PAGE INDEX

Introduction

The Indian Premier League (IPL) is a Professional men's Twenty20 cricket league in which 10 Teams compete from ten different locations. Millions of People, especially Indians, are obsessed with the Indian Premier League (IPL), and our job involves data analysis and match prediction for IPL matches. In recent years, Analytics has been used to predict and draw various Insights in the field of sports. IPL Data Analysis is all About utilizing data science, machine learning to analyze Data that is already existing in a data collection. This Application design will be implemented for the purpose Of analyzing the IPL data by fetching different attributes And building a predictive model that could predict the Score, batsmen run, predict the winner, overall Performance of the team, and the players, head to head-Analysis. Data analysis is a process of inspecting, cleansing, transforming, and modeling data with the goal of discovering useful information, informing conclusions, and supporting decision-making.

1.1 Feature Selection

It is the process of reducing the number of input variables when developing a predictive model. It is desirable to reduce the number of input variables to both reduce the computational cost of modeling and in some cases, to improve the performance of the model. Feature selection is the process of selecting the best collection of features from a set of input features by minimizing data dimension and improving time and space complexity by deleting unnecessary characteristics. Choosing features increases the model's performance while also conserving time and space.

1.2 Problem Statement

Analytics has been used to predict and draw various Insights in the field of sports. Cricket Data Analysis is all About utilizing data science, machine learning to analyze Data that is already existing in a data collection. This Application design will be implemented for the purpose Of analyzing the international matches data by fetching different attributes And building a predictive model that could predict the Score, batsmen run, predict the best playesr, overall Performance of the player, and the players, head to head-Analysis.

1.3 Objectives of Study

The complete work done has been compactly organized into this architecture. It starts with the preparation of datasets and their loading into the backend. The user interface is then provided with several features that can be used on the players or matches. It may also be used to make predictions. We can implement the following modules for analysis, prediction, ranking, and visualization.

Implementation

2.1 Use of Libraries

Numpy : Numpy is one of the most commonly used packages for scientific computing in Python. It provides a multidimensional array object, as well as variations such as masks and matrices, which can be used for various math operations. Numpy is compatible with, and used by many other popular Python packages, including pandas and matplotlib.

Pandas : Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays. As one of the most popular data wrangling packages, Pandas works well with many other data science modules inside the Python ecosystem, and is typically included in every Python distribution.

Stats : Stats is the SciPy sub-package. It is mainly used for probabilistic distributions and statistical operations. There is a wide range of probability functions. The statistical functionality is expanding as the library is open-source.

Scipy : Scipy in Python is an open-source library used for solving mathematical, scientific, engineering, and technical problems. It allows users to manipulate the data and visualize the data using a wide range of high-level Python commands. SciPy is built on the Python NumPy extention.

2.2 Statistical Tools

Mean : Mean (usually referred to Arithmetic Mean, also called Average) is calculated as sum of all numbers in the dataset and dividing by the total number of values.

Standard Deviation : Standard deviation and variance measures the spread of a dataset. If the data is spread out largely, standard deviation (and variance) is greater.

Formula

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}}$$

σ = population standard deviation
N = the size of the population
x_i = each value from the population
μ = the population mean

Coefficient of Variation : The coefficient of variation (CV) is the ratio of the standard deviation to the mean. The higher the coefficient of variation, the greater the level of dispersion around the mean.

$$CV = \frac{\sigma}{\mu}$$

σ = population standard deviation
μ = population mean

Graphs : Graph can be defined as pictorial representation or a diagram that represent data or values in an organized manner. The points on graph often represent the relationship between two or more things.

Types of graph :

Bar Graph : A bar graph is the representation of numerical data by rectangles (or bars) of equal width and varying height. The gap between one bar and another is uniform throughout. Bar graphs can be either horizontal or vertical. The height or length of each bar relates directly to its value.

Pia chart : A pie chart is a type of graph that represents the data in the circular graph. The slices of pie show the relative size of the data, and it is a type of pictorial representation of data. A pie chart requires a list of categorical variables and numerical variables. Here, the term “pie” represents the whole, and the “slices” represent the parts of the whole.

Stacked Bar Graph : The stacked bar graphs are used to show dataset subgroups. However, the bars are stacked on top of each other.

Line Graph : It displays a sequence of data points as markers. The points are ordered typically by their x-axis value. These points are joined with straight line segments. A line graph is used to visualize a trend in data over intervals of time.

Multiple Line Graph : Multiple line graphs contain more than one line. They represent multiple variables in a dataset. This type of graph can be used to study more than one variable over the same period.

Code used for data processing :

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
m=np.mean(df,axis=1)
print("average of players in  matches\n\n",m)
max=np.max(df,axis=1)
print("highest individual score by   playeres \n",max)
```

```
total=np.sum(df,axis=1)
print("total score of 10 matches= \n",total)
print("standard deviation ")
d=round((df.std(axis = 1, numeric_only = True)),2)
print(d)
print("coeeficient of variation = \n")
for i in range (len(d)):
  print(round((((d[i])/m[i])),4))
```

After processing data we get this output : because outputs are lenthly we converted it into excel file as follow

1	player	Highest	Total	Average	S.D	C.V
2	devon convey	92	356	35.6	29.49	0.8284
3	glenn philips	104	356	35.6	33.26	0.9343
4	suryakumar yadav	67	221	22.1	21.27	0.9624
5	aaron finch	63	243	24.3	21.36	0.879
6	dawid malan	82	303	30.3	29.23	0.9647
7	alex hales	86	318	31.8	33.34	1.0484
8	jos buttler	80	389	38.9	29.28	0.7527
9	virat kohli	82	408	40.8	28.49	0.6983
10	quinton de kock	69	276	27.6	30.21	1.0946
11	lokesh rahul	57	247	24.7	23.93	0.9688
12	rohit sharma	53	176	17.6	18.4	1.0455
13	rassie van de dussen	94	286	28.6	32.3	1.1294
14	martin guptill	45	232	23.2	13.36	0.5759
15	mitchell marsh	45	203	20.3	17.38	0.8562
16	bhanuka rajapaksha	71	220	22	19.93	0.9059
17	david miller	106	257	25.7	33.79	1.3148
18	ishan kishan	54	224	22.4	15.9	0.7098
19	kane williamson	61	368	36.8	20.09	0.5459
20	glenn maxwell	54	141	14.1	16.91	1.1993
21	daryl mitchel	53	258	25.8	18.75	0.7267
22	david warner	75	270	27	27.12	1.0044
23	marcus stoinis	59	238	23.8	18.3	0.7689
24	moeen ali	44	111	11.1	14.26	1.2847
25	hardik pandya	63	205	20.5	19.7	0.961
26	jonny bairstow	90	199	19.9	26.59	1.3362

In this table data is ruduced to half by doing statical approach in python then collecting all values to excel file

Converted 10 matches data into five column to predict like highest , Total , Average , S.D, C.V

Code used for plotting graph :

```
import seaborn as sns
import matplotlib.pyplot as plt
pl = df['palyer'].tolist()
plt.plot(m, pl)
#plt.plot(m2, pl, '-.')

plt.xlabel("average run by player")
plt.ylabel("players name ")
plt.title('average of
player')
plt.show()
```

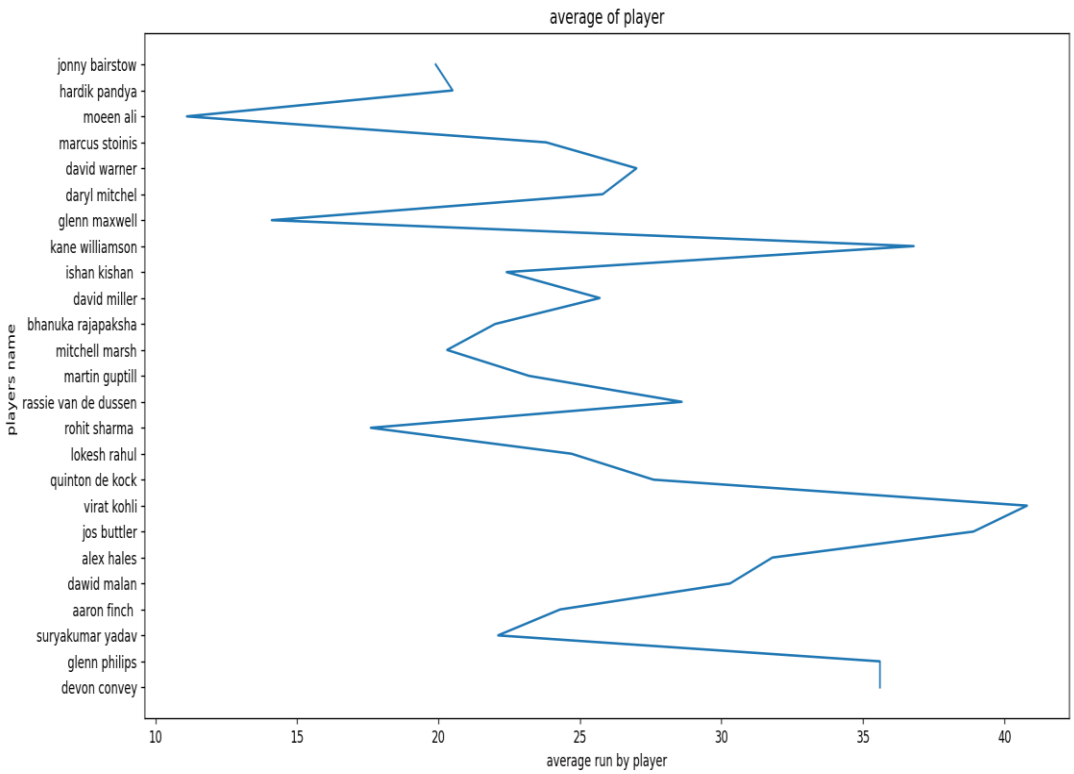
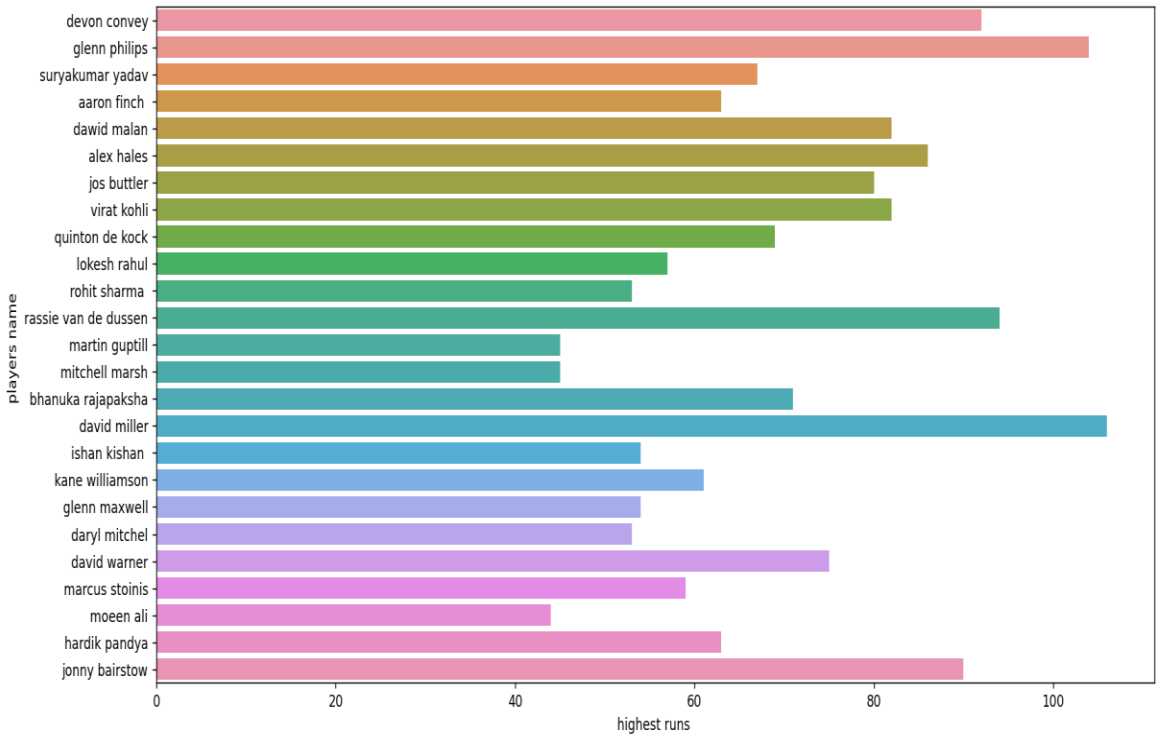


Chart showing that virat kohli has highest average in the series he has average of “ 40.8 ”

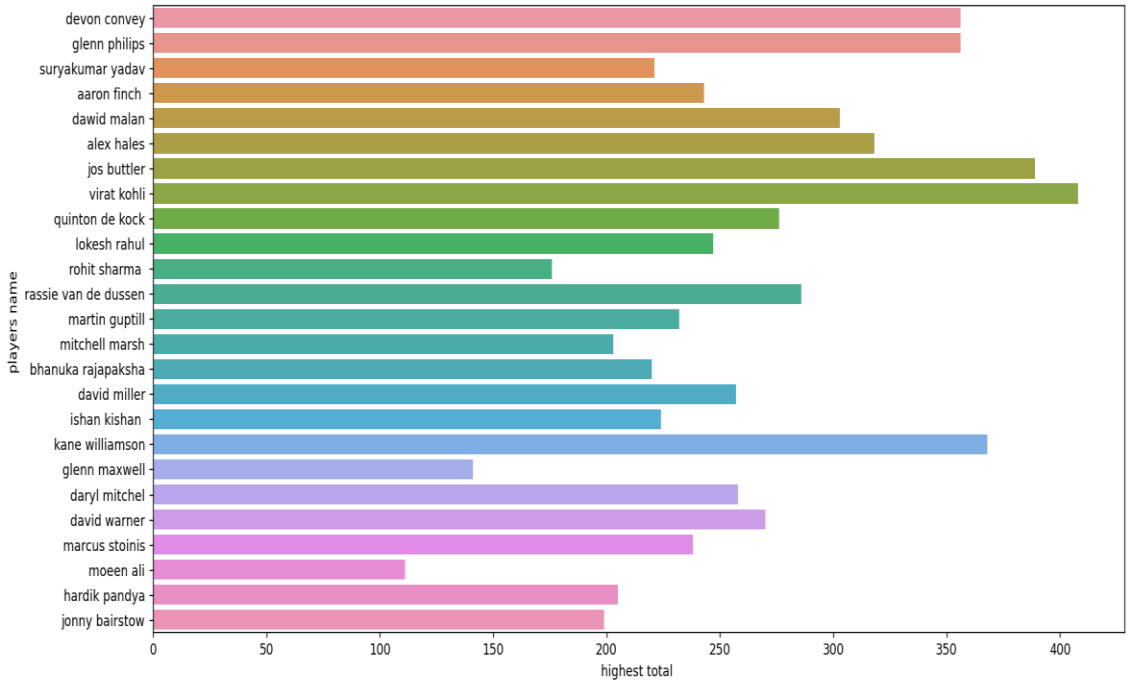
Barchart showing highest individual runs into 10 matches series :

```
sns.barplot(x=np.max(df,axis=1),y='palyer',data=df)
plt.xlabel("highest runs ")
plt.ylabel("players name ")
plt.show()
```



This graph shows that highest score in 10 matches is by “ David miller ” and “ gleen Philips”

Barchart showing highest total run scorer in 10 matches :

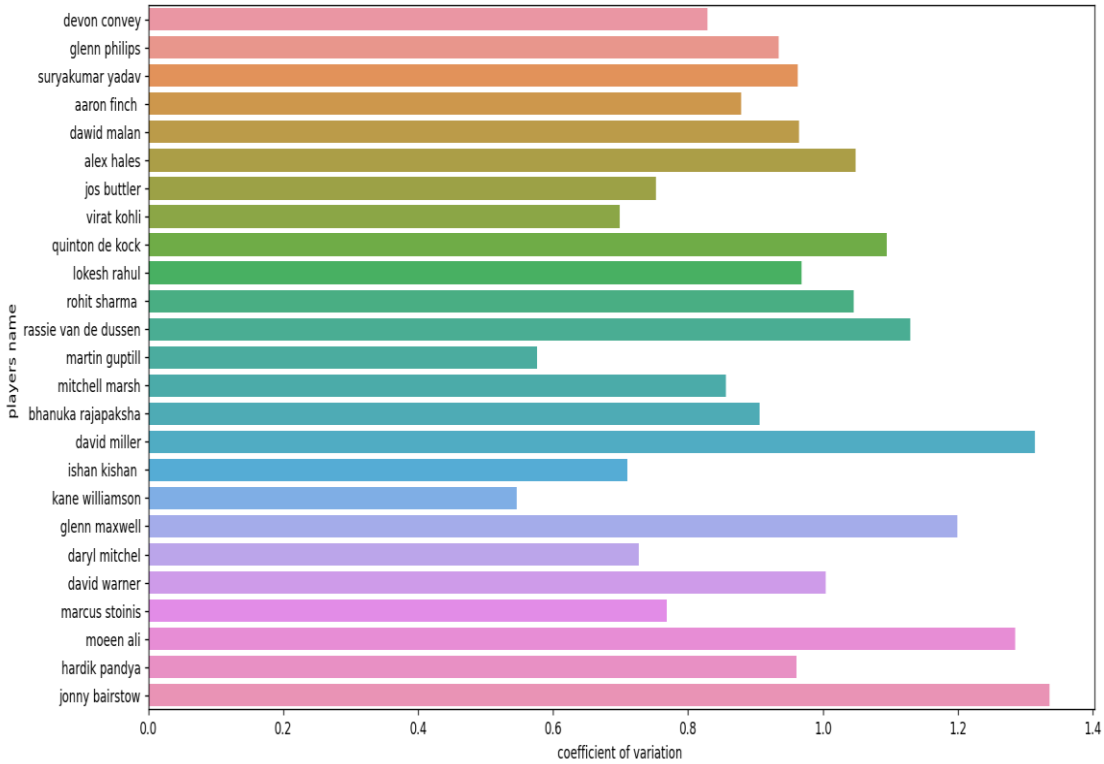


Virat kohli has highest runs total in 10 matches And second is alex hales .

Barchart showing Coefficient of variation :

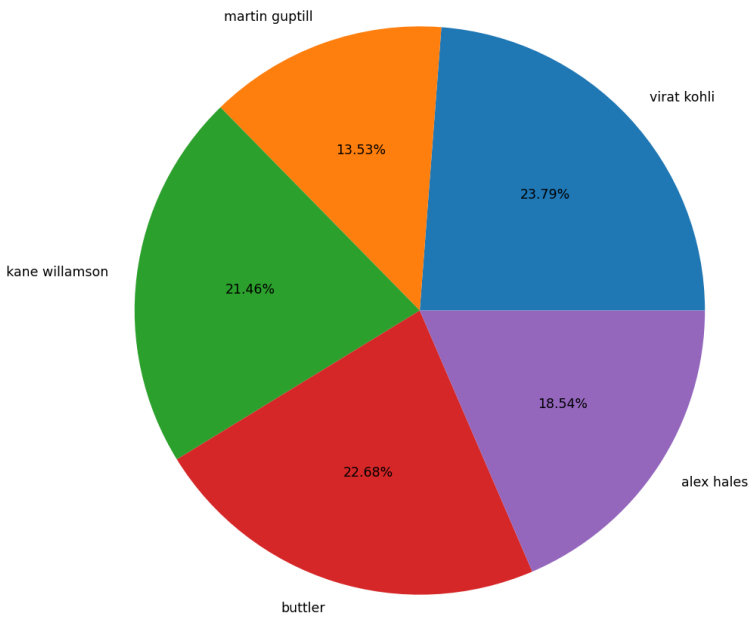
Kane willimsion and martin guptil and virat kohli has lowest c.v. so they are consistant in last 10 matches

They don’t show high variation in performance

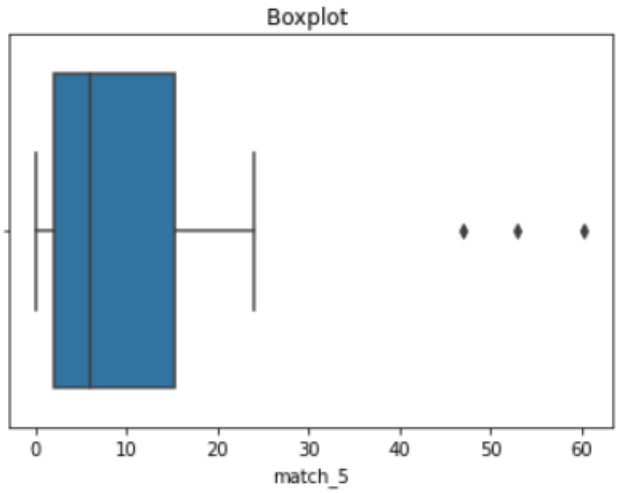



```
import matplotlib.pyplot as plt
mean = np.array([40.8, 23.2, 36.8,38.9,31.8 ])
total=np.array([408,232,368,389,318])
high=np.array([82,45,61,80,86])
cv=np.array([0.6983,0.5759,0.5459,0.7527,1.0484])
mylabels = ["virat kohli", "martin guptill", "kane willamson", "buttle
r","alex hales"]

fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.axis('equal')
ax.pie(mean, labels = mylabels,autopct='%1.2f%%')
plt.legend(title = "average runes by player")
plt.show()
```



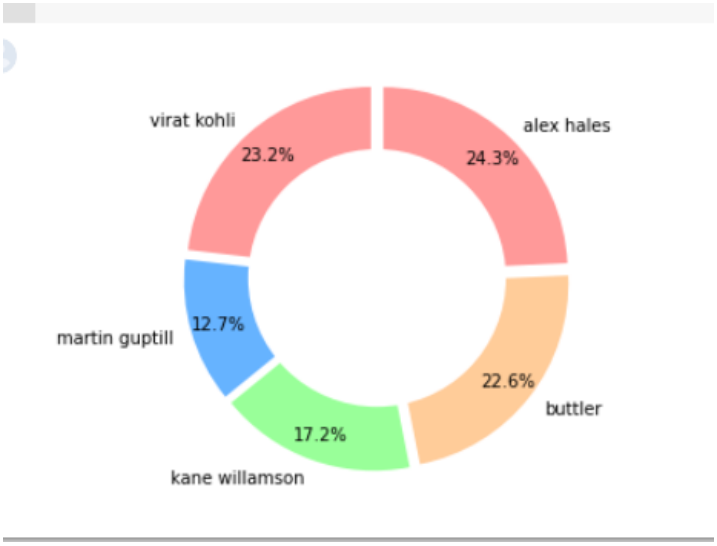

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(data=df,x=df["match_5"])
plt.title("Boxplot ")
```




```
mean = np.array([40.8, 23.2, 36.8,38.9,31.8 ])
total=np.array([408,232,368,389,318])
high=np.array([82,45,61,80,86])
cv=np.array([0.6983,0.5759,0.5459,0.7527,1.0484])
mylabels = ["virat kohli", "martin guptill", "kane willamson", "buttle
r","alex hales"]
```

```
colors = ['#ff9999', '#66b3ff', '#99ff99', '#ffcc99']
# explsion
explode = (0.05, 0.05, 0.05, 0.05,0.05)

plt.pie(high, colors=colors, labels=mylabels, autopct='%1.1f%%', start
angle=90, pctdistance=0.85, explode=explode)
# draw circle
centre_circle = plt.Circle((0, 0), 0.70, fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
# Equal aspect ratio ensures that pie is drawn as a circle
ax1.axis('equal')
plt.tight_layout()
plt.show()
```



Data Set 2

Importing data

```
import pandas as pd
import io

df=pd.read_csv(io.BytesIO(uploaded['batting.csv']))
df[:90]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
0	Ewin lewis	MI	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	382.0	48.0	151.0	73.0
1	Wriddiman Saha	SRH	Batsman	68.0	6.0	70.0	362.0	249.0	270.0	234.0	122.0	86.0	214.0	131.0	317.0
2	AB De Villiers	RCB	Batsman	312.0	319.0	360.0	395.0	513.0	687.0	216.0	480.0	442.0	454.0	313.0	0.0
3	Ajinkya Rahane	CSK	Batsman	120.0	560.0	488.0	339.0	540.0	480.0	382.0	370.0	393.0	113.0	8.0	44.0
4	Ambati Raydu	CSK	Batsman	395.0	333.0	265.0	361.0	281.0	334.0	91.0	602.0	292.0	359.0	257.0	274.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
77	Vijay Shankar	GT	All rounder	NaN	NaN	NaN	NaN	NaN	0.0	101.0	212.0	244.0	97.0	58.0	19.0
78	Virat Kohli	RCB	Batsman	557.0	364.0	634.0	359.0	505.0	973.0	208.0	530.0	464.0	466.0	405.0	341.0
79	Washigton Sunder	SRH	All rounder	NaN	NaN	NaN	NaN	NaN	NaN	9.0	65.0	1.0	111.0	31.0	101.0
80	Yashasvi jaiswal	RR	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	34.0	249.0	258.0
81	Yusuf Pathan	SRH	All rounder	283.0	194.0	332.0	268.0	312.0	361.0	143.0	260.0	40.0	189.0	0.0	0.0

```
all=df.describe()
round(df.describe(),4)
```

	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
count	32.0000	32.0000	35.0000	39.0000	44.0000	51.0000	54.0000	59.0000	66.0000	70.0000	71.0000	71.0000
mean	230.0000	279.0312	303.8571	268.1282	272.7727	253.9608	249.2222	264.2542	239.6818	232.0286	198.0141	241.2113
std	164.4016	186.7127	203.4306	162.8088	144.6622	192.7816	161.2476	190.4834	170.9908	169.2191	165.8585	174.8262
min	8.0000	6.0000	2.0000	8.0000	0.0000	0.0000	9.0000	1.0000	1.0000	0.0000	0.0000	0.0000
25%	95.7500	154.2500	128.5000	145.5000	160.0000	111.0000	107.2500	100.5000	91.2500	88.7500	60.0000	87.0000
50%	228.0000	246.5000	311.0000	325.0000	238.0000	237.0000	241.5000	228.0000	221.0000	202.0000	151.0000	250.0000
75%	360.7500	379.5000	447.5000	374.0000	372.5000	344.5000	364.7500	426.0000	379.7500	349.7500	294.5000	351.0000
max	608.0000	773.0000	708.0000	660.0000	540.0000	973.0000	848.0000	684.0000	692.0000	670.0000	635.0000	863.0000

From the above output we can see that maximum runs by single player in season for is 608, from that we can player detail i.e there team , name , player's all season runs

```
df[df['Season 4'] == 608]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
10	Chris Gayle	RCB	Batsman	608.0	773.0	708.0	196.0	491.0	227.0	200.0	368.0	490.0	288.0	193.0	0.0

Like that we can find out maximum scorer same for other seasons also

```
df[df['Season 6'] == 708]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
10	Chris Gayle	RCB	Batsman	608.0	773.0	708.0	196.0	491.0	227.0	200.0	368.0	490.0	288.0	193.0	0.0

```
df[df['Season 7'] == 660]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
57	Robin Uthappa	CSK	Batsman	264.0	405.0	434.0	660.0	364.0	394.0	388.0	351.0	282.0	196.0	115.0	230.0

```
df[df['Season 8'] ==540]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
3	Ajinkya Rahane	CSK	Batsman	120.0	560.0	488.0	339.0	540.0	480.0	382.0	370.0	393.0	113.0	8.0	44.0

```
df[df['Season 9'] == 973]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
78	Virat Kohli	RCB	Batsman	557.0	364.0	634.0	359.0	505.0	973.0	208.0	530.0	464.0	466.0	405.0	341.0

```
df[df['Season 10'] == 848]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
13	David Warnar	DC	Batsman	163.0	324.0	256.0	410.0	528.0	562.0	848.0	641.0	692.0	548.0	195.0	432.0

The reward of Orange Cup is given to those player who has more runs in particular season , runs of all player of all seasons are listed in our dataset. From that we can find the purple cap holder ,most scorer of the season. Chris Gayle , Robbin Uthappa, Ajinkya Rahane , Virat Kohli , David Warner are the Orange cap holder of respevtive seasons.

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
55	Rishab Pant	DC	Batsman	NaN	NaN	NaN	NaN	NaN	198.0	366.0	684.0	488.0	343.0	419.0	340.0

```
df[df['Season 12'] == 692]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
13	David Warnar	DC	Batsman	163.0	324.0	256.0	410.0	528.0	562.0	848.0	641.0	692.0	548.0	195.0	432.0

```
df[df['Season 13'] == 670]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
31	KL Rahul	LSG	Batsman	NaN	NaN	NaN	20.0	166.0	142.0	397.0	659.0	593.0	670.0	626.0	616.0

```
df[df['Season 14'] == 635]
```

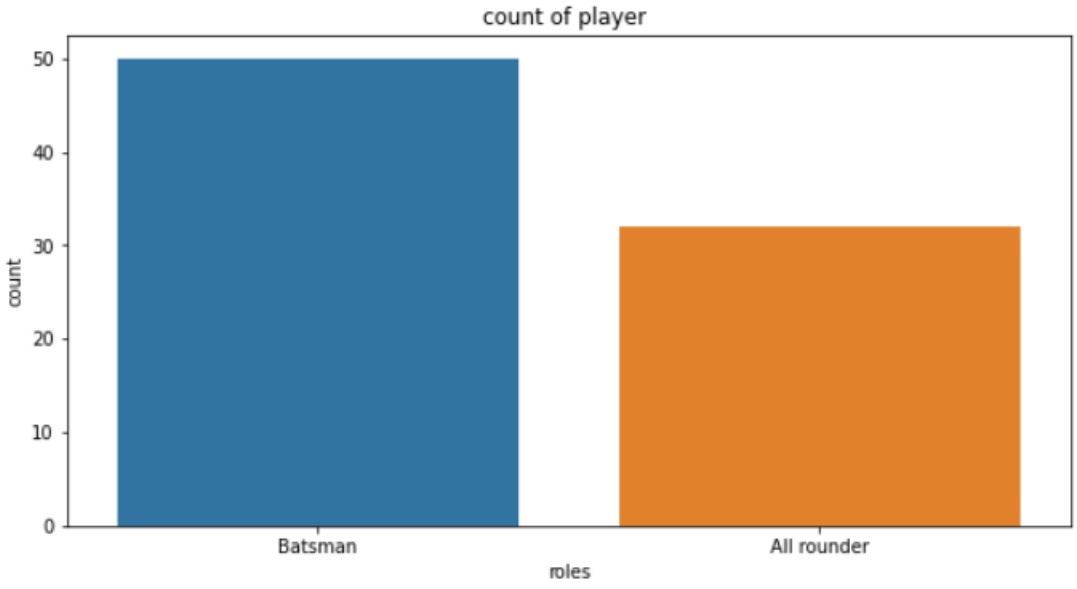
	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
60	Ruturaj Gaikwad	CSK	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	204.0	635.0	368.0

```
df[df['Season 15'] == 863]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
28	Jos Buttler	RR	Batsman	204.0	146.0	270.0	140.0	260.0	255.0	272.0	548.0	311.0	328.0	254.0	863.0

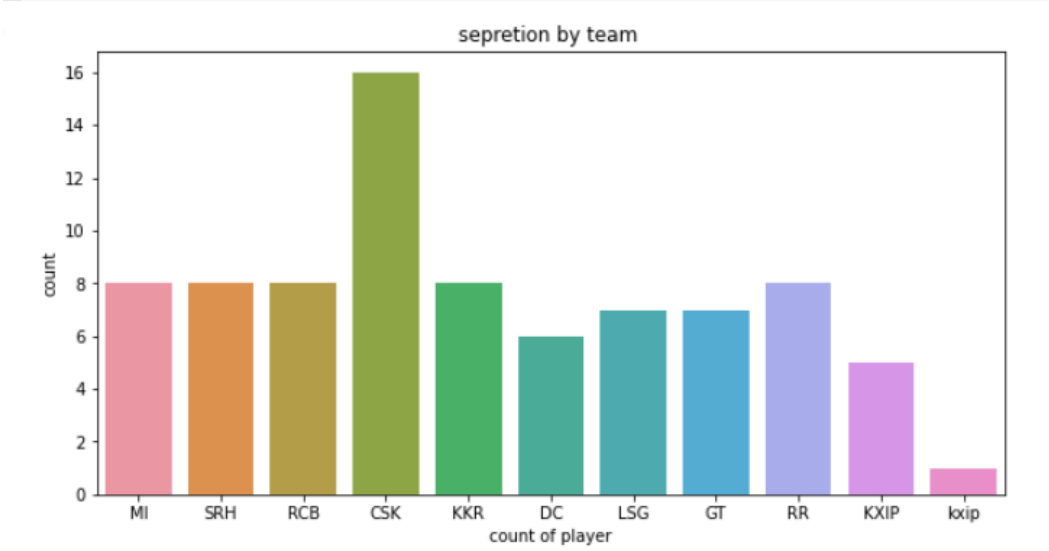
```
from matplotlib import pyplot as plt
import numpy as np
import seaborn as sns

plt.figure(figsize=(10,5))
sns.countplot(data=df,x='role')
plt.xlabel("roles")
plt.title("count of player ")
plt.show()
```



Batsman and all-rounder all are listed in our dataset , from these graph there will be crystal – clear comparision of number of batsman and all rounder although the all rounder also consider as Batsman but not all batsman are not all rounder

```
plt.figure(figsize=(10,5))
sns.countplot(data=df,x='Team Name')
plt.xlabel("count of player")
plt.title("sepretion by team ")
plt.show()
```



For sorting only batsman from the dataset:

The data set consist list of allrounder and batsman , so if some want to only the list of batsman then

```
df.loc[df['role'] == 'Batsman']
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15
0	Ewin lewis	MI	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	382.0	48.0	151.0	73.0
1	Wriddiman Saha	SRH	Batsman	68.0	6.0	70.0	362.0	249.0	270.0	234.0	122.0	86.0	214.0	131.0	317.0
2	AB De Villiers	RCB	Batsman	312.0	319.0	360.0	395.0	513.0	687.0	216.0	480.0	442.0	454.0	313.0	0.0
3	Ajinkya Rahane	CSK	Batsman	120.0	560.0	488.0	339.0	540.0	480.0	382.0	370.0	393.0	113.0	8.0	44.0
4	Ambati Raydu	CSK	Batsman	395.0	333.0	265.0	361.0	281.0	334.0	91.0	602.0	292.0	359.0	257.0	274.0
9	Brendon McCullum	CSK	Batsman	357.0	289.0	6.0	405.0	436.0	354.0	319.0	127.0	NaN	NaN	NaN	NaN
10	Chris Gayle	RCB	Batsman	608.0	773.0	708.0	196.0	491.0	227.0	200.0	368.0	490.0	288.0	193.0	0.0
12	David Miller	GT	Batsman	105.0	98.0	418.0	446.0	357.0	161.0	83.0	74.0	213.0	0.0	124.0	481.0
13	David Warnar	DC	Batsman	163.0	324.0	256.0	410.0	528.0	562.0	848.0	641.0	692.0	548.0	195.0	432.0
15	Devdatt padikal	RR	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	473.0	411.0	376.0
16	Devon conway	CSK	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	252.0
17	Dinesh Kartik	RCB	Batsman	282.0	238.0	510.0	325.0	141.0	335.0	361.0	498.0	253.0	169.0	149.0	254.0
20	Dwayne Smith	CSK	Batsman	22.0	157.0	418.0	566.0	399.0	324.0	239.0	NaN	NaN	NaN	NaN	NaN
21	Gautam Gambhir	KKR	Batsman	378.0	590.0	406.0	335.0	327.0	501.0	498.0	85.0	NaN	NaN	NaN	NaN
22	Gleen phillips	SRH	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	26.0	NaN
25	Ishan Kishan	MI	Batsman	NaN	NaN	NaN	NaN	NaN	42.0	277.0	275.0	101.0	516.0	241.0	418.0
27	Jason Roy	SRH	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	59.0	120.0	150.0	NaN
28	Jos Buttler	RR	Batsman	204.0	146.0	270.0	140.0	260.0	255.0	272.0	548.0	311.0	328.0	254.0	863.0
31	KL Rahul	LSG	Batsman	NaN	NaN	NaN	20.0	166.0	142.0	397.0	659.0	593.0	670.0	626.0	616.0
34	M.S.Dhoni	CSK	Batsman	392.0	358.0	461.0	371.0	372.0	284.0	290.0	455.0	416.0	200.0	114.0	232.0

If we want find some consistent player or batsman so first we will find mean and after that we will calculate standard deviation , from deviation we can able to understand how someone is more deviated or less deviated from the mean value. After that we will find Coefficient of Variation (CV) , from CV we able to understand who is most consistence. CV is consider as one of the important measure to find consistency .

```
mean=np.mean(df,axis=1)
print(mean)
```

```
mean=np.mean(df,axis=1
print(mean)

0     163.500000
1     177.416667
2     374.250000
3     319.750000
4     320.333333
         ...
77    104.428571
78    483.833333
79     53.000000
80    180.333333
81    198.500000
```

Now finding CV

```
cv =  round((np.std(df,axis=1) / np.mean(df,axis=1)),4)
print(cv)
```

```
0     0.7207
1     0.5813
2     0.4163
3     0.5676
4     0.3382
      ...
77    0.7681
78    0.3640
79    0.7458
80    0.4972
81    0.5928
```

Now adding that column into the data and making new dataframe

```
df.insert(15, "average", mean)
df.insert(16, "cv", cv)
df[:90]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15	average	cv
0	Ewin lewis	MI	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	382.0	48.0	151.0	73.0	163.500000	0.7207
1	Wriddiman Saha	SRH	Batsman	68.0	6.0	70.0	362.0	249.0	270.0	234.0	122.0	86.0	214.0	131.0	317.0	177.416667	0.5813
2	AB De Villiers	RCB	Batsman	312.0	319.0	360.0	395.0	513.0	687.0	216.0	480.0	442.0	454.0	313.0	0.0	374.250000	0.4163
3	Ajinkya Rahane	CSK	Batsman	120.0	560.0	488.0	339.0	540.0	480.0	382.0	370.0	393.0	113.0	8.0	44.0	319.750000	0.5676
4	Ambati Raydu	CSK	Batsman	395.0	333.0	265.0	361.0	281.0	334.0	91.0	602.0	292.0	359.0	257.0	274.0	320.333333	0.3382
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
77	Vijay Shankar	GT	All rounder	NaN	NaN	NaN	NaN	NaN	0.0	101.0	212.0	244.0	97.0	58.0	19.0	104.428571	0.7681
78	Virat Kohli	RCB	Batsman	557.0	364.0	634.0	359.0	505.0	973.0	208.0	530.0	464.0	466.0	405.0	341.0	483.833333	0.3640
79	Washigton Sunder	SRH	All rounder	NaN	NaN	NaN	NaN	NaN	NaN	9.0	65.0	1.0	111.0	31.0	101.0	53.000000	0.7458
80	Yashasvi jaiswal	RR	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	34.0	249.0	258.0	180.333333	0.4972
81	Yusuf Pathan	SRH	All rounder	283.0	194.0	332.0	268.0	312.0	361.0	143.0	260.0	40.0	189.0	0.0	0.0	198.500000	0.5928

Then finding the outliers using cv :

```
outl=df.loc[df['cv'] != 0]
outl[:90]
```

	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15	average	cv
0	Ewin lewis	MI	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	382.0	48.0	151.0	73.0	164.0	0.8058
1	Wriddiman Saha	SRH	Batsman	68.0	6.0	70.0	362.0	249.0	270.0	234.0	122.0	86.0	214.0	131.0	317.0	177.0	0.6050
2	AB De Villiers	RCB	Batsman	312.0	319.0	360.0	395.0	513.0	687.0	216.0	480.0	442.0	454.0	313.0	0.0	374.0	0.4333
3	Ajinkya Rahane	CSK	Batsman	120.0	560.0	488.0	339.0	540.0	480.0	382.0	370.0	393.0	113.0	8.0	44.0	320.0	0.5908
4	Ambati Raydu	CSK	Batsman	395.0	333.0	265.0	361.0	281.0	334.0	91.0	602.0	292.0	359.0	257.0	274.0	320.0	0.3520
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
77	Vijay Shankar	GT	All rounder	NaN	NaN	NaN	NaN	NaN	0.0	101.0	212.0	244.0	97.0	58.0	19.0	104.0	0.8212
78	Virat Kohli	RCB	Batsman	557.0	364.0	634.0	359.0	505.0	973.0	208.0	530.0	464.0	466.0	405.0	341.0	484.0	0.3788
79	Washigton Sunder	SRH	All rounder	NaN	NaN	NaN	NaN	NaN	NaN	9.0	65.0	1.0	111.0	31.0	101.0	53.0	0.8055
80	Yashasvi jaiswal	RR	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	34.0	249.0	258.0	180.0	0.5742
81	Yusuf Pathan	SRH	All rounder	283.0	194.0	332.0	268.0	312.0	361.0	143.0	260.0	40.0	189.0	0.0	0.0	198.0	0.6170

77 rows × 17 columns

We removed outliers from data using ‘cv’ and then creating new data with zero cv which is more crystal clear to analyze player performance .

Then sorting data by average and exporting data into new excel sheet.

```
outlier=outl.sort_values(by=['average'])
outlier[:90]
outlier.to_csv("/C:/Users/ACER/OneDrive/Desktop/final.csv",index=Fals)
```

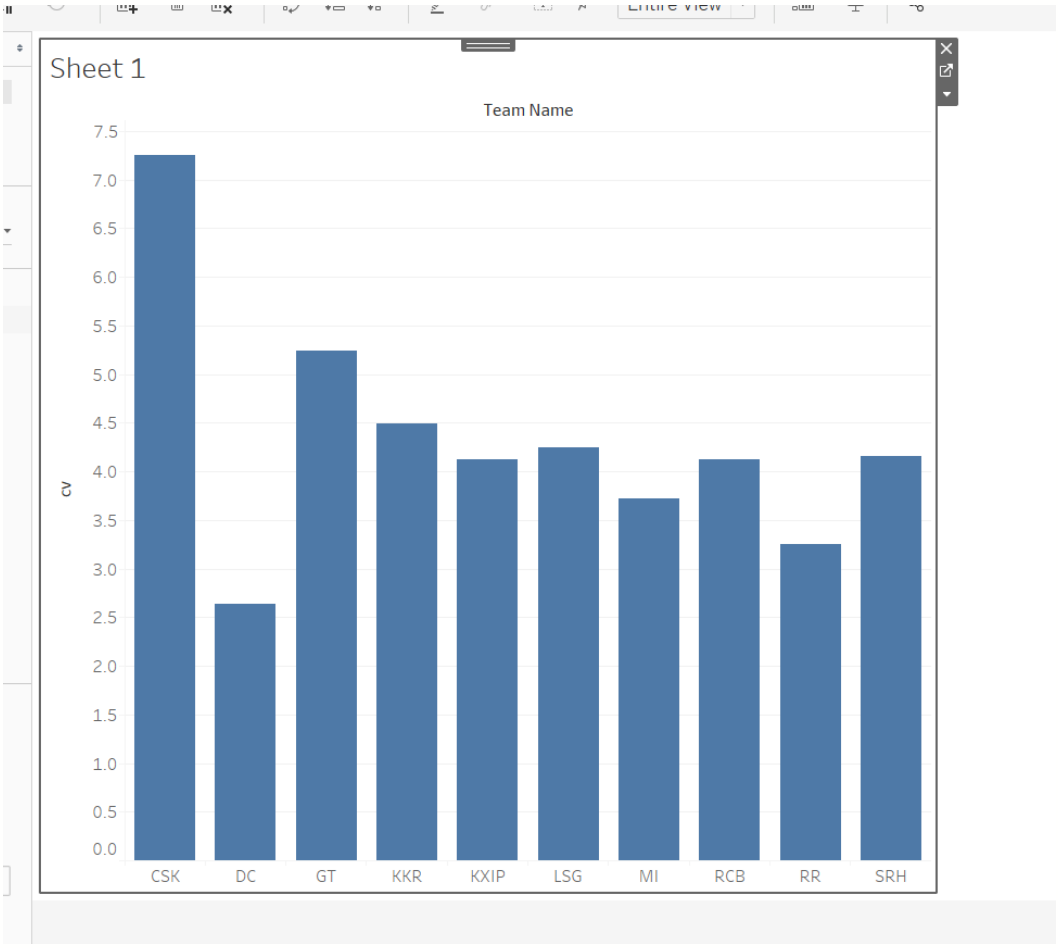
	Player Name	Team Name	role	Season 4	Season 5	Season 6	Season 7	Season 8	Season 9	Season 10	Season 11	Season 12	Season 13	Season 14	Season 15	average	cv
42	Mitchell Satner	CSK	All rounder	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	32.0	0.0	22.0	54.0	27.0	0.7191
39	Marcus stonis	LSG	All rounder	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	6.0	38.0	44.0	29.0	0.5686
26	Jason holder	LSG	All rounder	NaN	NaN	NaN	NaN	0.0	NaN	NaN	16.0	22.0	66.0	85.0	58.0	41.0	0.7366
7	Ben Cutting	MI	All rounder	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.0	65.0	51.0	96.0	18.0	48.0	0.6710
48	R Ashwin	RR	All rounder	8.0	30.0	2.0	18.0	35.0	52.0	41.0	102.0	42.0	37.0	44.0	191.0	50.0	0.9741
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
15	Devdatt padikal	RR	Batsman	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	473.0	411.0	376.0	420.0	0.0955
31	KL Rahul	LSG	Batsman	NaN	NaN	NaN	20.0	166.0	142.0	397.0	659.0	593.0	670.0	626.0	616.0	432.0	0.5623
13	David Warnar	DC	Batsman	163.0	324.0	256.0	410.0	528.0	562.0	848.0	641.0	692.0	548.0	195.0	432.0	467.0	0.4292
66	Shikar Dhawan	KXIP	Batsman	400.0	569.0	311.0	377.0	353.0	501.0	479.0	497.0	521.0	618.0	587.0	460.0	473.0	0.1953
78	Virat Kohli	RCB	Batsman	557.0	364.0	634.0	359.0	505.0	973.0	208.0	530.0	464.0	466.0	405.0	341.0	484.0	0.3788

77 rows × 17 columns

2.3 Using Tableu

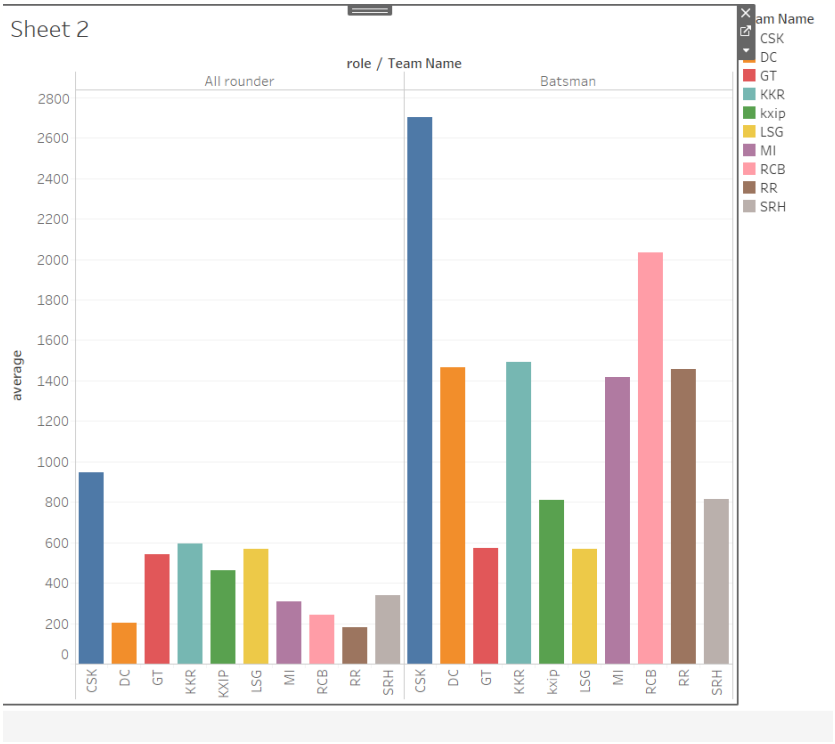
We have converted data set into new by removing outlier and then imported in tableue. We have created calculated field for batsman to remove poor performance , we have labelled player who have all season average greater than 350 as ‘best’. And less than 350 and greater than 250 as ‘good’ , who has average less than 250 ‘poor’ and they are outliers.

Fig 1 for showing Average CV of player of teams:

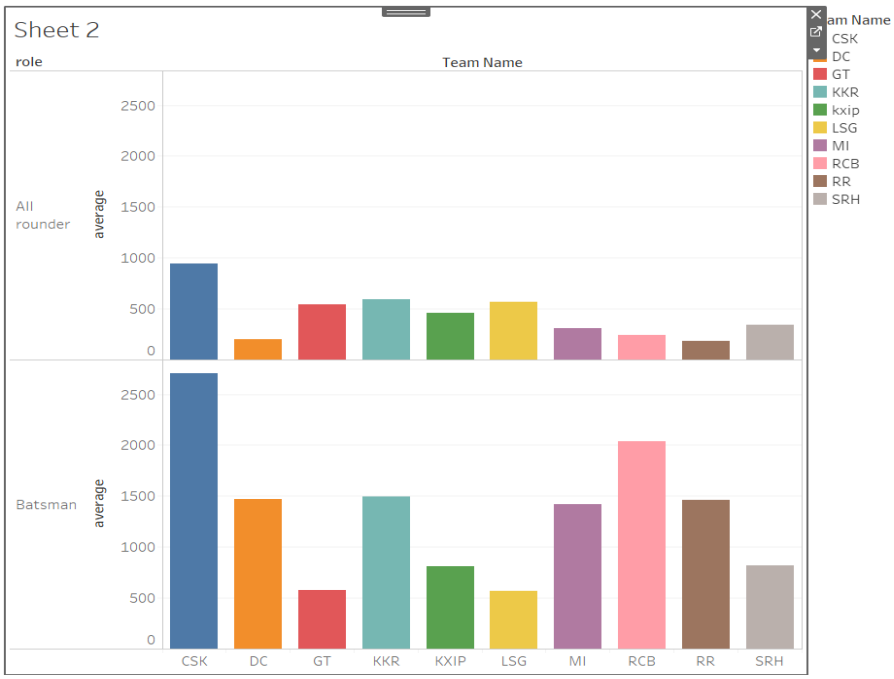


From the above graph the team DC has player who has least CV and CSK has player who’s CV is highest amoung other all

Fig 1.1 for showing Average total of All rounder and Batsman as per their particular team

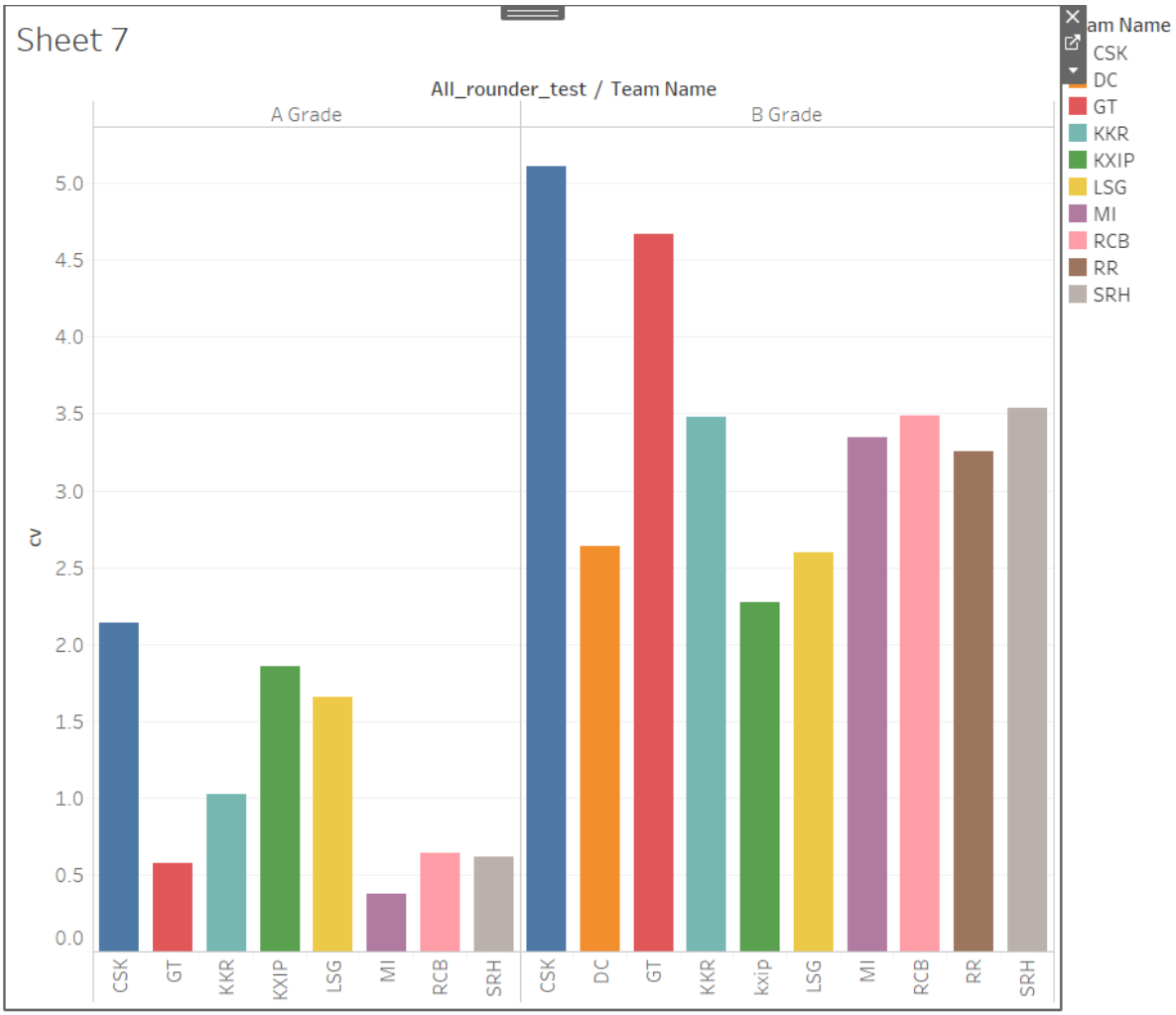


From the above graph we can see that CSK has player which have highest runs amoung the season by both batsman and all-rounder, after that RCB batsman scores more runs.



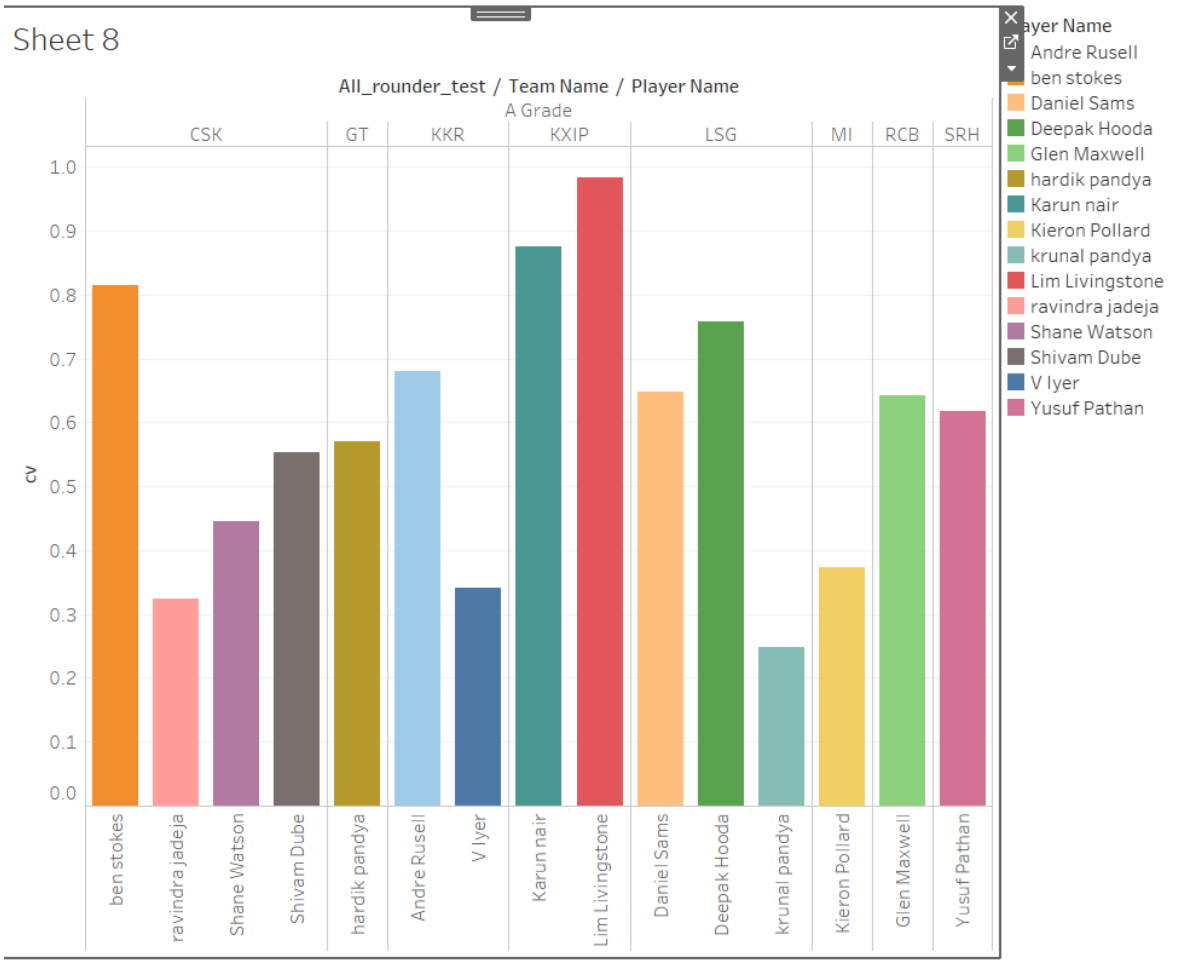
Next , we have created second created calculated field to sort and remove outlier that because of all rounders . we created calculated field as who has greater than 150 runs so they are ‘a grade ‘ and less than 150 is ‘b grade’ named .So from this we got outlier and selected some main batsman this is a feature selection .because of this we got best all rounder so we conclude anything easily

Fig 1.2 showing ‘A Grade’ and ‘B Grade’ players average cv by teams



from above graph we can see that csk has high cv in ‘a grade’ and ‘b grade’ because they have highest no of all rounder in their team and mi had lowest cv of their ‘a grade ‘ all rounders . and then GT team players has also low cv . so this become simply to predict and analysis

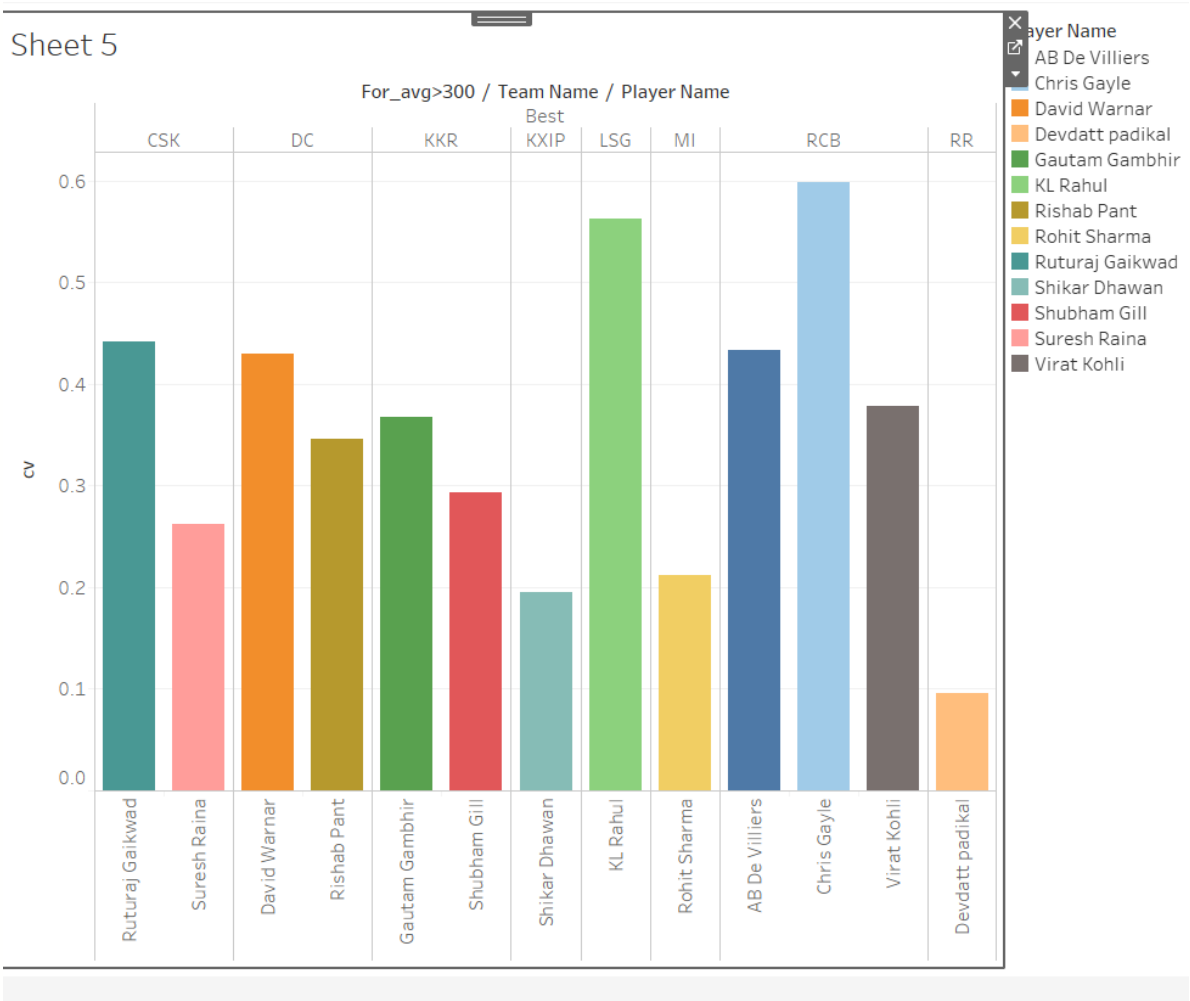
Fig 1.3 for showing number of ‘A Grade’ All rounder in each team and showing there CV



First we have sorted A Grade all rounder from the create calculated field as showing ‘A Grade’ and ‘B Grade’. The graph shows CSK has highest number of ‘A Grade’ all rounder and by graph we can see that the Ravindra Jadeja , V.Iyer and Kruanl pandya has the lowest CV , we can conclude that Ravindra Jadeja is best all rounder as his consistency,

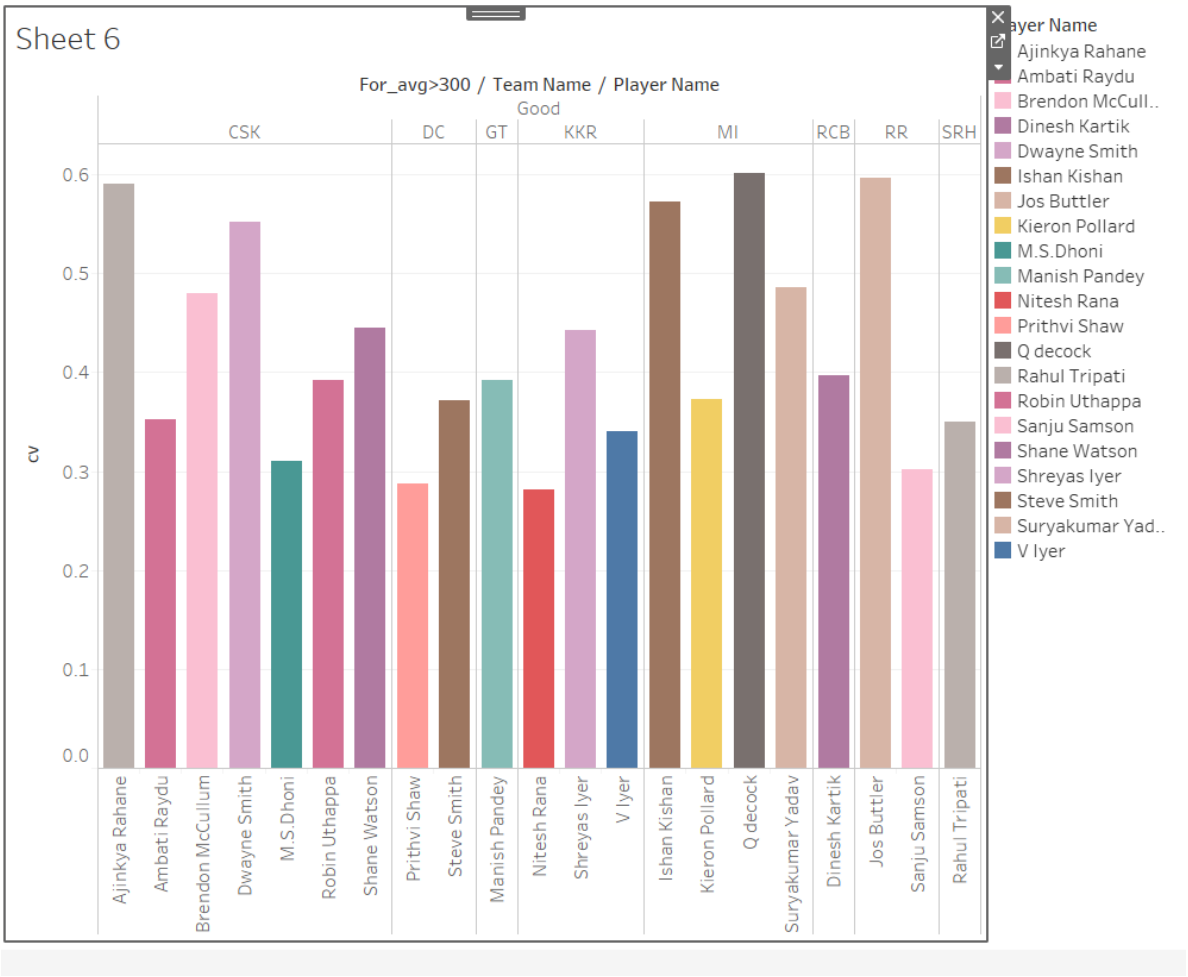
played more season and still one of the most consistence amoung all ALL-rounders.

Fig 1.4 for showing player average more than 300



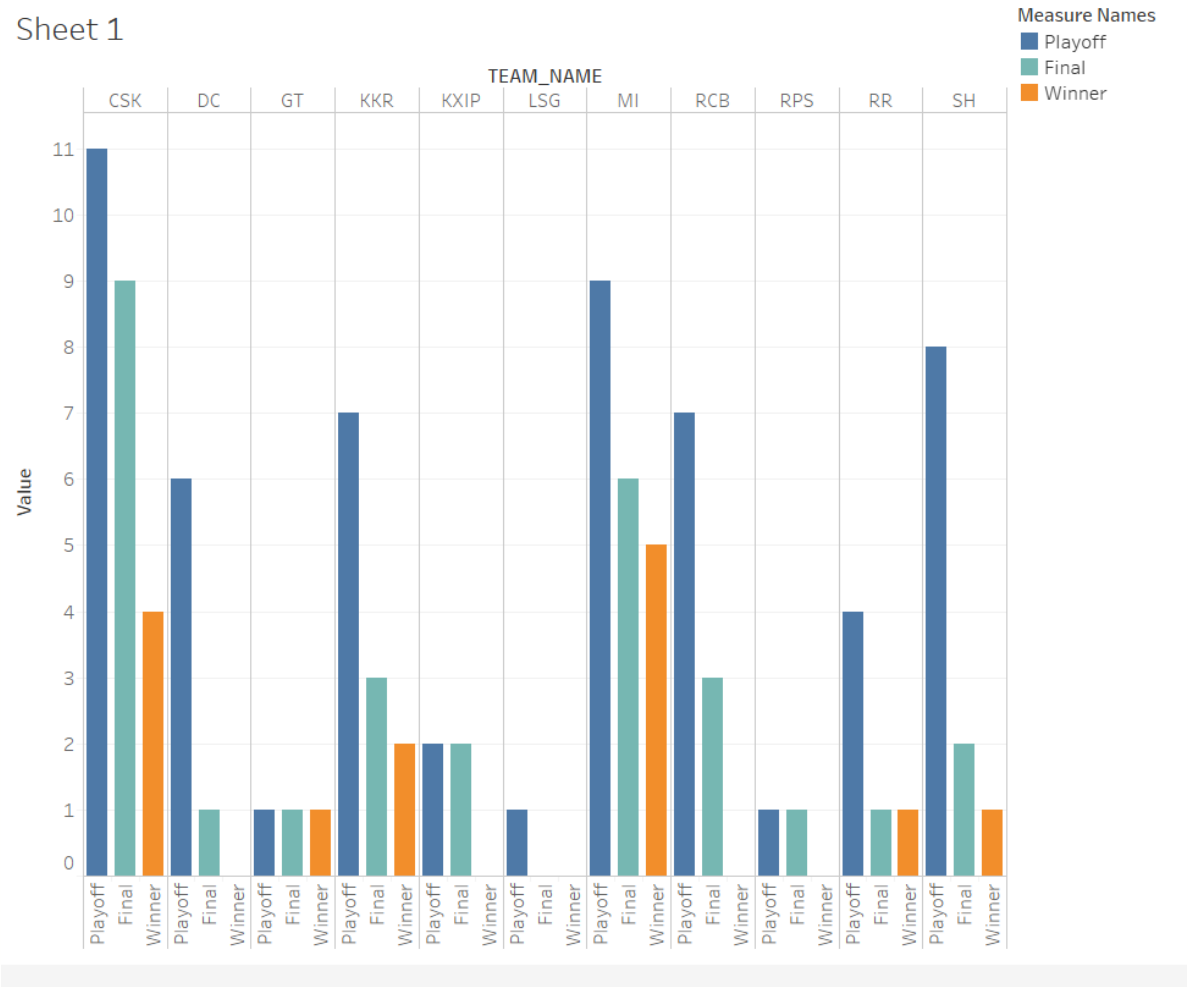
The above graph shows the number of player of particular team whose average is greater than 350. RCB has more number of batsman and RR , KXIP and MI has only one player fitted in list but there CV is less means they are consistence as they are always in top of list in run scorer in each season which they played.

Fig 1.5 for showing second class player mean we labelled them as ‘Good’



We sorted player who has average season runs between 250 to 350 and Labelled as them ‘Good’ . In this graph we can see that CSK has more number average players and after that second team is MI which have highest number of ‘Good’ player. Both team perform very good in all season and over all ipl. in graph we can see that they has average cv so their player perforemed best .

Fig 1.6 for Comparing team performance



We have ploted bar gragh for comparing how many times team has qualified for plaoff , Final and Winner. We can see here CSK has played most number of playoff in their carrier and after that MI has played most number of playoff and MI has win final 5 times and CSK won 4 times, and DC , KXIP , LSG , RCB does not win single final in their carrier .and KXIP is even didn't qualified for final .

CONCLUSION

From the above charts we can say that virat kohli has best performance in ipl as his consistency is so good, so he is one of best players of ipl . As we can retrived orange cap of all season we seen that there is 3 time orange cap title winned by virat kohli and he has more than 350 average in all season .The best all rounder is Ravindra Jadeja as he is most consistence and he has good number of runs with less CV

From the insights of all dashboard we can say that the CSK and MI are the Best amoung all teams as CSK and MI has played more matches in playoff , final and CSK has able to win 4 times the IPL where the MI have won 5 times. Many player of CSK has average season runs more than other team player and CSK all rounders has also good performance and same for MI
Finally we can conclude that the CSK and MI has been successful team in over all season of IPL

REFERENCE

- https://www.mykhel.com/
- https://www.mykhel.com/
- https://practice.geeksforgeeks.com
- https://www.iplt20.com/
- https://www.cricbuzz.com/
- https://en.wikipedia.org/wiki/Indian_Premier_League