from 2008 to 2020. After this, we have renamed the name of the "id" to "total count" which will be displayed next to the Season column. It is shown in Fig. 4.

	Season	total count
0	2008	58
1	2009	57
2	2010	60
3	2011	73
4	2012	74
5	2013	76
6	2014	60
7	2015	59
8	2016	60
9	2017	59
10	2018	60
11	2019	60
12	2020	60

Fig. 4. Analyzing Total Matches Played

4. Merging Match Data Set with the Ball Data Set using Right Join

In this process, firstly we will create a new variable named "season_data" and we will merge both the match data set and ball data set. The "id" and "season" column from the match data is merged with ball data by assigning the left column for match data's "id" and the right column for ball data's "id" and this merging is done using "Left Join". Then the panda "season_data.head" function is used to print the outcome. It is shown in Fig. 5.

Season	inning	over	ball
2008	1	6	5
2008	1	6	6
2008	1	7	1
2008	1	7	2
2008	1	7	3

Fig. 5. Merge Ball Data with Season Data

5. Line Graph Visualization for the Total Number of Runs Scored in the Entire IPL Season

Here we will use the above data which is being extracted from the updated ball data which is nothing but the season data. Initially, we have calculated the total number of runs scored per season by using the Python "sum" function. Then we have set a variable "p" to set the index of the "Season". Then "ax" variable is allotted to plot the line graph using the matplotlib library. Then we have used some styling factors like setting the face color of the graph using "ax.set" command. Finally, we have used the Seaborn "sns.lineplot" function to draw a line graph using the data which is imported using the "p" variable. The palette color is then set to "magma". Then we use the matplotlib "plot.title" to set the title of the line graph and, we have set the font size and font weight that needs to be printed as output. At last, we have used the matplotlib "plt.show()" function to visualize the line graph. It is shown in Fig. 6.

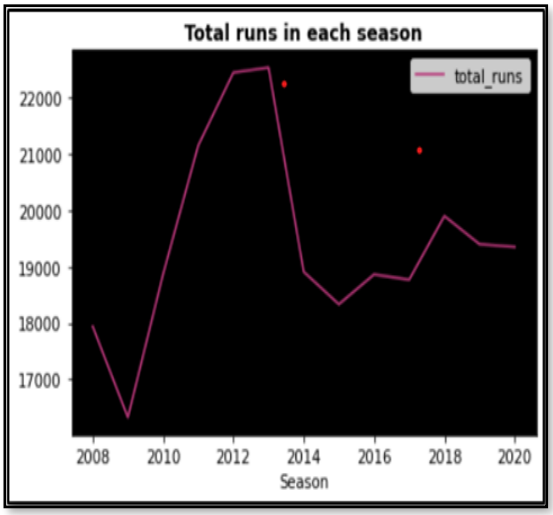


Fig. 6. Total Runs Scored in Entire Season

6. Bar Graph Visualization for the Total Matches Played in IPL Seasons

In this, we have used the Seaborn "sns.countplot" function to visualize the "Season" column from the match data set. Then we use the matplotlib "plt.xticks" and "plt.yticks" function to set the attributes such as rotation and font size of the x-axis and y-axis of the bar graph. We also use "plt.xlabel" and "plt.ylabel" matplotlib functions to set the name of the x-axis as "Season" and y-axis as "Count" respectively. Finally, we used the "plt.title" function to assign the title as "Total Matches played in a season" in the bar graph. It is shown in Fig. 7.

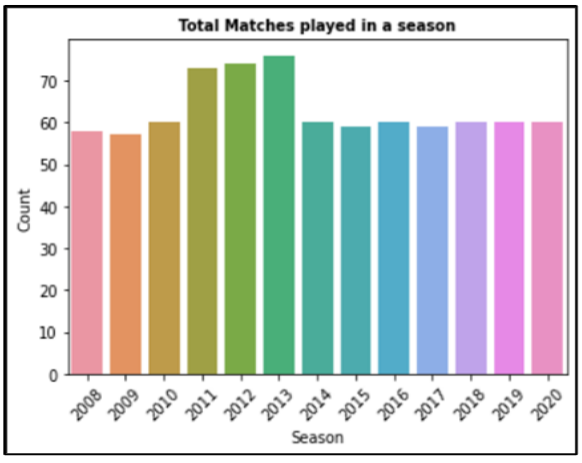


Fig. 7. Total Matches Played in Entire Season

7. Bar Plot Showing the Number of Toss Won by Each Team

We will be analyzing the match data and extracting the "toss win" column and will use the "value_counts()" function to get the total number of tosses won by each team. In this, we will be using the same procedure which we have implemented for the bar graph visualization. Along with this, we have used the "sns.barplot" function to set the position of the bar graph and to assign the color settings like the palette, saturation, etc. It is shown in Fig. 8.

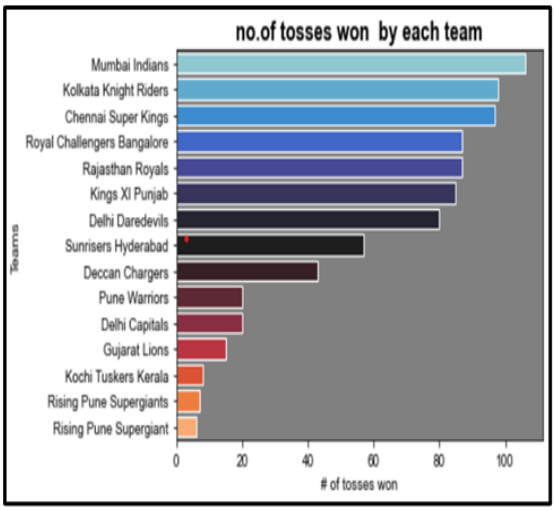


Fig. 8. Total No. of Toss won by Each Team

8. Extraction of a Particular Data from Data Sheet

In this, we will be using the Panda library to pull a particular set of data from the complete data set. Firstly, we have created a variable named "player", in which a particular data is imported from the ball data set. For Instance, {player=(ball_data['batsman'] =='SK Raina'} In this command a typical batsman is picked from the entire data set and extracted to visualize the outcome. In this, we have also assigned a new variable named "df_raina" to extract the specific player details and to visualize. It is shown in Fig. 9.

inning	over	ball	batsman	non_striker	bowler	batsman_runs
1	10	3	SK Raina	MEK Hussey	PP Chawla	2
1	10	4	SK Raina	MEK Hussey	PP Chawla	0
1	10	5	SK Raina	MEK Hussey	PP Chawla	6
1	10	6	SK Raina	MEK Hussey	PP Chawla	4
1	11	4	SK Raina	MEK Hussey	K Goel	6

Fig. 9. Extraction of Particular Data

9. Visualization of Dismissal Kind in the form of Pie Chart

In this step, we have extracted the specific column "dismissal_kind" inside the variable "df_raina".

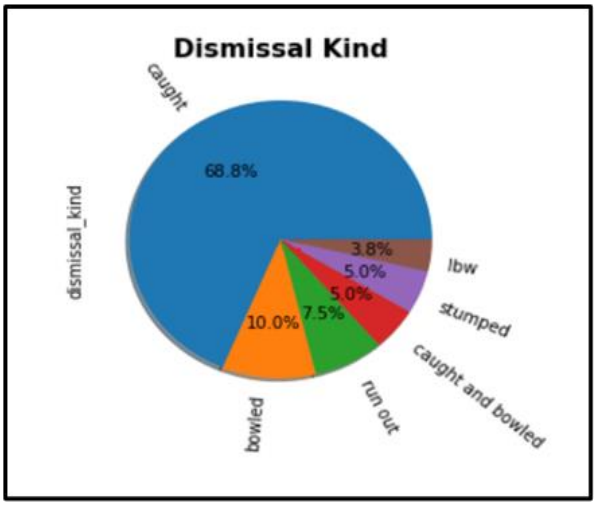


Fig. 10. Dismissal Kind of Data

After this, we will be counting the total number of dismissal kinds available in the specific column of the entire IPL season data. Then we will convert the same into percentage values so that they can be used for the pictorial representation of the pie chart in Fig. 10.

10. Data Extraction for the Runs Scored by Unspecific Player

In this process, we will initially define a Python function "def.count(df_raina,runs)". This function is defined to extract and deliver a detailed score point of a specific player in the entire duration of the season from 2008 to 2020. We will use the "print" statement to produce the sum of the score points in a detailed manner like the number of singles, doubles, 3's, fours,5's, and sixes taken by that specific player by using the "df_raina" data set. It is shown in Fig. 10. Exploratory Data Evaluation and Envisage the Top 10 Run Scorers in the IPL Season using the same procedure which we have done above to extract the data. The extracted data is then visualized in the form of a bar chart. Along with this, we have used the "sns.barplot" function to set the position of the bar graph and to assign the color settings like palette, saturation, rotation, font size, and font weight. It is shown in Fig. 11.

```
Runs scored from 1's : 1666
Runs scored from 2's : 528
Runs scored from 3's : 33
Runs scored from 4's : 1972
Runs scored from 6's : 1164
```

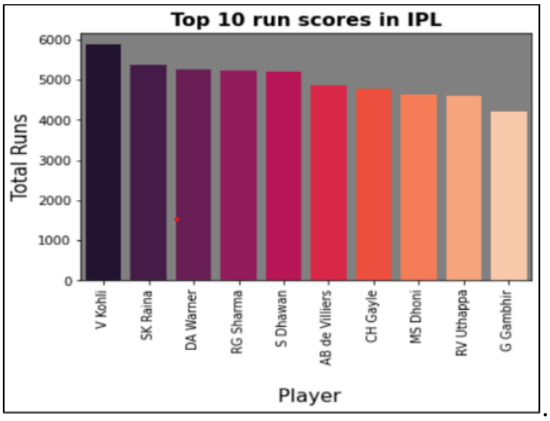
Fig. 11. Runs Scored by Unspecific Player

In this step, we have created a variable named “run” and have imported the data from the ball data set. Once we import the data, we use the “group by” function to group all details of the batsmen and the sum of the total runs scored by them. We assign the column names by using the “runs.columns” function. Finally, we use the “runs.sort_values” function to sort the group of the batsmen and the total sum of their scores. We have specified some conditions such as “sort by run column” and the sorting should be in “descending order" so that we can get the top 10 batsmen who scored the maximum runs in the entire IPL season. It is shown in Fig. 12.

	batsman	runs
0	V Kohli	5878
1	SK Raina	5368
2	DA Warner	5254
3	RG Sharma	5230
4	S Dhawan	5197
5	AB de Villiers	4849
6	CH Gayle	4772
7	MS Dhoni	4632
8	RV Uthappa	4607
9	G Gambhir	4217

Fig. 12. Envisage of Top 10 Runs Scorer

Envisage the Top 10 Run Scorers in the IPL Season in the form of Bar Chart. We will be analyzing the batsman data and extracting the "runs" column and “batsman” column and will use the "ax” function to get the total runs scored by each team. It is shown in Fig. 13.



Kumar, H. Sharma and R. Pal, “Popularity Measuring and

Fig. 13. Top Run Scorer in the Season

Visualization of the Highest Number of Mom Award Winners in the form of Bar Chart. In this process, we have analysed and extracted the “playerofthematch” column and have counted the total in the entire season. After this, we have visualized it in the form of a bar graph with the highest number of “players of the match” to be displayed at first and it should continue in the decreasing order. We have assigned the bar graph with a title “Highest MOM award winner” and the x and y axes are named as “Players” and “Count” respectively. It is shown in Fig. 14.

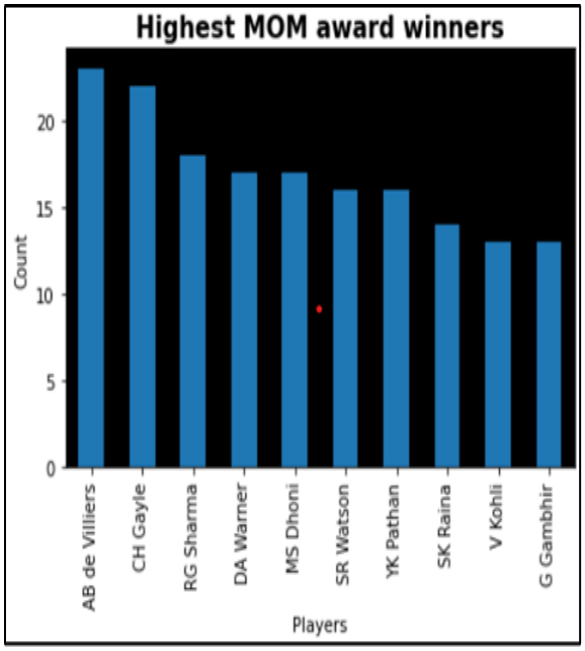


Fig. 14. Highest MOM Award Winners

V. Conclusion

This report has been implemented to investigate the results of IPL matches from 2008 to 2020 using Python algorithms on both proportional and uneven datasets. The benchmark which is used to scrutinize the results of matches was constructed successfully with a precision rate of 94% for the congruous dataset using the classifiers, which is after resampling the imbalanced IPL dataset. This report emphasises player performance, particularly batsmen, and addresses the study that is done for the maximum number of Men of the Match, Leading Batsmen, and top 10 performers on the Most Runs. Statistics from approximately 816 matches were used in this investigation, as well as toss-related breakdowns such as the total number of toss wins, judgments made by each squad after winning the toss, and toss decisions made by each squad throughout the season.