A

SCE / PBL

On

“(IPL Data Analysis)”

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

INFORMATION TECHNOLOGY DEPARTMENT

BY:

Sr No.	Roll no.	Name	PRN
1	231027	Sagar Jadhav	22010365
2	231032	Satyam kangle	22010972

Division: A **Sem -IV** **Batch: A-2**

Sub: ITW-Python

Introduction :-

The use of analytical methods in various aspects of cricket such as Batting, Bowling, Fielding, Team Selection, Result Prediction, Target Revision in a rain affected match are very important. There is a huge demand of various algorithms for each and every aspect of cricket because of its popularity and the huge amounts of money involved in the game.

In India the followers of cricket are also followers of statistical records. Thus, the analysis of a league like IPL becomes more important .

Prediction of outcome of matches using algorithm approaches is one of the very important aspects in cricket. No comprehensive attempt has been made in the literature to this end. This is an important problem because franchises invest huge amounts of money.

With the help of a coin toss one can predict with 50% probability the winning team treating the game as one of perfect chance without even considering the relative merits of the respective teams playing against each other. In contrast to this records of past performance of players, their current form and other cricket related data can be analyzed statistically to create mathematical models with better probability of success in predicting the winning team.

Data description:

In this project we use different type of the parameter related to IPL team's information

Name of players, Team, Nationality, Player Type, Capped, Matches Played, Runs Average, Strike Rate, Wickets, Bowling average, Economy, Bowling Strike Rate, Catches, Run outs, Stump outs.

Code:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

%matplotlib inline

import warnings
warnings.filterwarnings('ignore')

ipl=pd.read_csv('IPLData.csv')
ipl1=pd.read_csv('IPLData1.csv')

# prints the first five lines of code
ipl1.head()
```

Output:

|:

	Player Name	Team	Nationality	Player_Type	Capped	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_R
0	Shikhar Dhawan	Punjab	Indian	Batter	1	192.0	5783.0	34.63	126.60	4.0	16.5	8.25	1
1	Shreyas Iyer	Kolkata	Indian	Batter	1	87.0	2375.0	31.67	123.96	NaN	NaN	NaN	N
2	Faf Du Plessis	Bangalore	South Africa	Batter	1	100.0	2935.0	34.94	131.09	NaN	NaN	NaN	N
3	Manish Pandey	Lucknow	Indian	Batter	1	154.0	3560.0	30.69	121.83	NaN	NaN	NaN	N
4	Shimron Hetmyer	Rajasthan	West Indies	Batter	1	31.0	517.0	25.85	151.17	NaN	NaN	NaN	N

Code:

```
# display the first five lines
ipl.head()
```

Output:

	Player Name	Team	Nationality	Player_Type	Capped	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Ra
0	Shikhar Dhawan	Punjab	Indian	Batter	1	192.0	5783.0	34.63	126.60	4.0	16.5	8.25	1
1	Shreyas Iyer	Kolkata	Indian	Batter	1	87.0	2375.0	31.67	123.96	NaN	NaN	NaN	N
2	Faf Du Plessis	Bangalore	Overseas	Batter	1	100.0	2935.0	34.94	131.09	NaN	NaN	NaN	N
3	Manish Pandey	Lucknow	Indian	Batter	1	154.0	3560.0	30.69	121.83	NaN	NaN	NaN	N
4	Shimron Hetmyer	Rajasthan	Overseas	Batter	1	31.0	517.0	25.85	151.17	NaN	NaN	NaN	N

Code:

```
import pandas as pd
# Gives the information about following parameters
ipl.describe()
```

Output:

	Capped	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate	Catches	Run_outs
count	235.000000	215.000000	165.000000	161.000000	163.000000	140.000000	135.000000	143.000000	119.000000	27.000000	27.000000
mean	0.838298	43.897674	840.575758	21.792391	121.009939	31.485714	32.907185	8.223182	24.686134	30.962963	3.444444
std	0.561802	48.695302	1270.341831	11.664156	30.739189	36.872420	18.191441	1.223541	12.982049	34.544822	5.010246
min	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	5.360000	0.000000	0.000000	0.000000
25%	0.500000	11.500000	67.000000	13.800000	112.635000	6.000000	23.025000	7.390000	18.495000	3.500000	0.000000
50%	1.000000	25.000000	289.000000	22.410000	128.630000	19.500000	29.070000	8.190000	21.750000	19.000000	1.000000
75%	1.000000	56.000000	954.000000	29.300000	137.550000	40.500000	36.030000	8.785000	26.190000	51.500000	4.000000
max	2.000000	220.000000	6283.000000	58.500000	190.240000	167.000000	153.000000	13.120000	108.000000	126.000000	21.000000

Code:

```
#check the number of null values in each column
ipl.isna().sum()
```

Output:

```
Player Name                0
Team                       0
Nationality                0
Player_Type                0
Capped                     0
Matches_Played            20
Runs                      70
Average                   74
Strike_Rate               72
Wickets                   95
Bowling_average          100
Economy                   92
Bowling_Strike_Rate      116
Catches                  208
Run_outs                 208
Stumps                   208
dtype: int64
```

Code:

```
# no of non-nullvalues
ipl.info()
#retun information about the data set
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 235 entries, 0 to 234
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Player Name           235 non-null    object
 1   Team                  235 non-null    object
 2   Nationality           235 non-null    object
 3   Player_Type           235 non-null    object
 4   Capped                235 non-null    int64
 5   Matches_Played        215 non-null    float64
 6   Runs                  165 non-null    float64
 7   Average               161 non-null    float64
 8   Strike_Rate           163 non-null    float64
 9   Wickets               140 non-null    float64
 10  Bowling_average       135 non-null    float64
 11  Economy               143 non-null    float64
 12  Bowling_Strike_Rate   119 non-null    float64
 13  Catches               27 non-null     float64
 14  Run_outs              27 non-null     float64
 15  Stumps                27 non-null     float64
dtypes: float64(11), int64(1), object(4)
memory usage: 29.5+ KB
```

Code:

```
#1.Capped players- batters,bowlers,all-rounder,wicket-keeper
#2.Uncapped players- batters,bowlers,all-rounder,wicket-keeper

# seperating capped batters

batters = ipl.loc[(ipl["Player_Type"] == "Batter")]
batters_new = batters.loc[(batters["Capped"] == 1)]

Capped_Batters = batters_new[['Player
Name','Team','Nationality','Matches_Played','Runs','Average','Strike_Rate'
]]
Capped_Batters.head()
```

Output:

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate
0	Shikhar Dhawan	Punjab	Indian	192.0	5783.0	34.63	126.60
1	Shreyas Iyer	Kolkata	Indian	87.0	2375.0	31.67	123.96
2	Faf Du Plessis	Bangalore	Overseas	100.0	2935.0	34.94	131.09
3	Manish Pandey	Lucknow	Indian	154.0	3560.0	30.69	121.83
4	Shimron Hetmyer	Rajasthan	Overseas	31.0	517.0	25.85	151.17

Code:

```
# capped keepers

keepers = ipl.loc[(ipl["Player_Type"] == "Keeper")]
keepers_new = keepers.loc[(keepers["Capped"] == 1)]

Capped_keepers = keepers_new[['Player
Name','Team','Nationality','Matches_Played','Runs','Average','Strike_Rate'
,'Catches',
                              'Run_outs','Stumps']]
Capped_keepers.head()
```

Output:

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.3	130.9	53.0	0.0	14.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.4	127.5	58.0	12.0	2.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.5	136.3	19.0	1.0	2.0
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.5	142.2	18.0	1.0	4.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.8	129.7	123.0	14.0	32.0

Code:

```
# capped all-rounders

Allrounders = ipl.loc[(ipl["Player_Type"] == "Allrounder")]
Allrounders_new = Allrounders.loc[(Allrounders["Capped"] == 1)]

Capped_Allrounders = Allrounders_new[['Player Name',
                                      'Team',
                                      'Nationality',
                                      'Matches_Played',
                                      'Runs',
                                      'Average',
                                      'Strike_Rate',
                                      'Wickets',
                                      'Bowling_average',
                                      'Economy',
                                      'Bowling_Strike_Rate']]
Capped_Allrounders.head()
```

Output:

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
127	Ravichandran Ashwin	Rajasthan	Indian	167.0	456.0	11.12	109.88	145.0	27.80	6.91	24.12
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95
129	Dwayne Bravo	Chennai	Overseas	151.0	1537.0	22.94	130.25	167.0	24.32	8.36	17.44
130	Nitish Rana	Kolkata	Indian	77.0	1820.0	28.00	132.46	7.0	22.00	8.03	16.43
131	Jason Holder	Lucknow	Overseas	26.0	189.0	14.54	121.15	35.0	22.46	8.20	16.43

Code:

```
import pandas as pd
#cleaning the data by making null value by 0

Capped_Batters = Capped_Batters.fillna(0)
Capped_Bowlers = Capped_Bowlers.fillna(0)
Capped_Allrounders = Capped_Allrounders.fillna(0)
Capped_keepers = Capped_keepers.fillna(0)

#checking null values in data

print(Capped_Batters.isna().sum())
print(Capped_Bowlers.isna().sum())
print(Capped_Allrounders.isna().sum())
print(Capped_keepers.isna().sum())
```

Output:

```
Player Name        0
Team               0
Nationality        0
Matches_Played     0
Runs               0
Average            0
Strike_Rate        0
dtype: int64
Player Name             0.0
Team                    0.0
Nationality             0.0
Matches_Played          0.0
Wickets                 0.0
Bowling_average         0.0
Economy                 0.0
Bowling_Strike_Rate     0.0
dtype: float64
Player Name             0
Team                    0
Nationality             0
Matches_Played          0
Runs                    0
Average                 0
Strike_Rate             0
Wickets                 0
Bowling_average         0
Economy                 0
Bowling_Strike_Rate     0
dtype: int64
Player Name        0
Team               0
Nationality        0
Matches_Played     0
Runs               0
Average            0
Strike_Rate        0
```

Code:

```
# analyzing the batters data
#batting average more than 32.0

top_batters = Capped_Batters.loc[(Capped_Batters["Average"] >= 32.0)]

#sorting the in descending order
top_batters_average = top_batters.sort_values('Average',ascending=False)
top_batters_strike_rate =
top_batters.sort_values('Strike_Rate',ascending=False)
top_batters_runs = top_batters.sort_values('Runs',ascending=False)
top_batters_matches =
top_batters.sort_values('Matches_Played',ascending=False)

# data of each of the batters in descending order
top_batters_average
# strike rate in descending order
top_batters_strike_rate
#runs scored in descending order
top_batters_runs
#matches played
top_batters_matches
```

#from our analysis,if we rank 0 to 10, the top three batters that will come while analysiseach of the above data are:

 # 1.David Warner

 # 2.KL Rahul

 # 3.Virat Kohli

Output:

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate
212	KL Rahul	Lucknow	Indian	94.0	3273.0	47.43	136.38
231	Ruturaj Gaikwad	Chennai	Indian	22.0	839.0	46.61	132.13
19	David Warner	Delhi	Overseas	150.0	5449.0	41.60	139.97
207	Kane Williamson	Hyderabad	Overseas	63.0	1885.0	40.11	131.27
208	Virat Kohli	Bangalore	Indian	207.0	6283.0	37.40	129.95
2	Faf Du Plessis	Bangalore	Overseas	100.0	2935.0	34.94	131.09
0	Shikhar Dhawan	Punjab	Indian	192.0	5783.0	34.63	126.60
26	David Miller	Gujarat	Overseas	89.0	1974.0	32.90	136.51

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate
19	David Warner	Delhi	Overseas	150.0	5449.0	41.60	139.97
26	David Miller	Gujarat	Overseas	89.0	1974.0	32.90	136.51
212	KL Rahul	Lucknow	Indian	94.0	3273.0	47.43	136.38
231	Ruturaj Gaikwad	Chennai	Indian	22.0	839.0	46.61	132.13
207	Kane Williamson	Hyderabad	Overseas	63.0	1885.0	40.11	131.27
2	Faf Du Plessis	Bangalore	Overseas	100.0	2935.0	34.94	131.09
208	Virat Kohli	Bangalore	Indian	207.0	6283.0	37.40	129.95
0	Shikhar Dhawan	Punjab	Indian	192.0	5783.0	34.63	126.60

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate
208	Virat Kohli	Bangalore	Indian	207.0	6283.0	37.40	129.95
0	Shikhar Dhawan	Punjab	Indian	192.0	5783.0	34.63	126.60
19	David Warner	Delhi	Overseas	150.0	5449.0	41.60	139.97
212	KL Rahul	Lucknow	Indian	94.0	3273.0	47.43	136.38
2	Faf Du Plessis	Bangalore	Overseas	100.0	2935.0	34.94	131.09
26	David Miller	Gujarat	Overseas	89.0	1974.0	32.90	136.51
207	Kane Williamson	Hyderabad	Overseas	63.0	1885.0	40.11	131.27
231	Ruturaj Gaikwad	Chennai	Indian	22.0	839.0	46.61	132.13

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate
208	Virat Kohli	Bangalore	Indian	207.0	6283.0	37.40	129.95
0	Shikhar Dhawan	Punjab	Indian	192.0	5783.0	34.63	126.60
19	David Warner	Delhi	Overseas	150.0	5449.0	41.60	139.97
2	Faf Du Plessis	Bangalore	Overseas	100.0	2935.0	34.94	131.09
212	KL Rahul	Lucknow	Indian	94.0	3273.0	47.43	136.38
26	David Miller	Gujarat	Overseas	89.0	1974.0	32.90	136.51
207	Kane Williamson	Hyderabad	Overseas	63.0	1885.0	40.11	131.27
231	Ruturaj Gaikwad	Chennai	Indian	22.0	839.0	46.61	132.13

Code:

```
import pandas as pd
#Analyzing the Allrounder Data

#we have marrowed our analysis by further sagregating the allrounders
based on strike rate equal to or more than 148.8.

top_allrounders =
Capped_Allrounders.loc[(Capped_Allrounders["Strike_Rate"] >= 140.0)]

top_allrounders_average = top_allrounders.sort_values('Average',
ascending=False)
top_allrounders_strike_rate = top_allrounders.sort_values('Strike_Rate',
ascending=False)
top_allrounders_runs = top_allrounders.sort_values('Runs',
ascending=False)
top_allrounders_matches = top_allrounders.sort_values('Matches_Played',
ascending=False)
top_allrounders_bowling_average =
top_allrounders.sort_values('Bowling_average')
top_allrounders_bowling_strike_rate =
top_allrounders.sort_values('Bowling_Strike_Rate')
top_allrounders_wickets = top_allrounders.sort_values('Wickets',
ascending=False)
top_allrounders_economy = top_allrounders.sort_values('Economy')

top_allrounders_average
top_allrounders_strike_rate
top_allrounders_runs
top_allrounders_matches
top_allrounders_bowling_average
top_allrounders_bowling_strike_rate
top_allrounders_wickets
top_allrounders_economy
```

 # rank for top allrounders based on above analysis
#1. Andre Russsell
#2. Sunil Narine
#3. Hardik Pandya
#4. Jofra Archer

Outputs:

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
204	Andre Russell	Kolkata	Overseas	84.0	1700.0	29.31	178.57	72.0	26.40	9.05	17.51
154	K Gowtham	Lucknow	Indian	24.0	186.0	14.31	169.09	13.0	43.23	8.26	31.38
232	Sunil Narine	Kolkata	Overseas	134.0	954.0	15.64	161.69	143.0	24.53	6.74	21.83
165	Jofra Archer	Mumbai	Overseas	35.0	195.0	15.00	157.26	46.0	21.33	7.13	17.93
211	Hardik Pandya	Gujarat	Indian	92.0	1476.0	27.33	153.91	42.0	31.26	9.07	20.69
218	Glen Maxwell	Bangalore	Overseas	97.0	2018.0	25.23	151.84	22.0	41.59	8.55	29.18
195	Mohammad Nabi	Kolkata	Overseas	17.0	180.0	15.00	151.26	13.0	31.38	7.14	26.38
233	Kieron Pollard	Mumbai	Overseas	178.0	3268.0	29.98	149.77	65.0	31.62	8.78	21.60
202	Aman Khan	Kolkata	Indian	5.0	40.0	13.30	148.10	0.0	0.00	7.00	0.00
223	Moeen Ali	Chennai	Overseas	34.0	666.0	22.97	146.37	16.0	29.31	6.85	25.69
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
233	Kieron Pollard	Mumbai	Overseas	178.0	3268.0	29.98	149.77	65.0	31.62	8.78	21.60
204	Andre Russell	Kolkata	Overseas	84.0	1700.0	29.31	178.57	72.0	26.40	9.05	17.51
211	Hardik Pandya	Gujarat	Indian	92.0	1476.0	27.33	153.91	42.0	31.26	9.07	20.69
218	Glen Maxwell	Bangalore	Overseas	97.0	2018.0	25.23	151.84	22.0	41.59	8.55	29.18
223	Moeen Ali	Chennai	Overseas	34.0	666.0	22.97	146.37	16.0	29.31	6.85	25.69
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95
232	Sunil Narine	Kolkata	Overseas	134.0	954.0	15.64	161.69	143.0	24.53	6.74	21.83
165	Jofra Archer	Mumbai	Overseas	35.0	195.0	15.00	157.26	46.0	21.33	7.13	17.93
195	Mohammad Nabi	Kolkata	Overseas	17.0	180.0	15.00	151.26	13.0	31.38	7.14	26.38
154	K Gowtham	Lucknow	Indian	24.0	186.0	14.31	169.09	13.0	43.23	8.26	31.38
202	Aman Khan	Kolkata	Indian	5.0	40.0	13.30	148.10	0.0	0.00	7.00	0.00

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
233	Kieron Pollard	Mumbai	Overseas	178.0	3268.0	29.98	149.77	65.0	31.62	8.78	21.60
218	Glen Maxwell	Bangalore	Overseas	97.0	2018.0	25.23	151.84	22.0	41.59	8.55	29.18
204	Andre Russell	Kolkata	Overseas	84.0	1700.0	29.31	178.57	72.0	26.40	9.05	17.51
211	Hardik Pandya	Gujarat	Indian	92.0	1476.0	27.33	153.91	42.0	31.26	9.07	20.69
232	Sunil Narine	Kolkata	Overseas	134.0	954.0	15.64	161.69	143.0	24.53	6.74	21.83
223	Moeen Ali	Chennai	Overseas	34.0	666.0	22.97	146.37	16.0	29.31	6.85	25.69
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95
165	Jofra Archer	Mumbai	Overseas	35.0	195.0	15.00	157.26	46.0	21.33	7.13	17.93
154	K Gowtham	Lucknow	Indian	24.0	186.0	14.31	169.09	13.0	43.23	8.26	31.38
195	Mohammad Nabi	Kolkata	Overseas	17.0	180.0	15.00	151.26	13.0	31.38	7.14	26.38
202	Aman Khan	Kolkata	Indian	5.0	40.0	13.30	148.10	0.0	0.00	7.00	0.00

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
233	Kieron Pollard	Mumbai	Overseas	178.0	3268.0	29.98	149.77	65.0	31.62	8.78	21.60
232	Sunil Narine	Kolkata	Overseas	134.0	954.0	15.64	161.69	143.0	24.53	6.74	21.83
218	Glen Maxwell	Bangalore	Overseas	97.0	2018.0	25.23	151.84	22.0	41.59	8.55	29.18
211	Hardik Pandya	Gujarat	Indian	92.0	1476.0	27.33	153.91	42.0	31.26	9.07	20.69
204	Andre Russell	Kolkata	Overseas	84.0	1700.0	29.31	178.57	72.0	26.40	9.05	17.51
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95
165	Jofra Archer	Mumbai	Overseas	35.0	195.0	15.00	157.26	46.0	21.33	7.13	17.93
223	Moeen Ali	Chennai	Overseas	34.0	666.0	22.97	146.37	16.0	29.31	6.85	25.69
154	K Gowtham	Lucknow	Indian	24.0	186.0	14.31	169.09	13.0	43.23	8.26	31.38
195	Mohammad Nabi	Kolkata	Overseas	17.0	180.0	15.00	151.26	13.0	31.38	7.14	26.38
202	Aman Khan	Kolkata	Indian	5.0	40.0	13.30	148.10	0.0	0.00	7.00	0.00

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
202	Aman Khan	Kolkata	Indian	5.0	40.0	13.30	148.10	0.0	0.00	7.00	0.00
165	Jofra Archer	Mumbai	Overseas	35.0	195.0	15.00	157.26	46.0	21.33	7.13	17.93
232	Sunil Narine	Kolkata	Overseas	134.0	954.0	15.64	161.69	143.0	24.53	6.74	21.83
204	Andre Russell	Kolkata	Overseas	84.0	1700.0	29.31	178.57	72.0	26.40	9.05	17.51
223	Moeen Ali	Chennai	Overseas	34.0	666.0	22.97	146.37	16.0	29.31	6.85	25.69
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95
211	Hardik Pandya	Gujarat	Indian	92.0	1476.0	27.33	153.91	42.0	31.26	9.07	20.69
195	Mohammad Nabi	Kolkata	Overseas	17.0	180.0	15.00	151.26	13.0	31.38	7.14	26.38
233	Kieron Pollard	Mumbai	Overseas	178.0	3268.0	29.98	149.77	65.0	31.62	8.78	21.60
218	Glen Maxwell	Bangalore	Overseas	97.0	2018.0	25.23	151.84	22.0	41.59	8.55	29.18
154	K Gowtham	Lucknow	Indian	24.0	186.0	14.31	169.09	13.0	43.23	8.26	31.38

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
232	Sunil Narine	Kolkata	Overseas	134.0	954.0	15.64	161.69	143.0	24.53	6.74	21.83
204	Andre Russell	Kolkata	Overseas	84.0	1700.0	29.31	178.57	72.0	26.40	9.05	17.51
233	Kieron Pollard	Mumbai	Overseas	178.0	3268.0	29.98	149.77	65.0	31.62	8.78	21.60
165	Jofra Archer	Mumbai	Overseas	35.0	195.0	15.00	157.26	46.0	21.33	7.13	17.93
211	Hardik Pandya	Gujarat	Indian	92.0	1476.0	27.33	153.91	42.0	31.26	9.07	20.69
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95
218	Glen Maxwell	Bangalore	Overseas	97.0	2018.0	25.23	151.84	22.0	41.59	8.55	29.18
223	Moeen Ali	Chennai	Overseas	34.0	666.0	22.97	146.37	16.0	29.31	6.85	25.69
154	K Gowtham	Lucknow	Indian	24.0	186.0	14.31	169.09	13.0	43.23	8.26	31.38
195	Mohammad Nabi	Kolkata	Overseas	17.0	180.0	15.00	151.26	13.0	31.38	7.14	26.38
202	Aman Khan	Kolkata	Indian	5.0	40.0	13.30	148.10	0.0	0.00	7.00	0.00

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Wickets	Bowling_average	Economy	Bowling_Strike_Rate
232	Sunil Narine	Kolkata	Overseas	134.0	954.0	15.64	161.69	143.0	24.53	6.74	21.83
223	Moeen Ali	Chennai	Overseas	34.0	666.0	22.97	146.37	16.0	29.31	6.85	25.69
202	Aman Khan	Kolkata	Indian	5.0	40.0	13.30	148.10	0.0	0.00	7.00	0.00
165	Jofra Archer	Mumbai	Overseas	35.0	195.0	15.00	157.26	46.0	21.33	7.13	17.93
195	Mohammad Nabi	Kolkata	Overseas	17.0	180.0	15.00	151.26	13.0	31.38	7.14	26.38
128	Pat Cummins	Kolkata	Overseas	37.0	316.0	19.75	140.44	38.0	30.13	8.24	21.95
154	K Gowtham	Lucknow	Indian	24.0	186.0	14.31	169.09	13.0	43.23	8.26	31.38
218	Glen Maxwell	Bangalore	Overseas	97.0	2018.0	25.23	151.84	22.0	41.59	8.55	29.18
233	Kieron Pollard	Mumbai	Overseas	178.0	3268.0	29.98	149.77	65.0	31.62	8.78	21.60
204	Andre Russell	Kolkata	Overseas	84.0	1700.0	29.31	178.57	72.0	26.40	9.05	17.51
211	Hardik Pandya	Gujarat	Indian	92.0	1476.0	27.33	153.91	42.0	31.26	9.07	20.69

Code:

```
#Analyzing the Keepers Data

#we have narrowed our analysis by further sagregating the keepers based on
avarage more than 25.0.

top_keepers = Capped_keepers.loc[(Capped_keepers["Average"] >= 25.0)]

top_Keepers_average = top_keepers.sort_values('Average', ascending=False)
top_Keepers_strike_rate = top_keepers.sort_values('Strike_Rate',
ascending=False)
top_Keepers_runs = top_keepers.sort_values('Runs', ascending=False)
top_Keepers_matches = top_keepers.sort_values('Matches_Played',
ascending=False)
top_Keepers_catches = top_keepers.sort_values('Catches',ascending=False)
top_Keepers_runouts = top_keepers.sort_values('Run_outs',ascending=False)
top_Keepers_stumps = top_keepers.sort_values('Stumps', ascending=False)

top_Keepers_average
top_Keepers_strike_rate
top_Keepers_runs
top_Keepers_matches
top_Keepers_catches
top_Keepers_runouts
top_Keepers_stumps

# Rank for keepers from above data
# 1.MS Dhoni
# 2.Dinesh Karthik
# 3.Rishabh Pant
```

Outputs:

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.50	142.20	18.0	1.0	4.0
213	MS Dhoni	Chennai	Indian	220.0	4746.0	39.50	135.80	126.0	21.0	39.0
111	KS Bharat	Delhi	Indian	8.0	191.0	38.20	122.40	4.0	0.0	1.0
206	Rishabh Pant	Delhi	Indian	84.0	2498.0	35.18	147.46	56.0	5.0	14.0
219	Jos Butler	Rajasthan	Overseas	65.0	1968.0	35.14	150.00	34.0	3.0	1.0
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.30	130.90	53.0	0.0	14.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.40	127.50	58.0	12.0	2.0
209	Sanju Samson	Rajasthan	Indian	121.0	3068.0	29.22	134.21	59.0	8.0	10.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.50	136.30	19.0	1.0	2.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.80	129.70	123.0	14.0	32.0

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
219	Jos Butler	Rajasthan	Overseas	65.0	1968.0	35.14	150.00	34.0	3.0	1.0
206	Rishabh Pant	Delhi	Indian	84.0	2498.0	35.18	147.46	56.0	5.0	14.0
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.50	142.20	18.0	1.0	4.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.50	136.30	19.0	1.0	2.0
213	MS Dhoni	Chennai	Indian	220.0	4746.0	39.50	135.80	126.0	21.0	39.0
209	Sanju Samson	Rajasthan	Indian	121.0	3068.0	29.22	134.21	59.0	8.0	10.0
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.30	130.90	53.0	0.0	14.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.80	129.70	123.0	14.0	32.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.40	127.50	58.0	12.0	2.0
111	KS Bharat	Delhi	Indian	8.0	191.0	38.20	122.40	4.0	0.0	1.0

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
213	MS Dhoni	Chennai	Indian	220.0	4746.0	39.50	135.80	126.0	21.0	39.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.80	129.70	123.0	14.0	32.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.40	127.50	58.0	12.0	2.0
209	Sanju Samson	Rajasthan	Indian	121.0	3068.0	29.22	134.21	59.0	8.0	10.0
206	Rishabh Pant	Delhi	Indian	84.0	2498.0	35.18	147.46	56.0	5.0	14.0
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.30	130.90	53.0	0.0	14.0
219	Jos Butler	Rajasthan	Overseas	65.0	1968.0	35.14	150.00	34.0	3.0	1.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.50	136.30	19.0	1.0	2.0
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.50	142.20	18.0	1.0	4.0
111	KS Bharat	Delhi	Indian	8.0	191.0	38.20	122.40	4.0	0.0	1.0

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
213	MS Dhoni	Chennai	Indian	220.0	4746.0	39.50	135.80	126.0	21.0	39.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.80	129.70	123.0	14.0	32.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.40	127.50	58.0	12.0	2.0
209	Sanju Samson	Rajasthan	Indian	121.0	3068.0	29.22	134.21	59.0	8.0	10.0
206	Rishabh Pant	Delhi	Indian	84.0	2498.0	35.18	147.46	56.0	5.0	14.0
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.30	130.90	53.0	0.0	14.0
219	Jos Butler	Rajasthan	Overseas	65.0	1968.0	35.14	150.00	34.0	3.0	1.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.50	136.30	19.0	1.0	2.0
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.50	142.20	18.0	1.0	4.0
111	KS Bharat	Delhi	Indian	8.0	191.0	38.20	122.40	4.0	0.0	1.0

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
213	MS Dhoni	Chennai	Indian	220.0	4746.0	39.50	135.80	126.0	21.0	39.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.80	129.70	123.0	14.0	32.0
209	Sanju Samson	Rajasthan	Indian	121.0	3068.0	29.22	134.21	59.0	8.0	10.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.40	127.50	58.0	12.0	2.0
206	Rishabh Pant	Delhi	Indian	84.0	2498.0	35.18	147.46	56.0	5.0	14.0
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.30	130.90	53.0	0.0	14.0
219	Jos Butler	Rajasthan	Overseas	65.0	1968.0	35.14	150.00	34.0	3.0	1.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.50	136.30	19.0	1.0	2.0
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.50	142.20	18.0	1.0	4.0
111	KS Bharat	Delhi	Indian	8.0	191.0	38.20	122.40	4.0	0.0	1.0

	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
213	MS Dhoni	Chennai	Indian	220.0	4746.0	39.50	135.80	126.0	21.0	39.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.80	129.70	123.0	14.0	32.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.40	127.50	58.0	12.0	2.0
209	Sanju Samson	Rajasthan	Indian	121.0	3068.0	29.22	134.21	59.0	8.0	10.0
206	Rishabh Pant	Delhi	Indian	84.0	2498.0	35.18	147.46	56.0	5.0	14.0
219	Jos Butler	Rajasthan	Overseas	65.0	1968.0	35.14	150.00	34.0	3.0	1.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.50	136.30	19.0	1.0	2.0
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.50	142.20	18.0	1.0	4.0
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.30	130.90	53.0	0.0	14.0
111	KS Bharat	Delhi	Indian	8.0	191.0	38.20	122.40	4.0	0.0	1.0

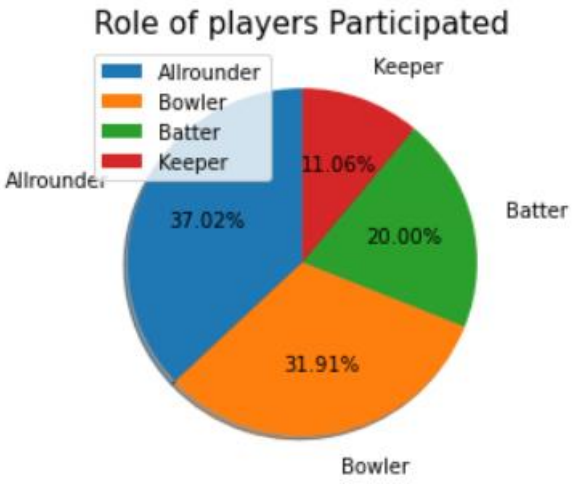
	Player Name	Team	Nationality	Matches_Played	Runs	Average	Strike_Rate	Catches	Run_outs	Stumps
213	MS Dhoni	Chennai	Indian	220.0	4746.0	39.50	135.80	126.0	21.0	39.0
109	Dinesh Karthik	Bangalore	Indian	213.0	4046.0	25.80	129.70	123.0	14.0	32.0
105	Quinton De Kock	Lucknow	Overseas	77.0	2256.0	31.30	130.90	53.0	0.0	14.0
206	Rishabh Pant	Delhi	Indian	84.0	2498.0	35.18	147.46	56.0	5.0	14.0
209	Sanju Samson	Rajasthan	Indian	121.0	3068.0	29.22	134.21	59.0	8.0	10.0
108	Jonny Bairstow	Punjab	Overseas	28.0	1038.0	41.50	142.20	18.0	1.0	4.0
106	Ambati Rayudu	Chennai	Indian	175.0	3916.0	29.40	127.50	58.0	12.0	2.0
107	Ishan Kishan	Mumbai	Indian	61.0	1452.0	28.50	136.30	19.0	1.0	2.0
111	KS Bharat	Delhi	Indian	8.0	191.0	38.20	122.40	4.0	0.0	1.0
219	Jos Butler	Rajasthan	Overseas	65.0	1968.0	35.14	150.00	34.0	3.0	1.0

Code:

```
#How many types of player have participated?

types = ipl['Player_Type'].value_counts()
types.reset_index()

plt.pie(types.values,
labels=types.index,labeldistance=1.2,autopct='%1.2f%%',shadow=True,startan
gle=60)
plt.title('Role of players Participated', fontsize =15)
plt.plot()
```

Output:

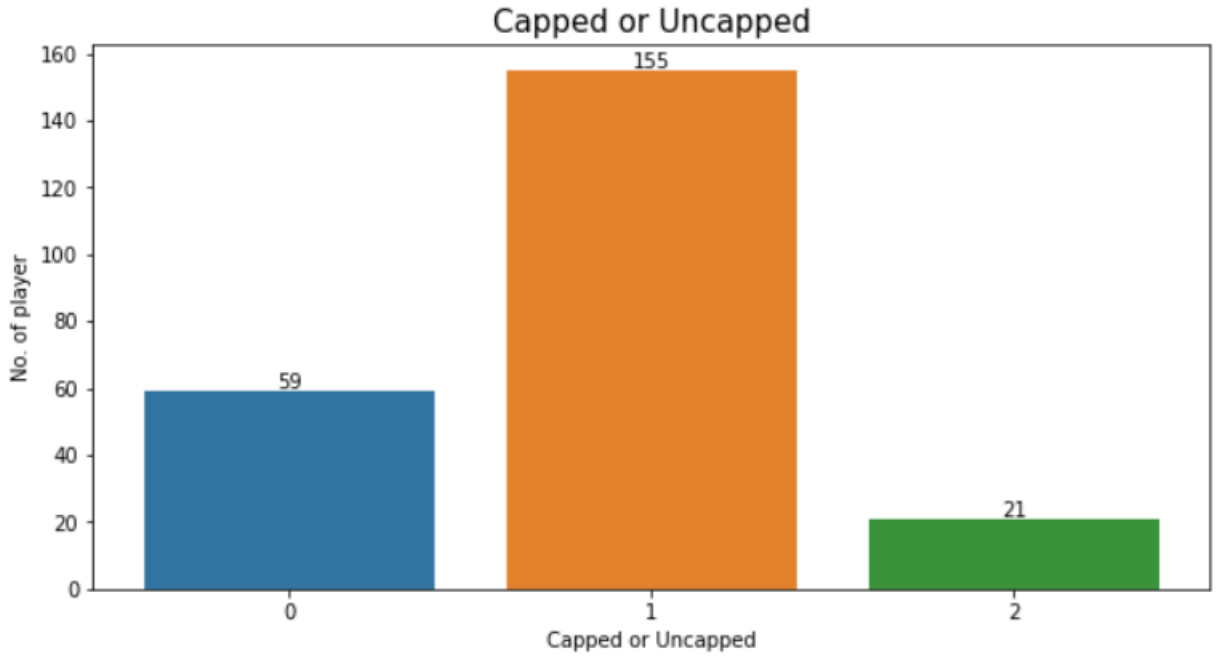
	index	Player_Type
0	Allrounder	87
1	Bowler	75
2	Batter	47
3	Keeper	26



Code:

```
plt.figure(figsize=(10,5))
#palette=['Orange','Pink','Red']
fig= sns.countplot(ipl['Capped'])
plt.xlabel('Capped or Uncapped')
plt.ylabel('No. of player')
plt.title('Capped or Uncapped',fontsize=15)
plt.plot()

for p in fig.patches:
    fig. annotate(format (p.get_height(), '.0f'), (p.get_x() +
    p.get_width()/2., p.get_height()), ha= 'center', va = 'center',
    xytext = (0, 4), textcoords= 'offset points')
```

Output:



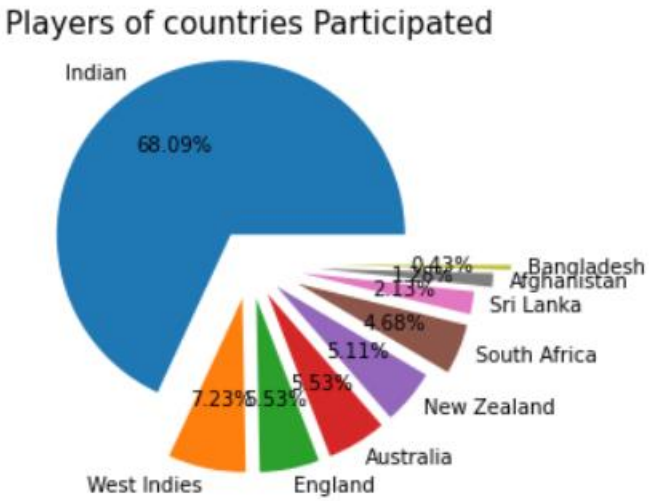
Code:

```
#How many types of countries player have participated?

types = ipl1['Nationality'].value_counts()
types.reset_index()

plt.pie(types.values,
labels=types.index,labeldistance=1.2,autopct='%1.2f%%',shadow=True,startan
gle=60)
```

```
plt.title('Players of countries Participated', fontsize =15)
plt.plot()
```

Output:

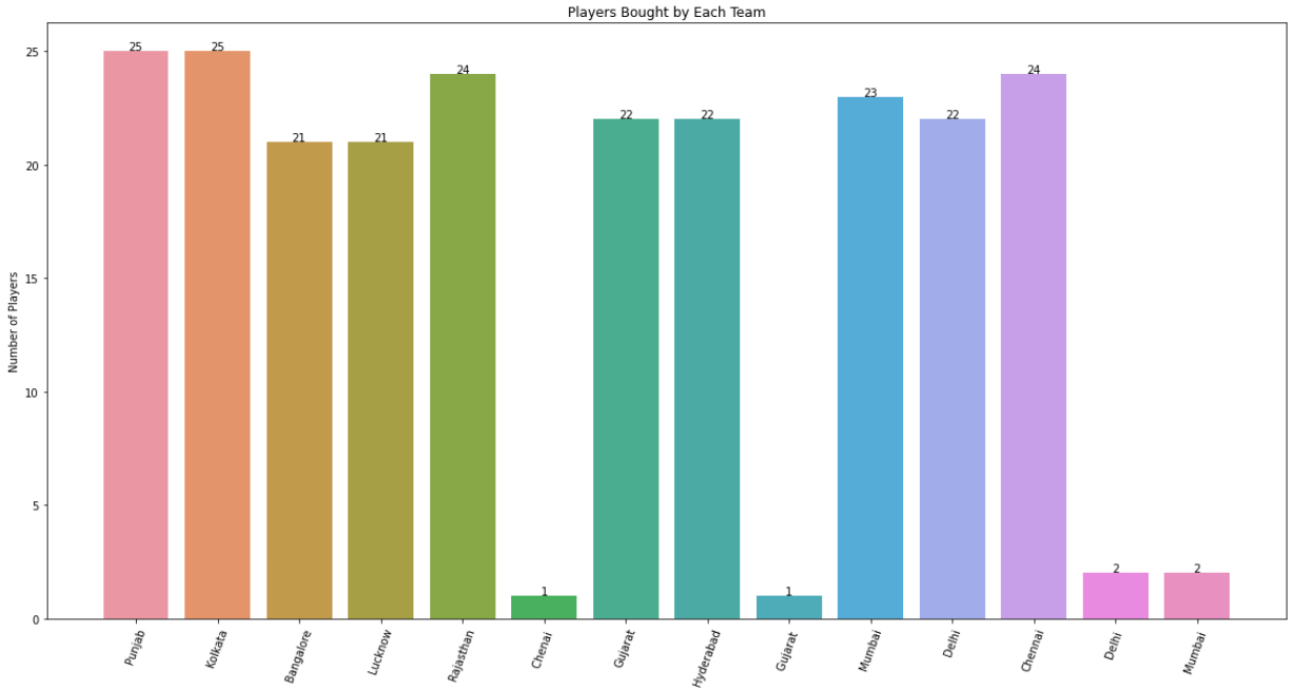
	index	Nationality
0	Indian	160
1	West Indies	17
2	England	13
3	Australia	13
4	New Zealand	12
5	South Africa	11
6	Sri Lanka	5
7	Afghanistan	3
8	Bangladesh	1



Code:

```
# Total number of players bought by each team.

plt.figure(figsize=(20,10))
fig = sns.countplot (ipl[ipl[ 'Team']!= 'Unsold' ][ 'Team'])
plt.xlabel('Team Names')
plt.ylabel('Number of Players')
plt.title('Players Bought by Each Team', fontsize=12)
plt.xticks (rotation=70)
plt.plot()

for p in fig.patches:
    fig. annotate(format (p.get_height(), '.0f'), (p.get_x() +
    p.get_width()/2., p.get_height()), ha = 'center', va = 'center',
xytext = (0, 4),
    textcoords='offset points')
```

Output:

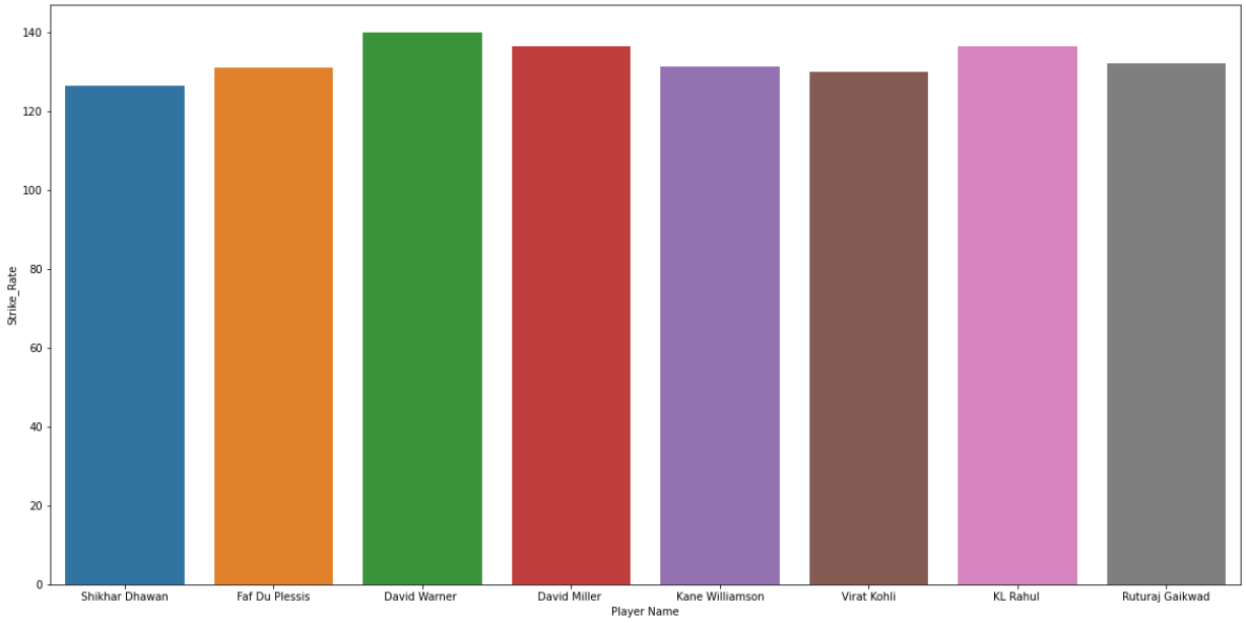


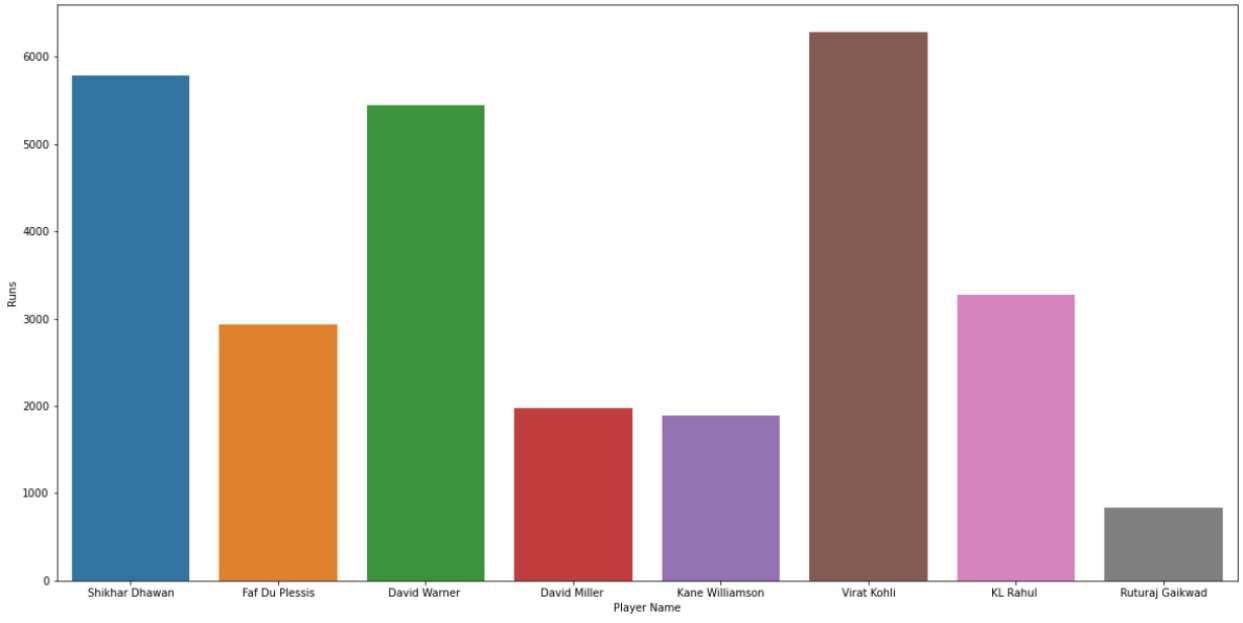
Code:

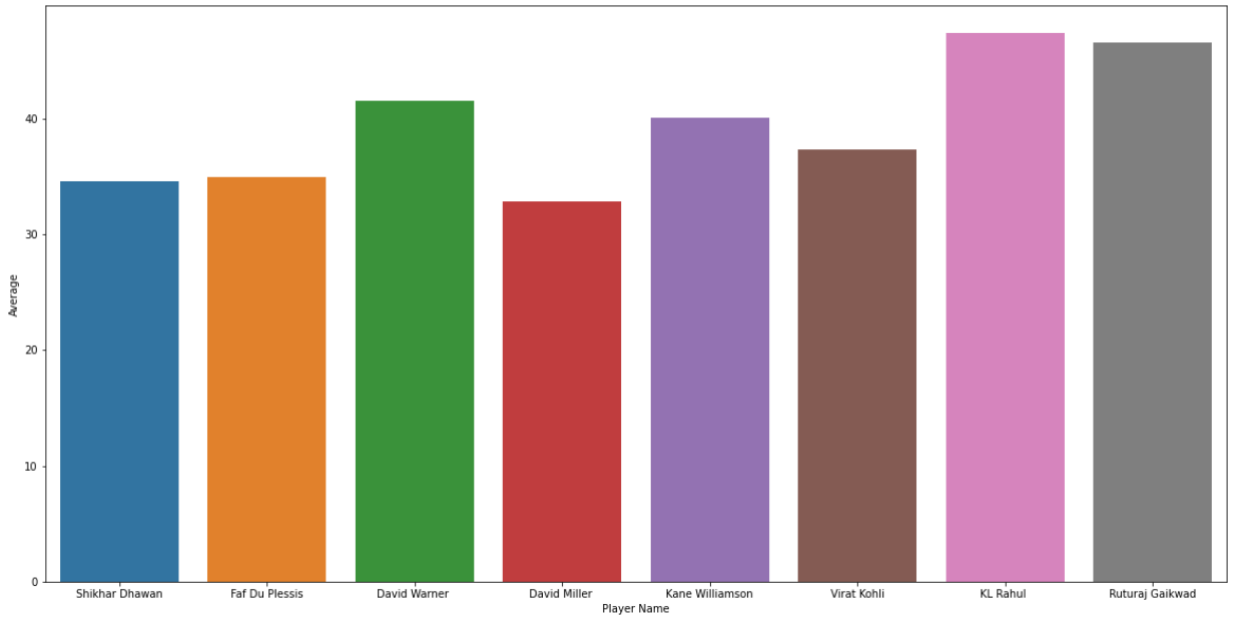
```
#visualization of batters data
#PLot Shows each of the top batters.

plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Strike_Rate', data = top_batters)

#visualization of batters data
#PLot Shows each of the top batters.

plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Runs', data = top_batters)

#visualization of batters data
#PLot Shows each of the top batters.

plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Average', data = top_batters)

#visualization of batters data
#PLot Shows each of the top batters.
```

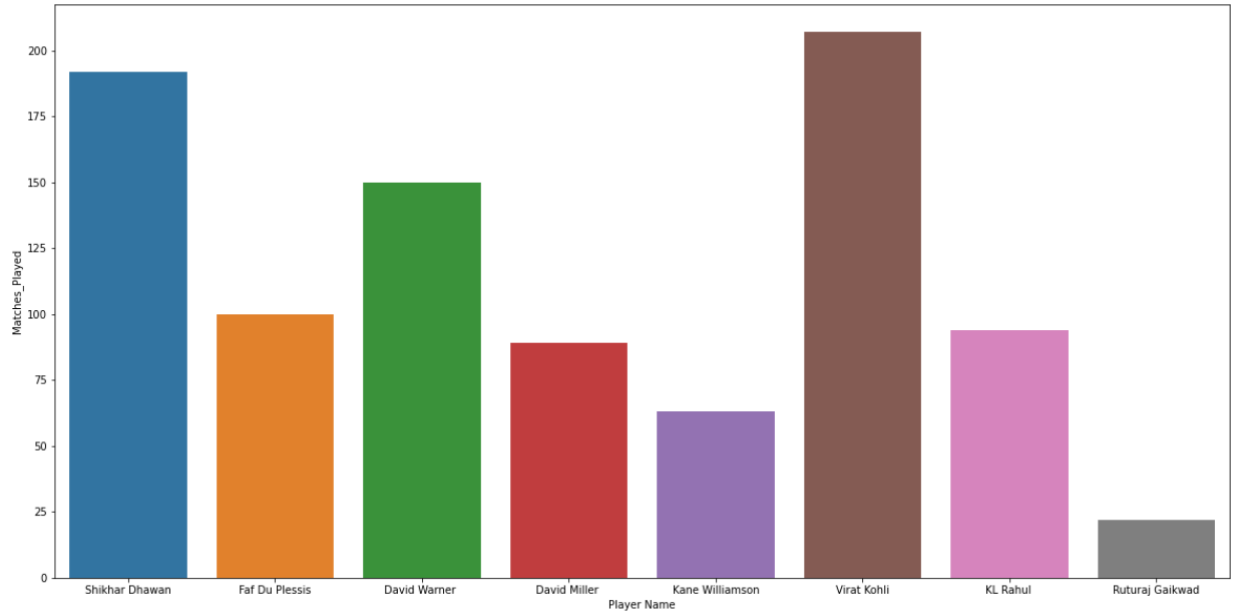
```
plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Matches_Played', data = top_batters)
```

Output:







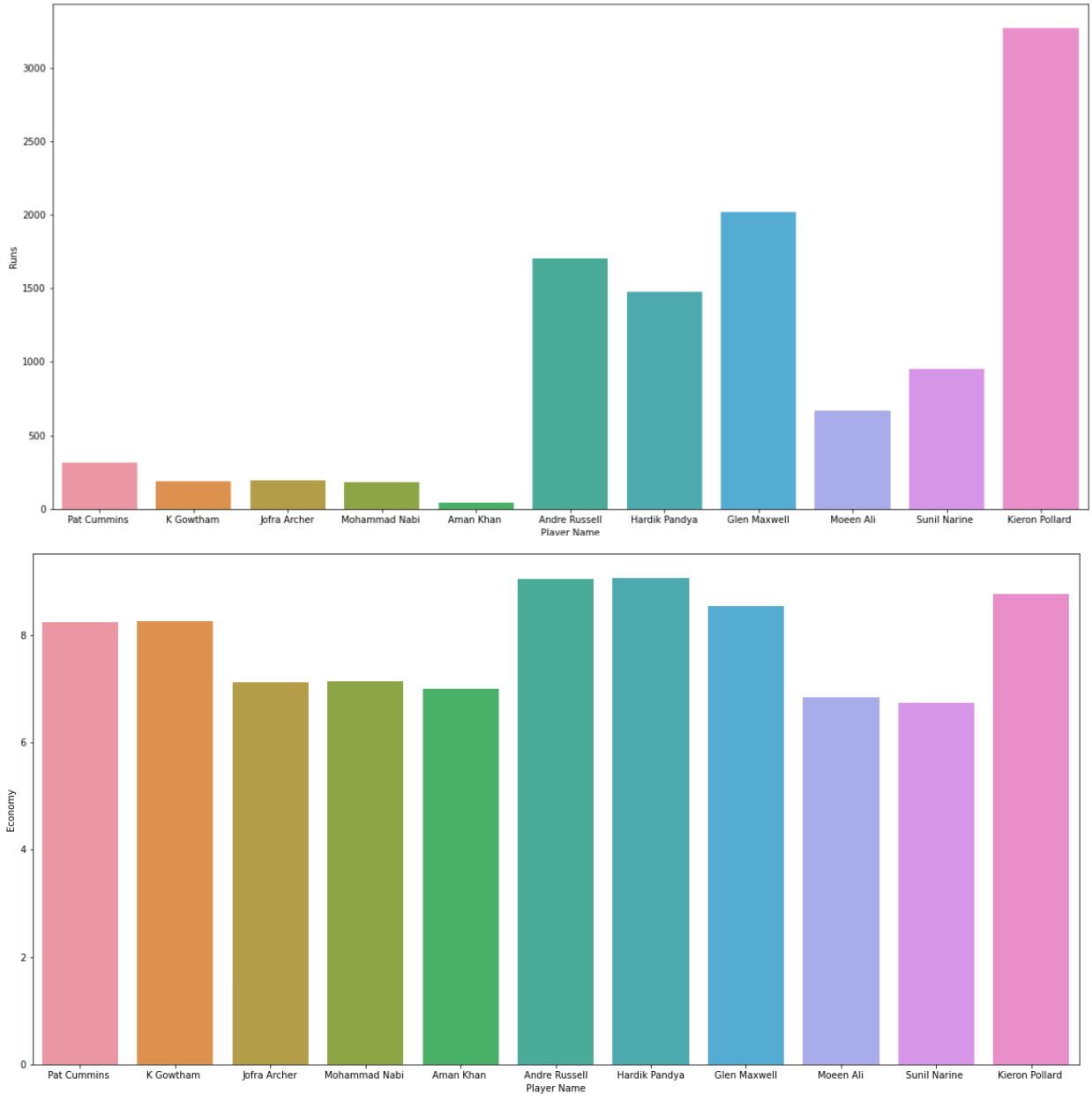


Code:

```
import pandas as pd
#visualization of allrounders data
#PLot Shows each of the top allrounders.

plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Runs', data = top_allrounders)

#visualization of allrounders data
#PLot Shows each of the top allrounders.
```

```
plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Economy', data = top_allrounders)
```

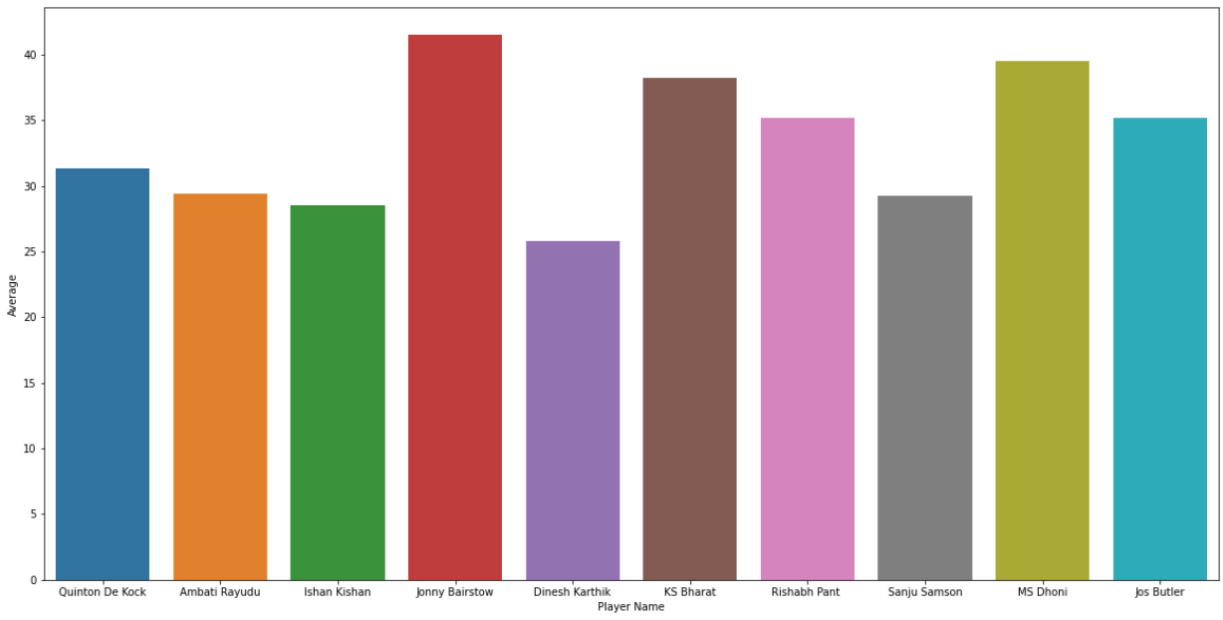
Output:

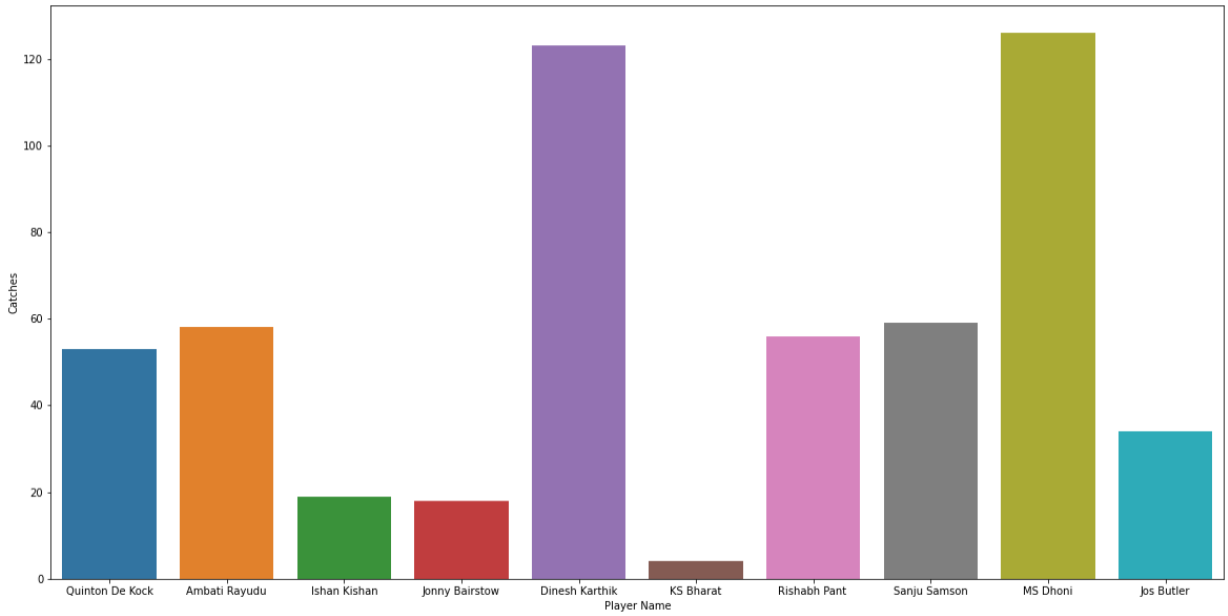


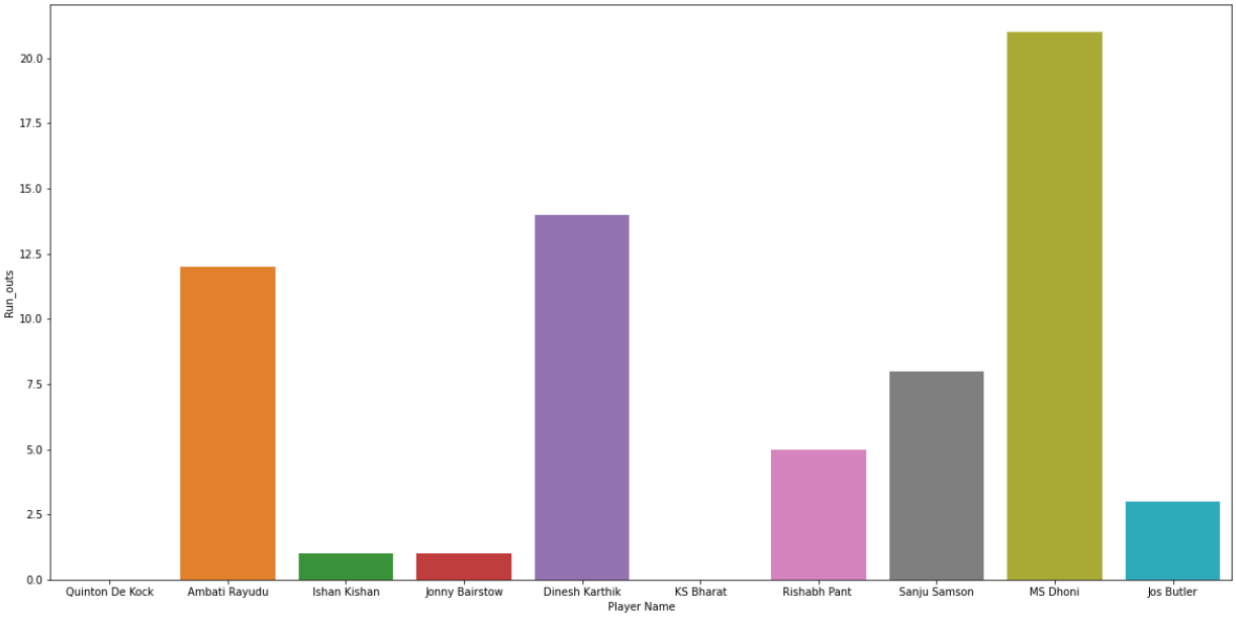
Code:

```
#visualization of keepers data
#PLot Shows each of the top keepers.

plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Average', data = top_keepers)

#visualization of keepers data
#PLot Shows each of the top keepers.

plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Catches', data = top_keepers)

#visualization of keepers data
#PLot Shows each of the top keepers.

plt.figure(figsize=(20,10))
sns.barplot(x= 'Player Name', y = 'Run_outs', data = top_keepers)
```

Output:



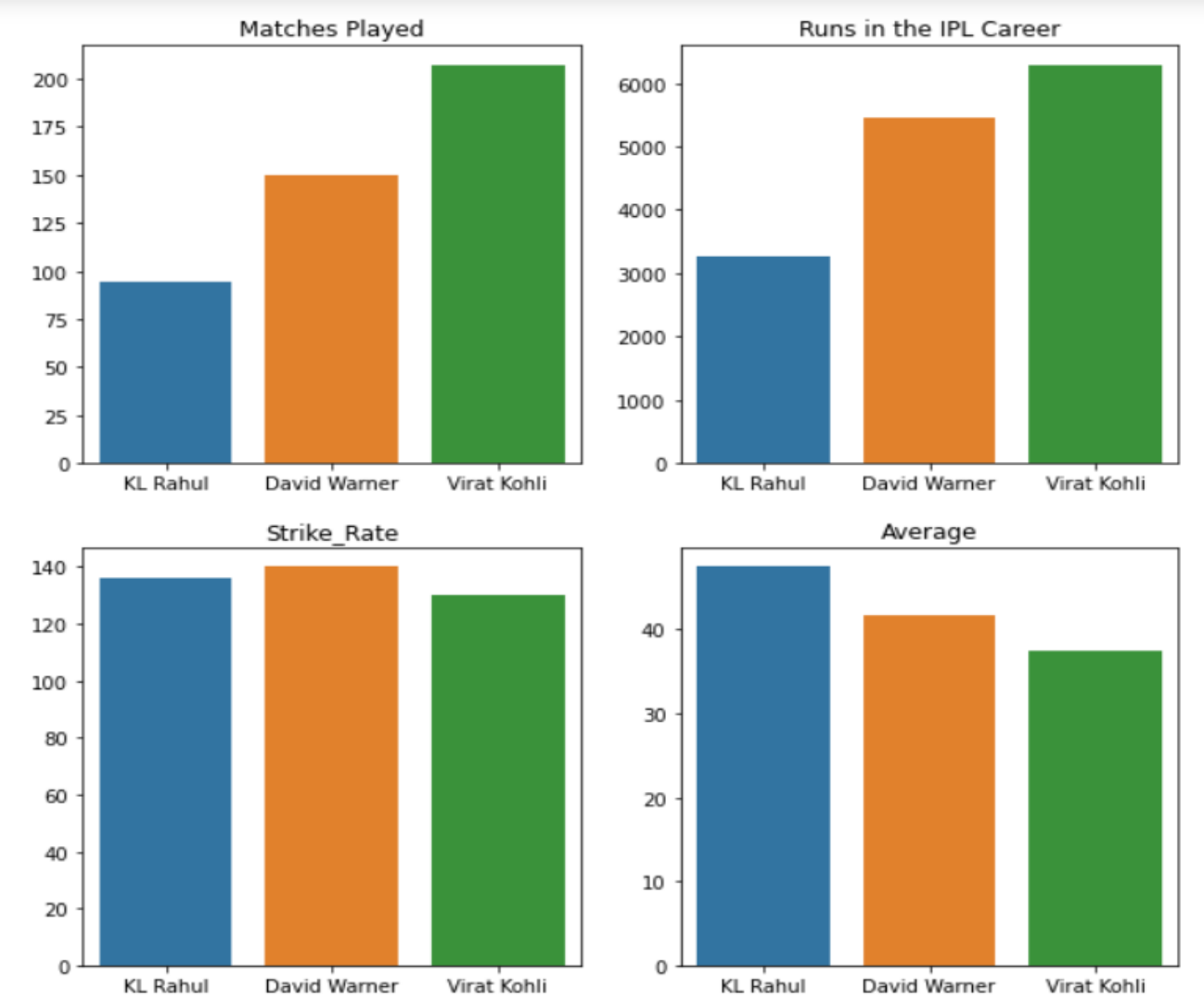

Catches
0
20
40
60
80
100
120
Quinton De Kock
Ambati Rayudu
Ishan Kishan
Jonny Bairstow
Dinesh Karthik
KS Bharat
Rishabh Pant
Sanju Samson
MS Dhoni
Jos Butler
Player Name


Run_outs
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Quinton De Kock
Ambati Rayudu
Ishan Kishan
Jonny Bairstow
Dinesh Karthik
KS Bharat
Rishabh Pant
Sanju Samson
MS Dhoni
Jos Butler
Player Name

Code:

```
#Batters for the Final 11 KL Rahul, Virat Kohli, David Warner
#here, we are storing the values of each player in a separate data frame
to use for displaying using the barplot.

top_batters.reset_index(drop=True)
matches_values =
[top_batters.iloc[6]['Matches_Played'],top_batters.iloc[2]['Matches_Played
'],top_batters.iloc[5]['Matches_Played']]
runs_values =
[top_batters.iloc[6]['Runs'],top_batters.iloc[2]['Runs'],top_batters.iloc[
5]['Runs']]
average_values =
[top_batters.iloc[6]['Average'],top_batters.iloc[2]['Average'],top_batters
.iloc[5]['Average']]
Strike_rate_values =
[top_batters.iloc[6]['Strike_Rate'],top_batters.iloc[2]['Strike_Rate'],top
_batters.iloc[5]['Strike_Rate']]
Labels = [ 'KL Rahul', "David Warner", "Virat Kohli"]

fig,axes =plt.subplots(2,2, figsize=(10,10))
axes[0][0].set_title("Matches Played")
axes[0][1].set_title("Runs in the IPL Career")
axes[1][0].set_title("Strike_Rate")
axes[1][1].set_title("Average")

sns.barplot(x=Labels, y=matches_values, ax=axes[0][0])
sns.barplot(x=Labels, y=runs_values, ax=axes[0][1])
sns.barplot(x=Labels, y=Strike_rate_values,ax=axes[1][0])
sns.barplot(x=Labels, y=average_values, ax=axes[1][1])
```

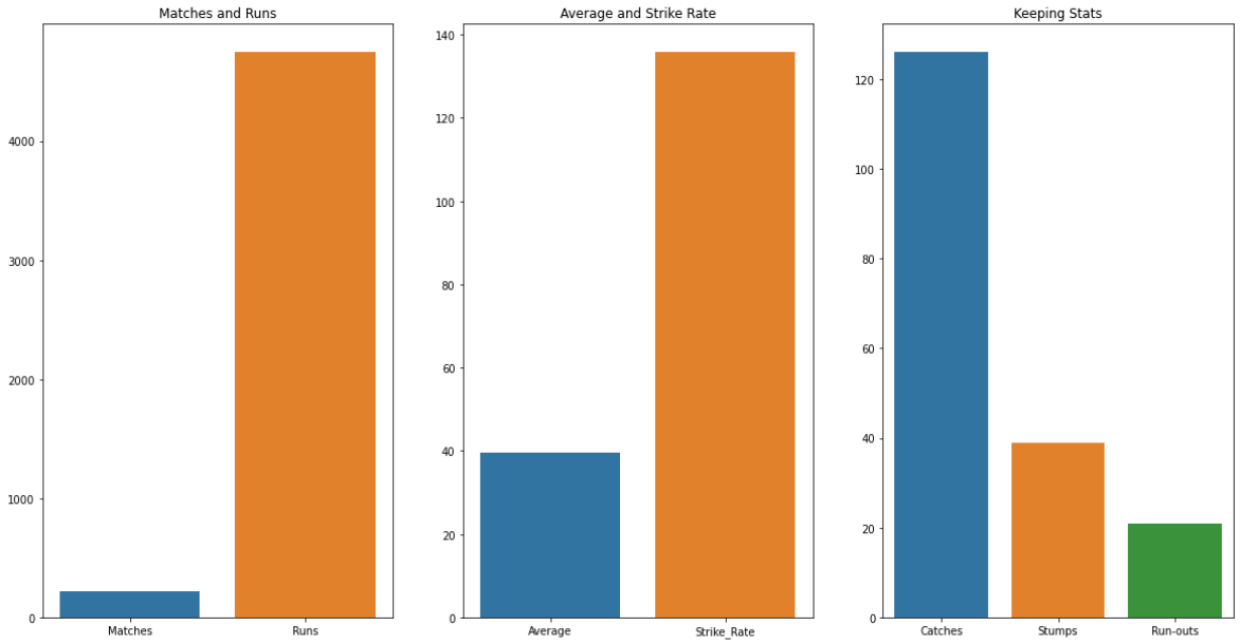
Output:


Matches Played
200
175
150
125
100
75
50
25
0
KL Rahul
David Warner
Virat Kohli
Runs in the IPL Career
6000
5000
4000
3000
2000
1000
0
KL Rahul
David Warner
Virat Kohli
Strike_Rate
140
120
100
80
60
40
20
0
KL Rahul
David Warner
Virat Kohli
Average
40
30
20
10
0
KL Rahul
David Warner
Virat Kohli

Code:

```
#Wicket Keeper For The Final 11- MS Dhoni

matches_values =
[top_keepers.iloc[8]['Matches_Played'],top_keepers.iloc[8]['Runs']]
average_values =
[top_keepers.iloc[8]['Average'],top_keepers.iloc[8]['Strike_Rate']]
keeping_values =
[top_keepers.iloc[8]['Catches'],top_keepers.iloc[8]['Stumps'],top_keepers.
iloc[8]['Run_outs']]


label1= ['Matches','Runs']
label2= ['Average','Strike_Rate']
label3= ['Catches','Stumps','Run-outs']


fig, axes = plt.subplots(1,3,figsize=(20,10))
axes[0].set_title('Matches and Runs' )
axes[1].set_title('Average and Strike Rate')
axes[2].set_title('Keeping Stats')

sns.barplot(x=label1, y= matches_values, ax= axes[0])
sns.barplot(x=label2, y= average_values, ax= axes[1])
sns.barplot(x=label3, y= keeping_values, ax= axes[2])
```

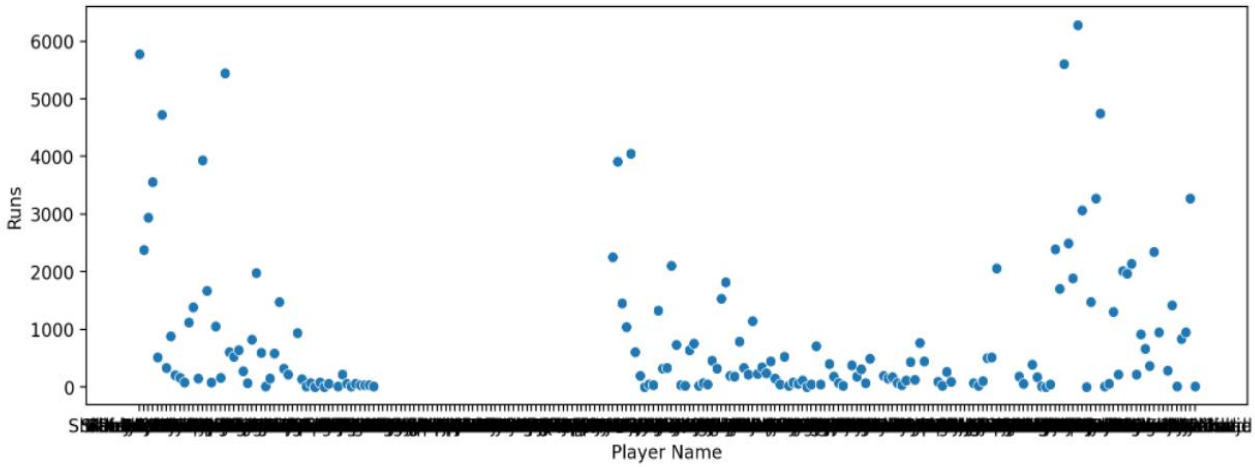
Outputs:



Code:

```
#Scatter plot
plt.figure(figsize=(12,4),dpi=200)
sns.scatterplot(x= 'Player Name', y = 'Runs', data = ipl)
```

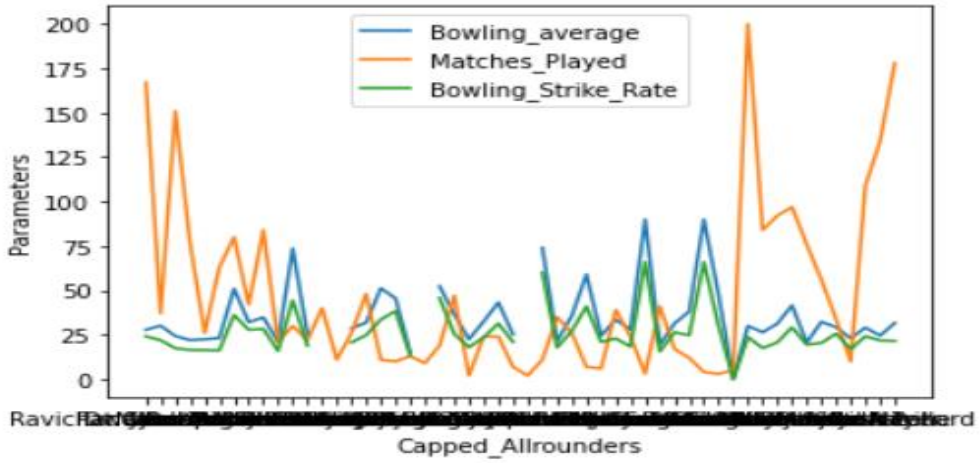
Output:



Code:

```
#Line Plot for Capped_Allrounders.
plt.plot(Capped_Allrounders.Player_Name,Capped_Allrounders.Bowling_average
,label="Bowling_average")
plt.plot(Capped_Allrounders.Player_Name,Capped_Allrounders.Matches_Played,
label="Matches_Played")
plt.plot(Capped_Allrounders.Player_Name,Capped_Allrounders.Bowling_Strike_
Rate,label="Bowling_Strike_Rate")

plt.xlabel('Capped_Allrounders')
plt.ylabel('Parameters')
plt.legend()
plt.show()
```

Output:



Conclusion and Future Scope:

Final Playing 11 on above analysis will be

#1.KL Rahul
#2.David Warner
#3.Virat Kohli
#4.Andre Russell
#5.Sunil Narine
#6.Hardik Pandya
#7.Ms Dhoni
#8.Yuzvendra Chahal
#9.Jasprit Bumrah
#10.Nathan Coulter-Nile
#11.Kagiso Rabada

The results show that the problem of match result prediction can be solved and a mathematical model can be created for prediction of the results of the matches prior to the match based on the knowledge of past matches, playing eleven and the toss result. In this work 5 features of IPL career and 5 features of International T20 Career have been taken into consideration for both batsmen and bowlers but in future work more features can be created and taken into account. The training data can be made larger in the future work for better model learning and classification. Various other data analytics techniques can also be used for improving the accuracy of the model in future work

References:

- https://www.youtube.com/watch?v=vC7xtwdE5-Q
- https://www.kaggle.com/kalilurrahman/ipl-player-auction-data-analysis
- https://www.w3schools.com/python/matplotlib_pyplot.asp