This Document containts the data visualization of IPL datasets using matplotlib and seaborn

```
#Import numpy
import numpy as np
```

```
#Seasons into List & dict

Seasons = ["2015","2016","2017","2018","2019","2020","2021","2022","2023","2024"]
Sdict = {"2015":0,"2016":1,"2017":2,"2018":3,"2019":4,"2020":5,"2021":6,"2022":7,"2023":8,"2024":9}
```

```
#Players into List & dict

Players = ["Sachin","Rahul","Smith","Sami","Pollard","Morris","Samson","Dhoni","Kohli","Sky"]
Pdict = {"Sachin":0,"Rahul":1,"Smith":2,"Sami":3,"Pollard":4,"Morris":5,"Samson":6,"Dhoni":7,"Kohli":8,"Sky":9}
```

```
# Convert Salaries to array

#Salaries
Sachin_Salary = [15946875,17718750,19490625,21262500,23034375,24806250,25244493,27849149,30453805,23500000]
Rahul_Salary = [12000000,12744189,13488377,14232567,14976754,16324500,18038573,19752645,21466718,23180790]
Smith_Salary = [4621800,5828090,13041250,14410581,15779912,14500000,16022500,17545000,19067500,20644400]
Sami_Salary = [3713640,4694041,13041250,14410581,15779912,17149243,18518574,19450000,22407474,22458000]
Pollard_Salary = [4493160,4806720,6061274,13758000,15202590,16647180,18091770,19536360,20513178,21436271]
Morris_Salary = [3348000,4235220,12455000,14410581,15779912,14500000,16022500,17545000,19067500,20644400]
Samson_Salary = [3144240,3380160,3615960,4574189,13520500,14940153,16359805,17779458,18668431,20068563]
Dhoni_Salary = [0,0,4171200,4484040,4796880,6053663,15506632,16669630,17832627,18995624]
Kohli_Salary = [0,0,0,4822800,5184480,5546160,6993708,16402500,17632688,18862875]
Sky_Salary = [3031920,3841443,13041250,14410581,15779912,14200000,15691000,17182000,18673000,15000000]

#Matrix
Salary = np.array([Sachin_Salary, Rahul_Salary, Smith_Salary, Sami_Salary, Pollard_Salary, Morris_Salary, Samson_Salary, Dhoni_Salary, Kohli_Salary, Sky_Salary])
```

```
# Convert Games to array

#Games
Sachin_G = [80,77,82,82,73,82,58,78,6,35]
Rahul_G = [82,57,82,79,76,72,60,72,79,80]
Smith_G = [79,78,75,81,76,79,62,76,77,69]
Sami_G = [80,65,77,66,69,77,55,67,77,40]
Pollard_G = [82,82,82,79,82,78,54,76,71,41]
Morris_G = [70,69,67,77,70,77,57,74,79,44]
Samson_G = [78,64,80,78,45,80,60,70,62,82]
Dhoni_G = [35,35,80,74,82,78,66,81,81,27]
Kohli_G = [40,40,40,81,78,81,39,0,10,51]
Sky_G = [75,51,51,79,77,76,49,69,54,62]

#Matrix
Games = np.array([Sachin_G, Rahul_G, Smith_G, Sami_G, Pollard_G, Morris_G, Samson_G, Dhoni_G, Kohli_G, Sky_G])
```

```
# Convert Points to array

#Points
Sachin_PTS = [2832,2430,2323,2201,1970,2078,1616,2133,83,782]
Rahul_PTS = [1653,1426,1779,1688,1619,1312,1129,1170,1245,1154]
Smith_PTS = [2478,2132,2250,2304,2258,2111,1683,2036,2089,1743]
Sami_PTS = [2122,1881,1978,1504,1943,1970,1245,1920,2112,966]
Pollard_PTS = [1292,1443,1695,1624,1503,1784,1113,1296,1297,646]
Morris_PTS = [1572,1561,1496,1746,1678,1438,1025,1232,1281,928]
Samson_PTS = [1258,1104,1684,1781,841,1268,1189,1186,1185,1564]
Dhoni_PTS = [903,903,1624,1871,2472,2161,1850,2280,2593,686]
Kohli_PTS = [597,597,597,1361,1619,2026,852,0,159,904]
Sky_PTS = [2040,1397,1254,2386,2045,1941,1082,1463,1028,1331]

#Matrix
Points = np.array([Sachin_PTS, Rahul_PTS, Smith_PTS, Sami_PTS, Pollard_PTS, Morris_PTS, Samson_PTS, Dhoni_PTS, Kohli_PTS, Sky_PTS])
```

```
# Print Seasons (Dictionary)

for i in Sdict:
  print('Season year:',i)
```

```
Season year: 2015
Season year: 2016
Season year: 2017
Season year: 2018
Season year: 2019
Season year: 2020
Season year: 2021
Season year: 2022
Season year: 2023
Season year: 2024
```

```
# Print Players (List)

for i in Players:
  print('Player Name:',i)
```

```
Player Name: Sachin
Player Name: Rahul
Player Name: Smith
Player Name: Sami
Player Name: Pollard
Player Name: Morris
Player Name: Samson
Player Name: Dhoni
Player Name: Kohli
Player Name: Sky
```

```
print(type(Salary))
print("\n")
```

```
print(Salary[Pdict['Dhoni']])
```

```
<class 'numpy.ndarray'>


[       0        0  4171200  4484040  4796880  6053663 15506632 16669630
 17832627 18995624]
```

```
print(type(Games))
print('\n')
print(Games[Pdict['Dhoni']])
```

```
<class 'numpy.ndarray'>


[35 35 80 74 82 78 66 81 81 27]
```

```
print(type(Points))
print('\n')
print(Points[Pdict['Dhoni']])
```

```
<class 'numpy.ndarray'>


[ 903  903 1624 1871 2472 2161 1850 2280 2593  686]
```

```
# Array Shapes

print(Salary.shape)
print(Games.shape)
print(Points.shape)
```

```
(10, 10)
(10, 10)
(10, 10)
```

```
# Slicing and Access

print(Points[-3:-1])
print('\n')
print(Points[-3,-1])
```

```
[[ 903  903 1624 1871 2472 2161 1850 2280 2593  686]
 [ 597  597  597 1361 1619 2026  852    0  159  904]]


686
```

```
# Get Salary, Games, Points of Sachin

print('Salary of Sachin is ',Salary[Pdict['Sachin']])
print('\n')
print('Games of Sachin is ',Games[Pdict['Sachin']])
print('\n')
print('Points of Sachin is ',Points[Pdict['Sachin']])


#Similarly we can get details of individual player using Pdict
```

```
Salary of Sachin is  [15946875 17718750 19490625 21262500 23034375 24806250 25244493 27849149
 30453805 23500000]


Games of Sachin is  [80 77 82 82 73 82 58 78  6 35]


Points of Sachin is  [2832 2430 2323 2201 1970 2078 1616 2133   83  782]
```

Matplotlib Pyplot

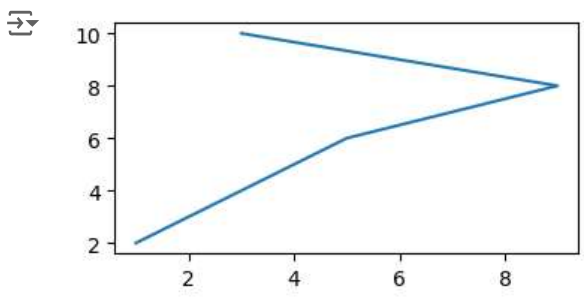
Matplotlib

It is a popular data visualization library in Python. It's often used for creating static, interactive, and animated visualizations in Python.
Matplotlib allows you to generate plots, histograms, bar charts, scatter plots, etc., with just a few lines of code.

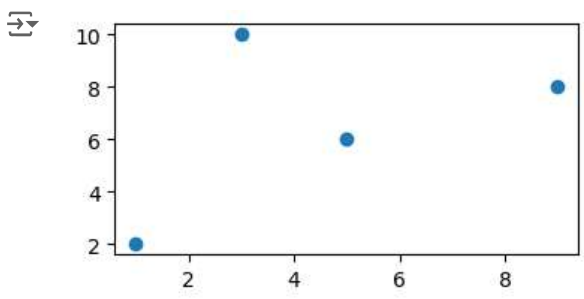
Most of the matplotlib utilities lies under the 'pyplot' submodule, and are usually imported under the 'plt' alias.

```
import matplotlib.pyplot as plt
```

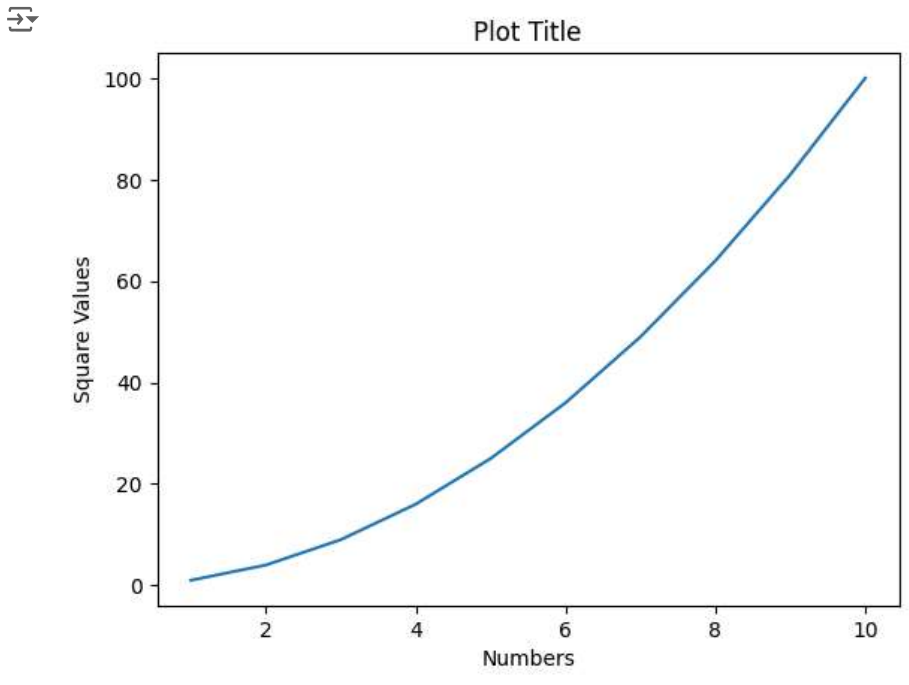
```
#Example
x=np.array([1,5,9,3])
y=np.array([2,6,8,10])

plt.figure(figsize=(4,2))
plt.plot(x,y)
plt.show()
```




```
#Example
x=np.array([1,5,9,3])
y=np.array([2,6,8,10])
plt.figure(figsize=(4,2))
plt.plot(x,y,'o')
plt.show()
```




```
x = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
y = np.array([1,4,9,16,25,36,49,64,81,100])

plt.title("Plot Title")
plt.xlabel("Numbers")
plt.ylabel("Square Values")

plt.plot(x, y)
plt.figure(figsize=(6,4))
plt.show()
```

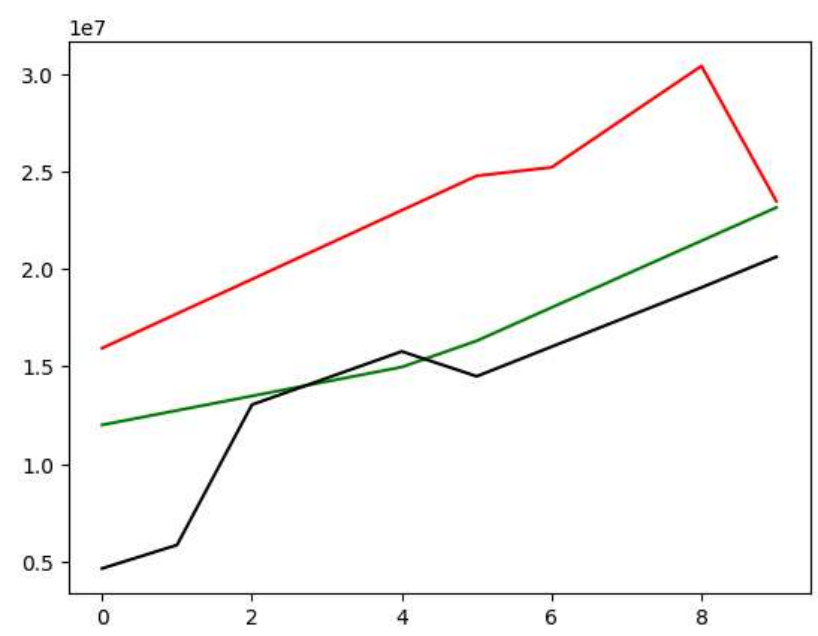



```
<Figure size 600x400 with 0 Axes>
```

1. Color

- Red: 'r'
- Green: 'g'
- Blue: 'b'
- Cyan: 'c'
- Magenta: 'm'
- Yellow: 'y'
- Black: 'k'
- White: 'w'

```
plt.plot(Salary[Pdict['Sachin']],color='r')
plt.plot(Salary[Pdict['Rahul']],color='g')
plt.plot(Salary[Pdict['Smith']],color='k')

plt.figure(figsize=(4,2))
plt.show()
```

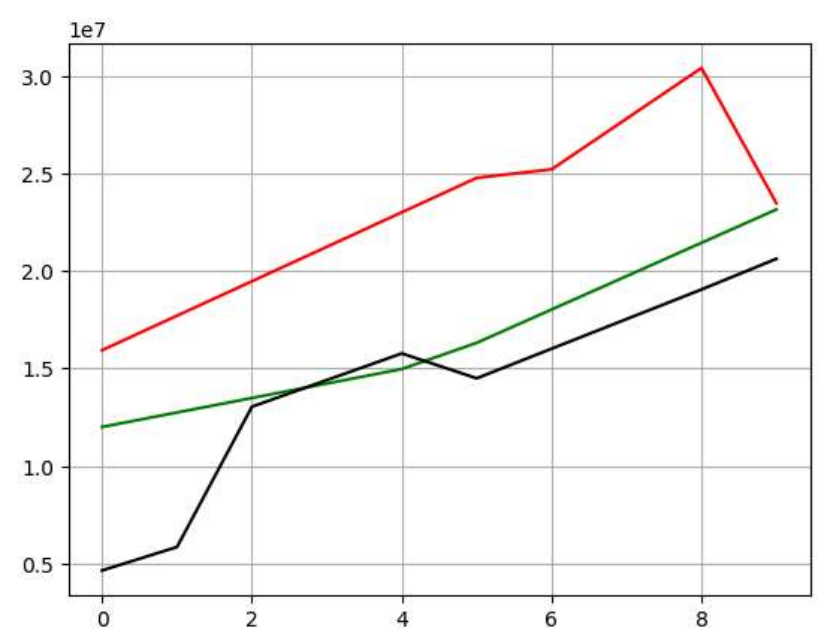



```
<Figure size 400x200 with 0 Axes>
```

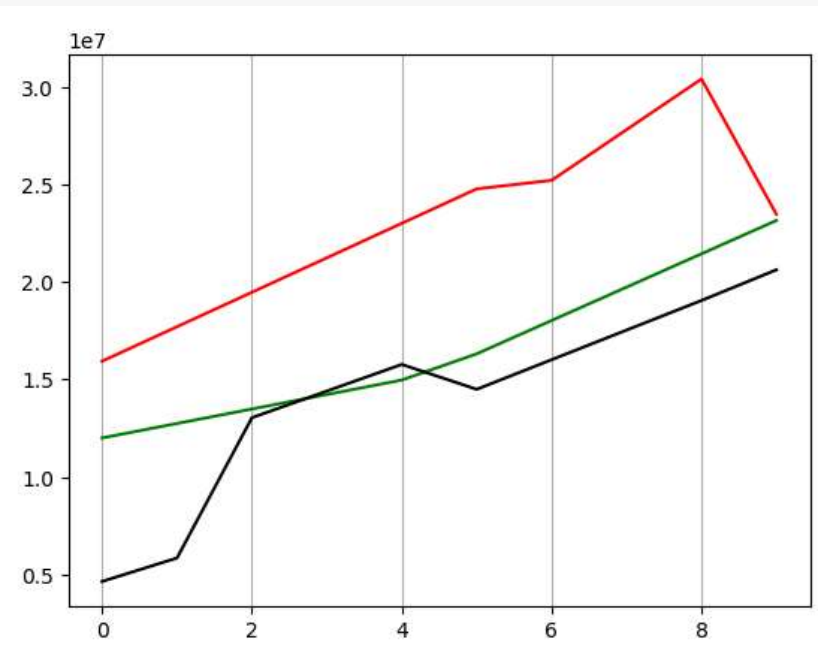
2. Grid Lines

add grid lines to the plot.

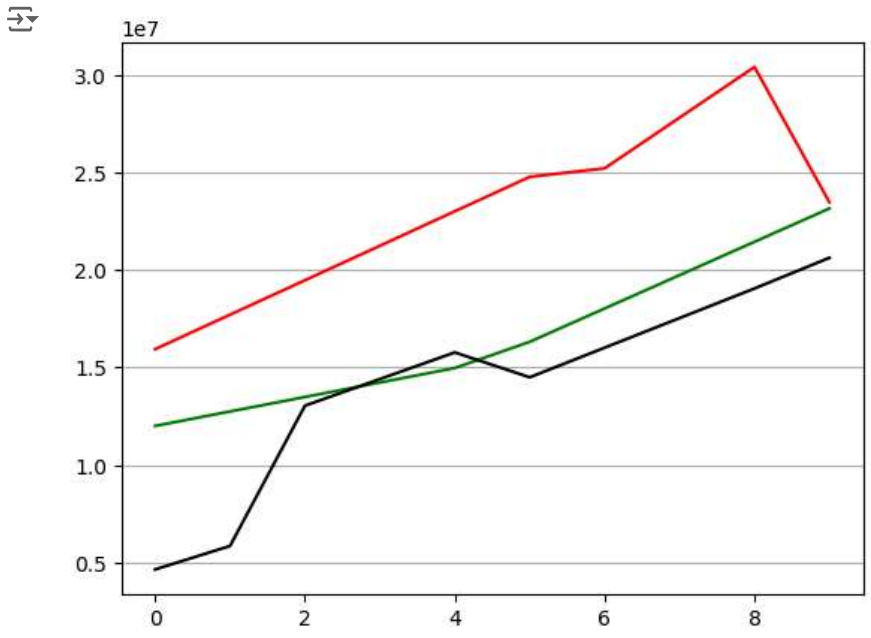
```
plt.plot(Salary[Pdict['Sachin']],color='r')
plt.plot(Salary[Pdict['Rahul']],color='g')
plt.plot(Salary[Pdict['Smith']],color='k')

plt.grid()
plt.show()
```



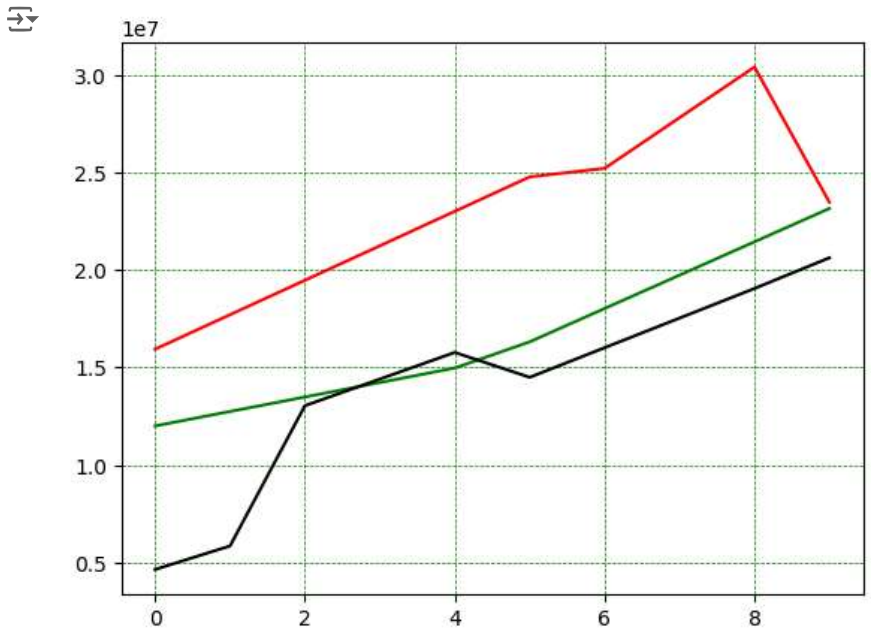

```
plt.plot(Salary[Pdict['Sachin']],color='r')
plt.plot(Salary[Pdict['Rahul']],color='g')
plt.plot(Salary[Pdict['Smith']],color='k')

plt.grid(axis='x')

plt.show()
```




```
plt.plot(Salary[Pdict['Sachin']],color='r')
plt.plot(Salary[Pdict['Rahul']],color='g')
plt.plot(Salary[Pdict['Smith']],color='k')

plt.grid(axis='y')

plt.show()
```




```
plt.plot(Salary[Pdict['Sachin']],color='r')
plt.plot(Salary[Pdict['Rahul']],color='g')
plt.plot(Salary[Pdict['Smith']],color='k')

plt.grid(color = 'green', linestyle = '--', linewidth = 0.5)

plt.show()
```

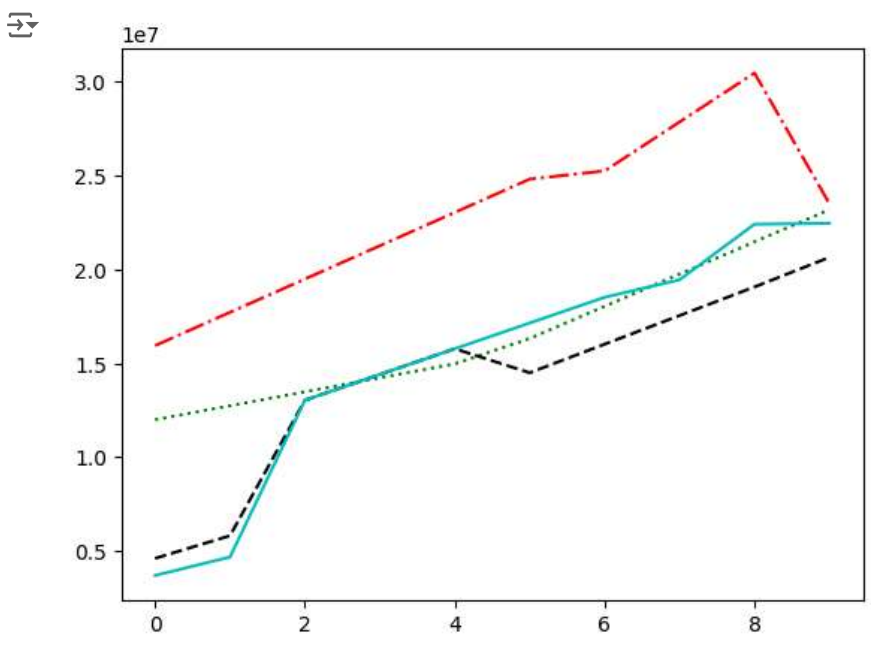


3. Line

LineStyle (ls)

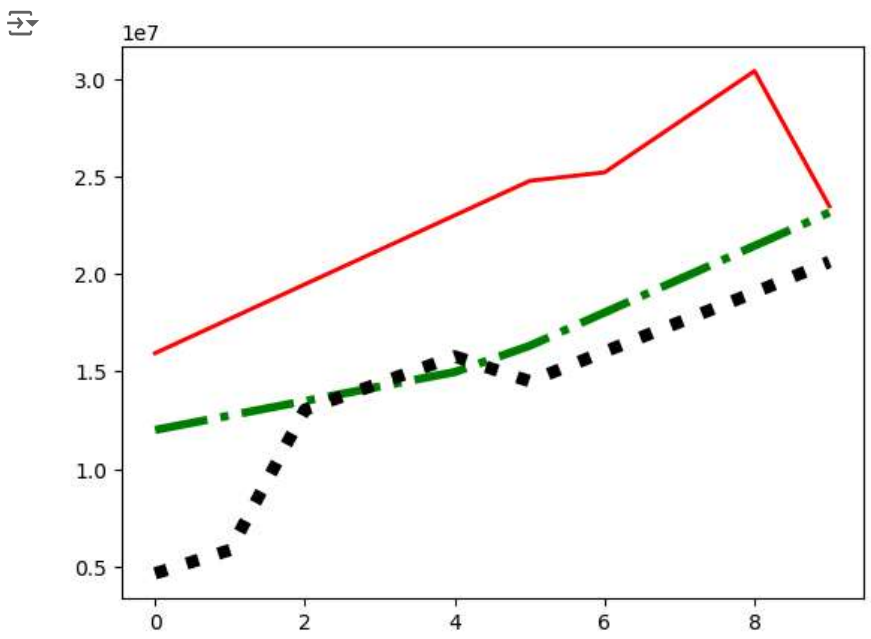
- Solid line: '-'
- Dotted line: ':'
- Dashed line: '--'
- Dashed/Dotted line: '-.'

```
plt.plot(Salary[Pdict['Sachin']],color='r',ls='-.')
plt.plot(Salary[Pdict['Rahul']],color='g',ls=':')
plt.plot(Salary[Pdict['Smith']],color='k',ls='--')
plt.plot(Salary[Pdict['Sami']],color='c',ls='-')

plt.show()
```



Line Width (lw)

```
plt.plot(Salary[Pdict['Sachin']],ls='-',color='r',lw=2)
plt.plot(Salary[Pdict['Rahul']],ls='-.',color='g',lw=4)
plt.plot(Salary[Pdict['Smith']],ls=':',color='k',lw=6)

plt.show()
```



4. Marker

- 'o' Circle
- '*' Star
- '.' Point
- ',' Pixel
- 'x' X
- 'X' X (filled)
- '+' Plus
- 'P' Plus (filled)
- 's' Square
- 'D' Diamond
- 'd' Diamond (thin)
- 'p' Pentagon
- 'H' Hexagon
- 'h' Hexagon
- 'v' Triangle Down
- '^' Triangle Up
- '<' Triangle Left
- '>' Triangle Right
- '1' Tri Down
- '2' Tri Up
- '3' Tri Left
- '4' Tri Right
- '|' Vline
- '_' Hline

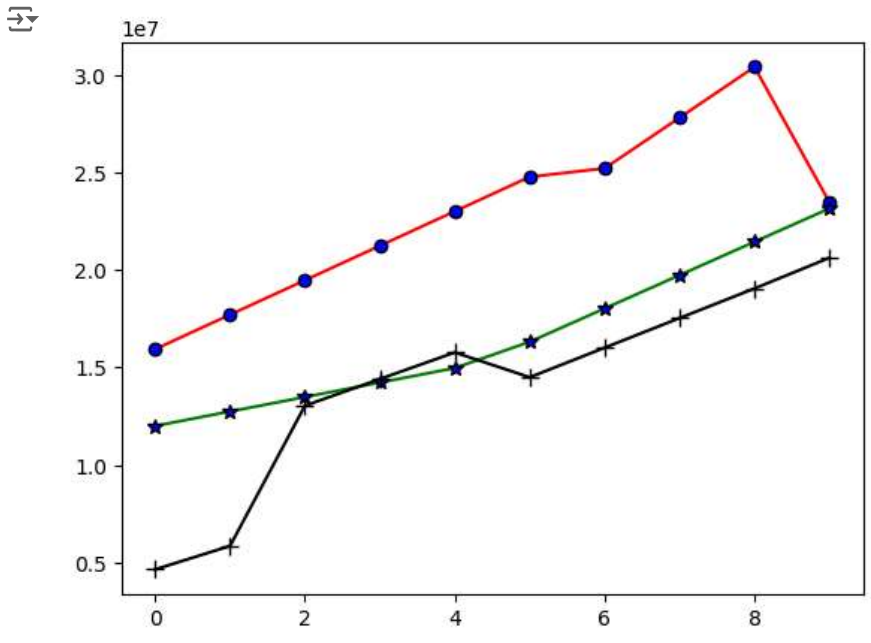
Marker Size (ms)

MarkerEdgeColor (mec)

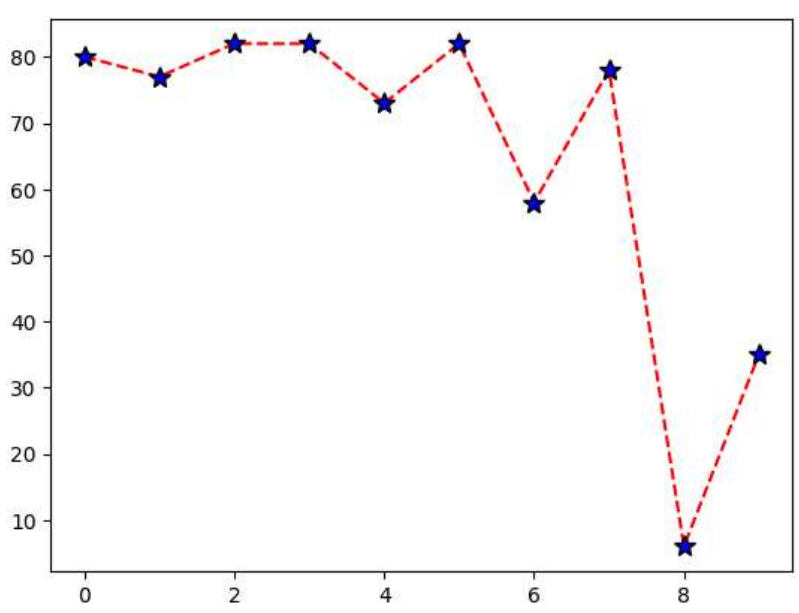
MarkerFaceColor (mfc)

Use both the mec and mfc arguments to color the entire marker

```
plt.plot(Salary[Pdict['Sachin']],color='r',marker='o',ms=6,mec='k',mfc='b')
plt.plot(Salary[Pdict['Rahul']],color='g',marker='*',ms=7,mec='k',mfc='b')
plt.plot(Salary[Pdict['Smith']],color='k',marker='+',ms=8,mec='k',mfc='b')

plt.show()
```



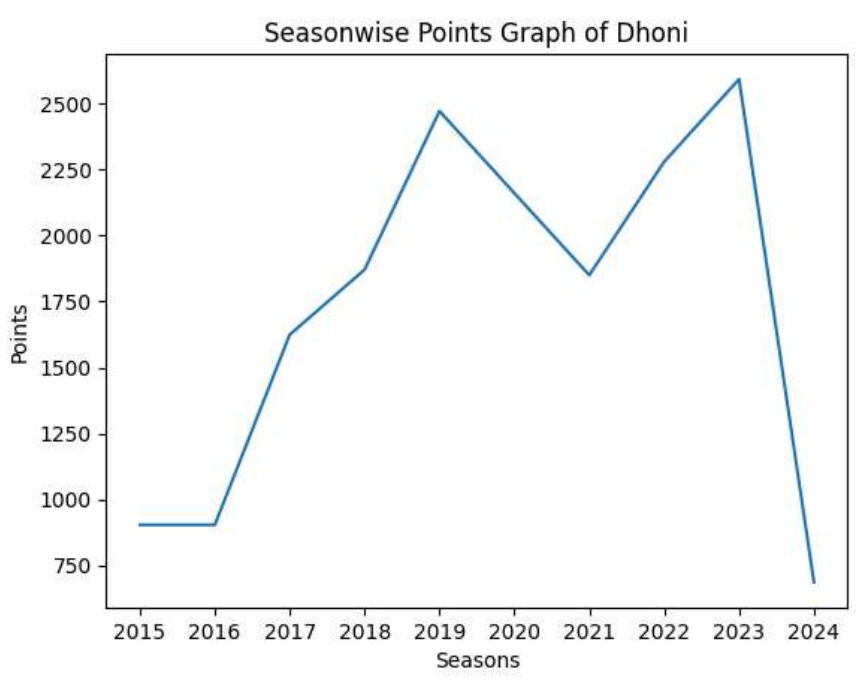

```
#Plot Games of Sachin
plt.plot(Games[Pdict['Sachin']],color='red',ls='--',marker='*',mec='k',mfc='b',ms='10')

plt.show()
```



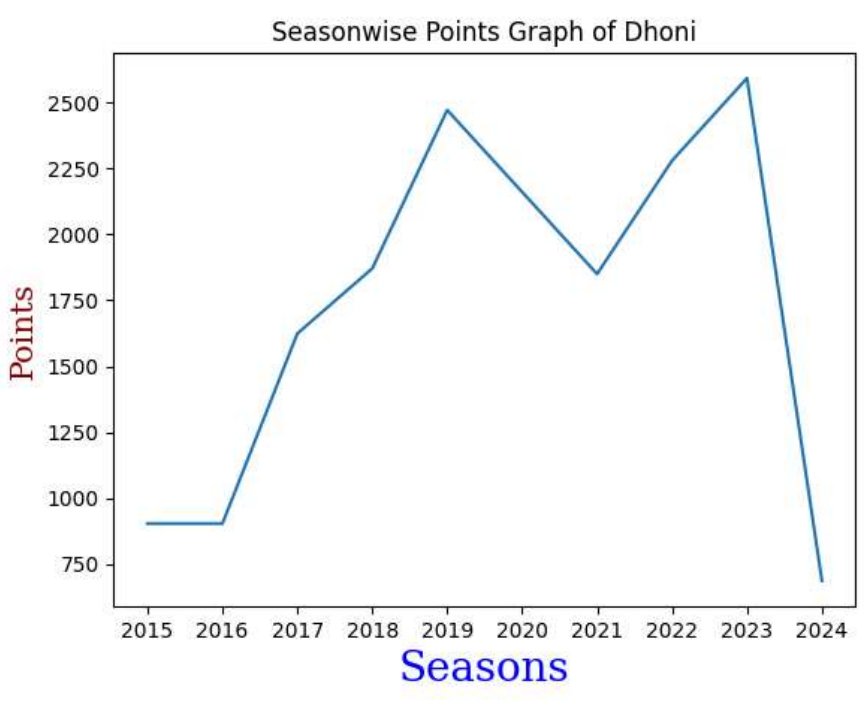
5. Labels

use the xlabel() and ylabel() functions to set a label for the x- and y-axis.

```
x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])

plt.title('Seasonwise Points Graph of Dhoni')
plt.xlabel('Seasons')
plt.ylabel('Points')
plt.plot(x,y)
plt.show()
```

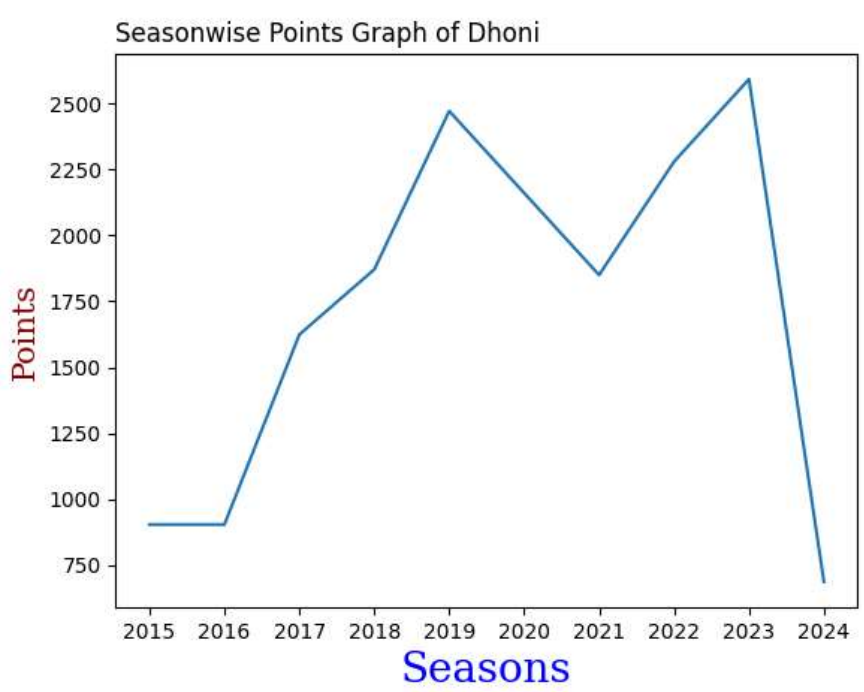



```
x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])

font1 = {'family':'serif','color':'blue','size':20}
font2 = {'family':'serif','color':'darkred','size':15}

plt.title('Seasonwise Points Graph of Dhoni')
plt.xlabel('Seasons',fontdict=font1)
plt.ylabel('Points',fontdict=font2)

plt.plot(x,y)
plt.show()
```




```
x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])

font1 = {'family':'serif','color':'blue','size':20}
font2 = {'family':'serif','color':'darkred','size':15}
```

```
plt.title('Seasonwise Points Graph of Dhoni',loc='left')
plt.xlabel('Seasons',fontdict=font1)
plt.ylabel('Points',fontdict=font2)

plt.plot(x,y)
plt.show()
```

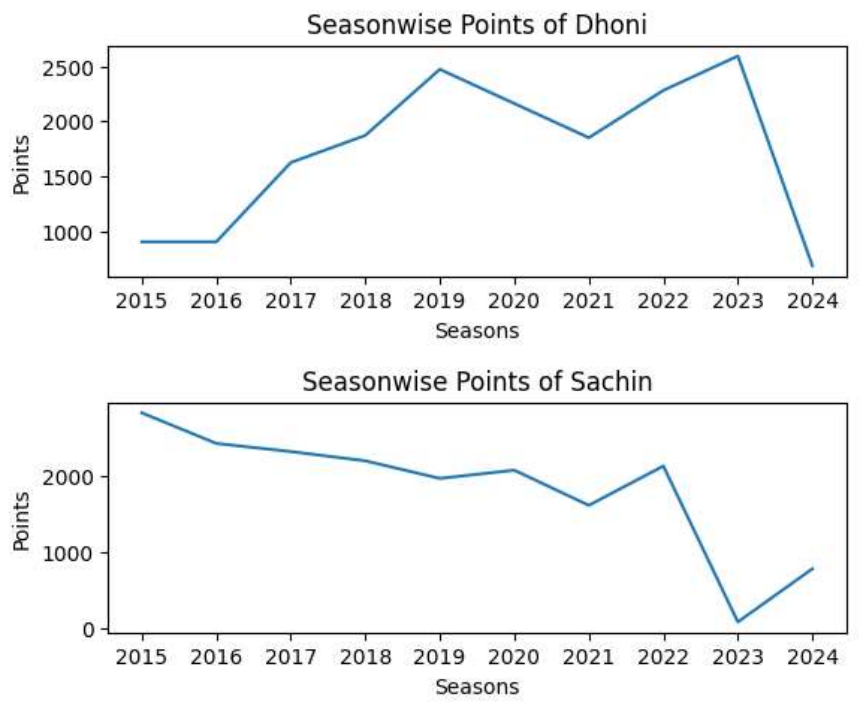


6. Subplot

draw multiple plots in one figure.

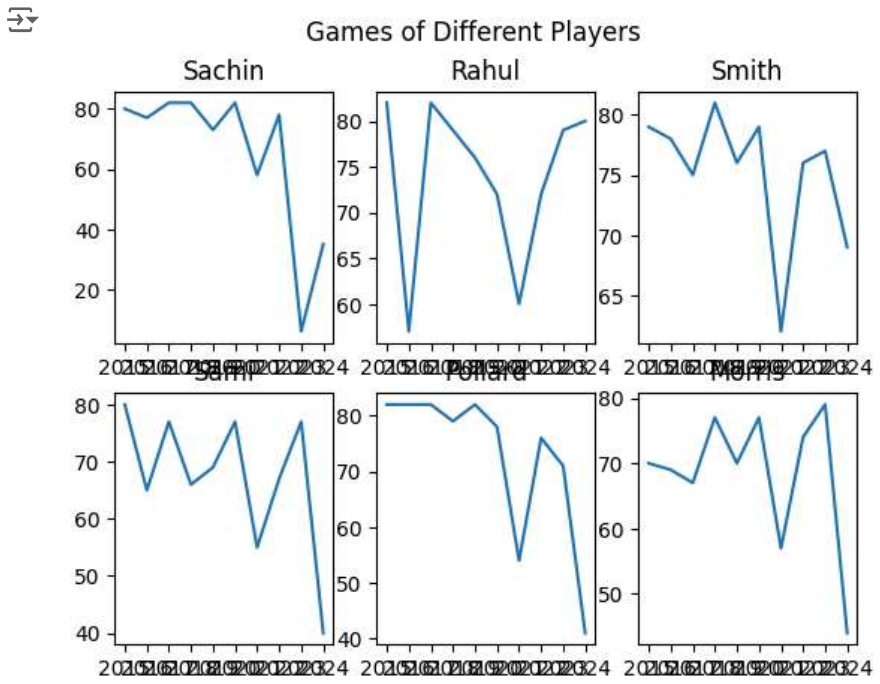
```
#Plot1:
x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])
plt.figure(figsize=(14,2))
plt.subplot(1,2,1) #the figure has 1 row, 2 columns, and this plot is the first plot.
plt.title('Seasonwise Points of Dhoni')
plt.xlabel('Seasons')
plt.ylabel('Points')
plt.plot(x,y)

#Plot2:
x=np.array(Seasons)
y=np.array(Points[Pdict['Sachin']])
plt.figure(figsize=(14,2))
plt.subplot(1,2,2) #the figure has 1 row, 2 columns, and this plot is the second plot.
plt.title('Seasonwise Points of Sachin')
plt.xlabel('Seasons')
plt.ylabel('Points')
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x79e2fac69e70>]
```




```
#Plot 1:
x=np.array(Seasons)
y=np.array(Games[Pdict['Sachin']])
plt.subplot(2,3,1) #the figure has 2 row, 3 columns, and this plot is the 1st plot.
plt.title('Sachin')
plt.plot(x,y)

#Plot 2:
x=np.array(Seasons)
y=np.array(Games[Pdict['Rahul']])
plt.subplot(2,3,2) #the figure has 2 row, 3 columns, and this plot is the 2nd plot.
plt.title('Rahul')
plt.plot(x,y)

#Plot 3:
x=np.array(Seasons)
y=np.array(Games[Pdict['Smith']])
plt.subplot(2,3,3) #the figure has 2 row, 3 columns, and this plot is the 3rd plot.
plt.title('Smith')
plt.plot(x,y)

#Plot 4:
```

```
#Plot 4:
x=np.array(Seasons)
y=np.array(Games[Pdict['Sami']])
plt.subplot(2,3,4) #the figure has 2 row, 3 columns, and this plot is the 4th plot.
plt.title('Sami')
plt.plot(x,y)

#Plot 5:
x=np.array(Seasons)
y=np.array(Games[Pdict['Pollard']])
plt.subplot(2,3,5) #the figure has 2 row, 3 columns, and this plot is the 5th plot.
plt.title('Pollard')
plt.plot(x,y)

#Plot 6:
x=np.array(Seasons)
y=np.array(Games[Pdict['Morris']])
plt.subplot(2,3,6) #the figure has 2 row, 3 columns, and this plot is the 6th plot.
plt.title('Morris')
plt.plot(x,y)

plt.suptitle('Games of Different Players')
plt.figure(figsize=(4,2))
plt.show()
```

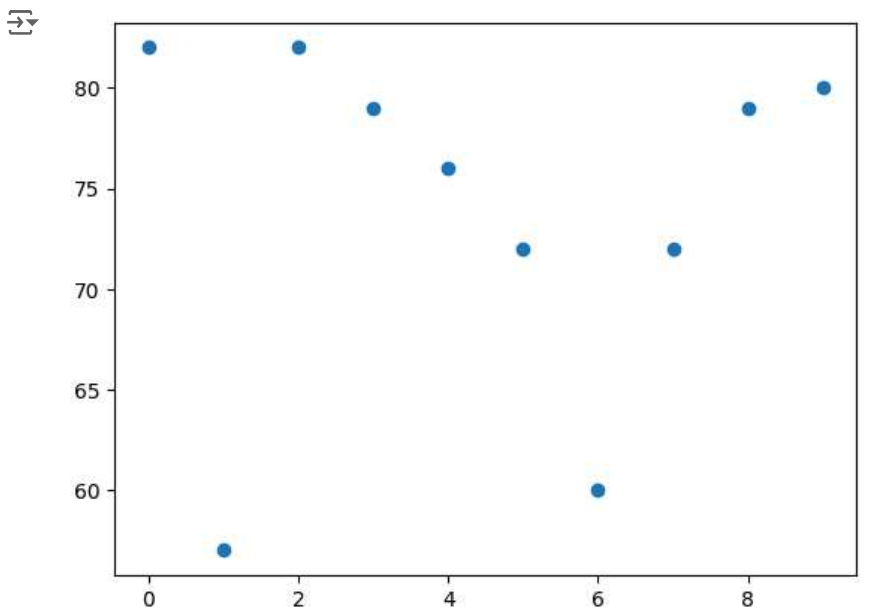


<Figure size 400x200 with 0 Axes>

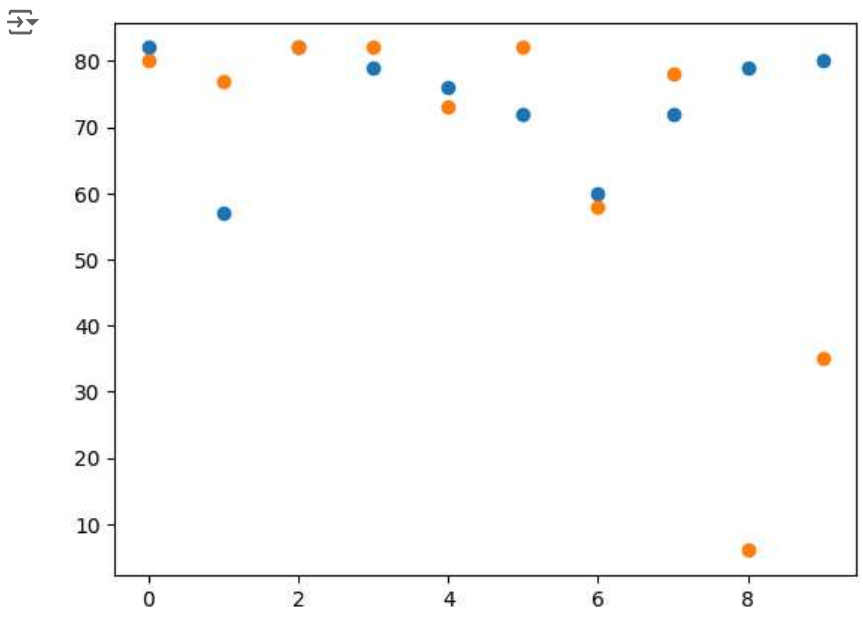
7. Scatter Plot

plots one dot for each observation. It needs two arrays of the same length, one for the values of the x-axis, and one for values on the y-axis.

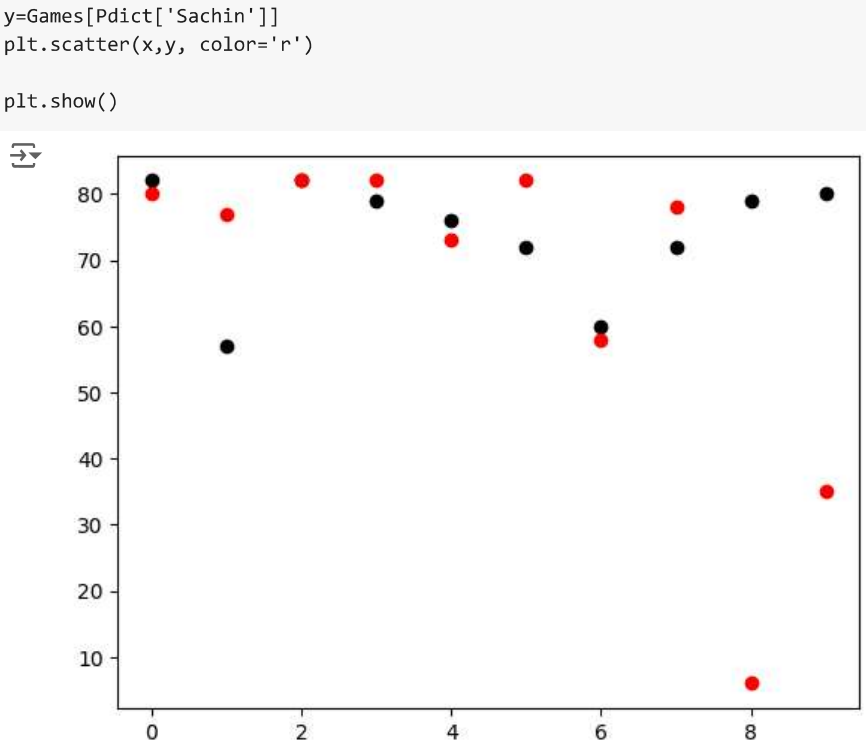
```
# Simple Scatter plot

x=np.array(range(0,10))
y=Games[Pdict['Rahul']]
plt.scatter(x,y)

plt.show()
```



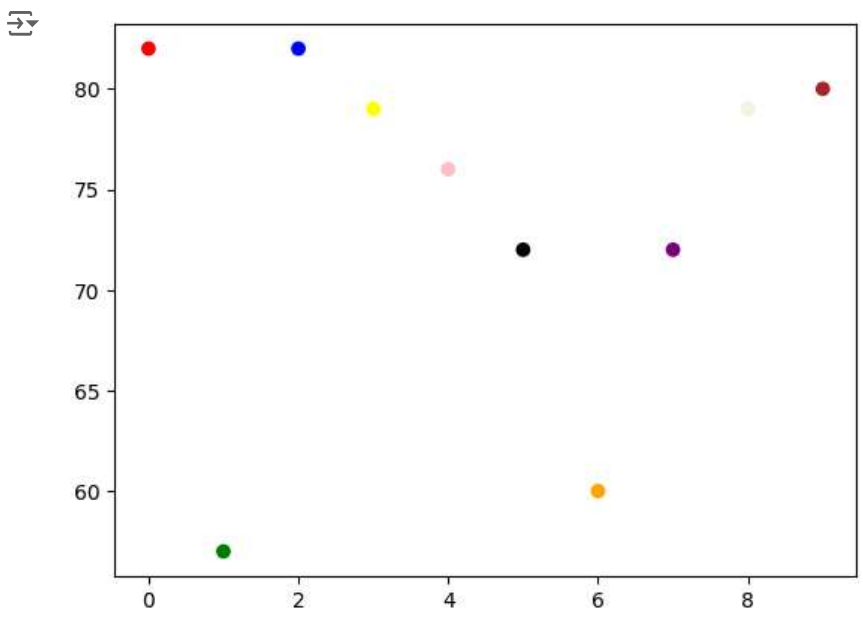
```
# Compare Plots

x=np.array(range(0,10))
y=Games[Pdict['Rahul']]
plt.scatter(x,y)

x=np.array(range(0,10))
y=Games[Pdict['Sachin']]
plt.scatter(x,y)

plt.show()
```



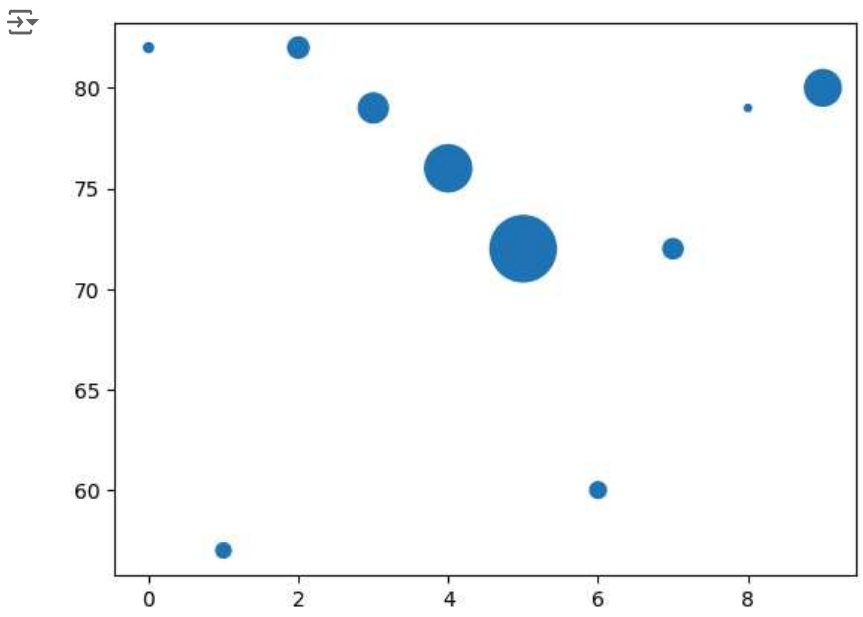

```
x=np.array(range(0,10))
y=Games[Pdict['Rahul']]
plt.scatter(x,y,color='k')

x=np.array(range(0,10))
y=Games[Pdict['Sachin']]
plt.scatter(x,y, color='r')

plt.show()
```



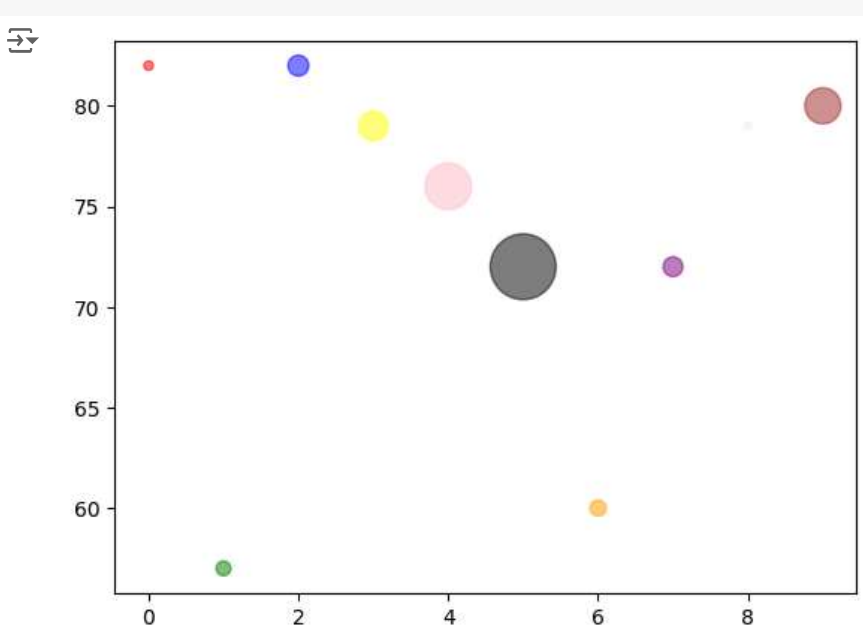

```
# Color each dot Scatter plot

x=np.array(range(0,10))
y=Games[Pdict['Rahul']]

colors = np.array(["red","green","blue","yellow","pink","black","orange","purple","beige","brown"])
plt.scatter(x,y, c=colors)
plt.show()
```




```
# Size each dot Scatter plot

x=np.array(range(0,10))
y=Games[Pdict['Rahul']]

sizes = np.array([20,50,100,200,500,1000,60,90,10,300])
plt.scatter(x,y, s=sizes)
plt.show()
```

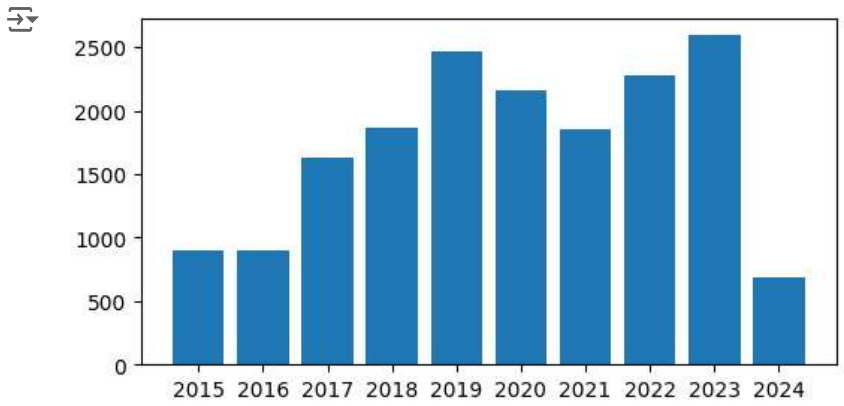



```
# Combine Color,Size and Alpha

x=np.array(range(0,10))
y=Games[Pdict['Rahul']]

sizes = np.array([20,50,100,200,500,1000,60,90,10,300])
colors = np.array(["red","green","blue","yellow","pink","black","orange","purple","beige","brown"])
plt.scatter(x,y,c=colors,s=sizes,alpha=0.5) #Alpha is for transparence
plt.show()
```

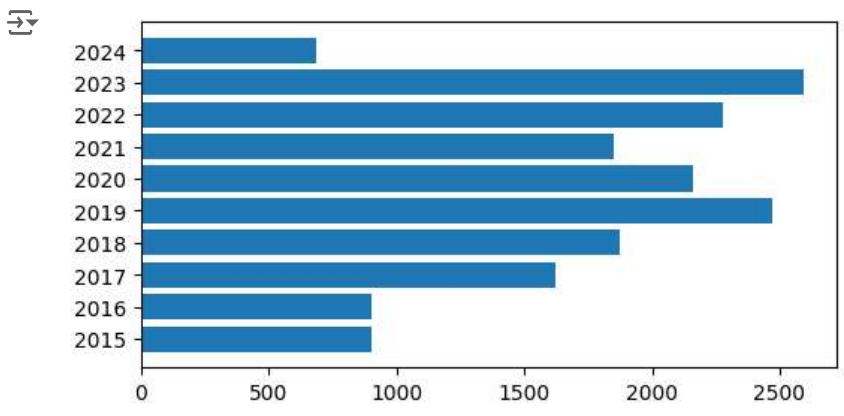


8. Bar Graph

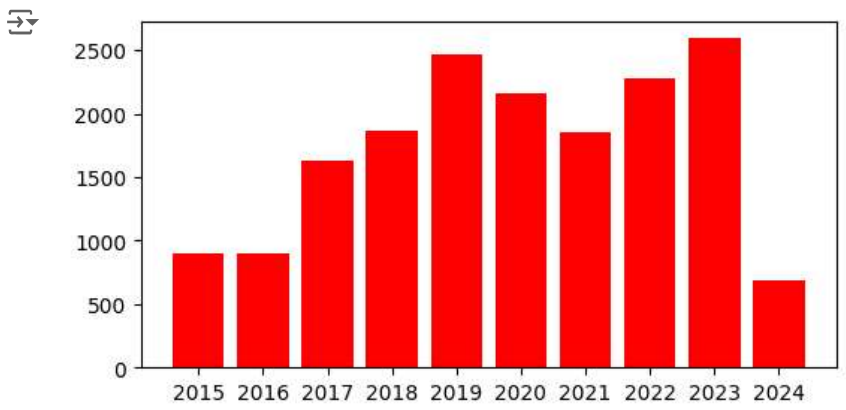
```
# Verticle Bar

x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])
plt.figure(figsize=(6,3))
plt.bar(x,y)
plt.show()
```



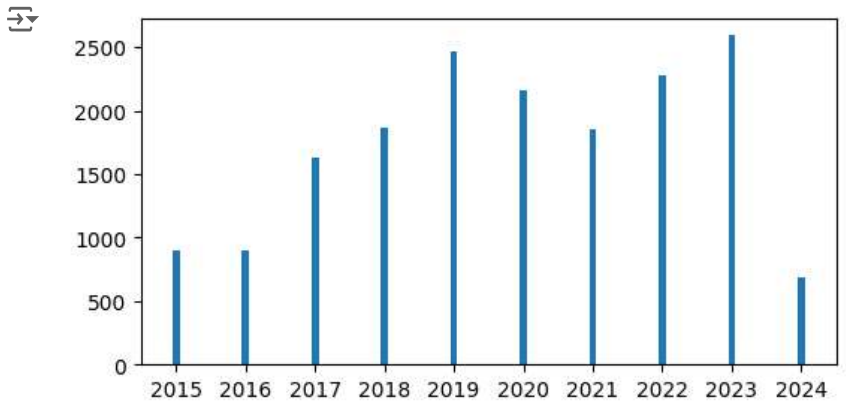

```
# Horizontal Bar

x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])
plt.figure(figsize=(6,3))
plt.barh(x,y)
plt.show()
```



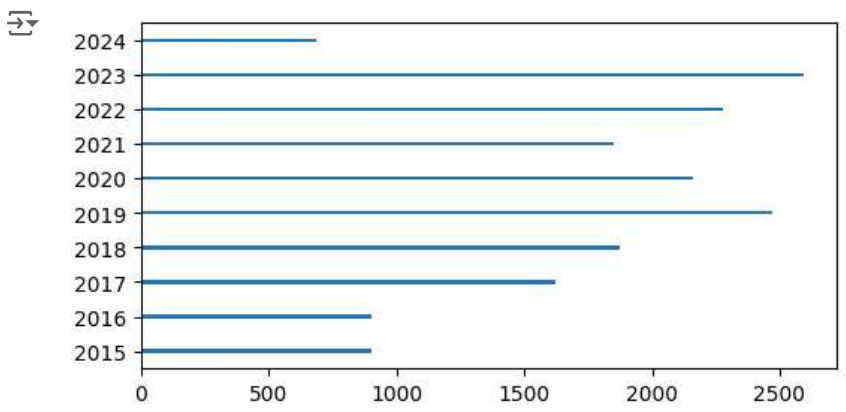
```
# Bar Color

x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])
plt.figure(figsize=(6,3))
plt.bar(x,y, color='r')
plt.show()
```




```
# Bar Width

x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])
plt.figure(figsize=(6,3))
plt.bar(x,y, width=0.1)
plt.show()
```



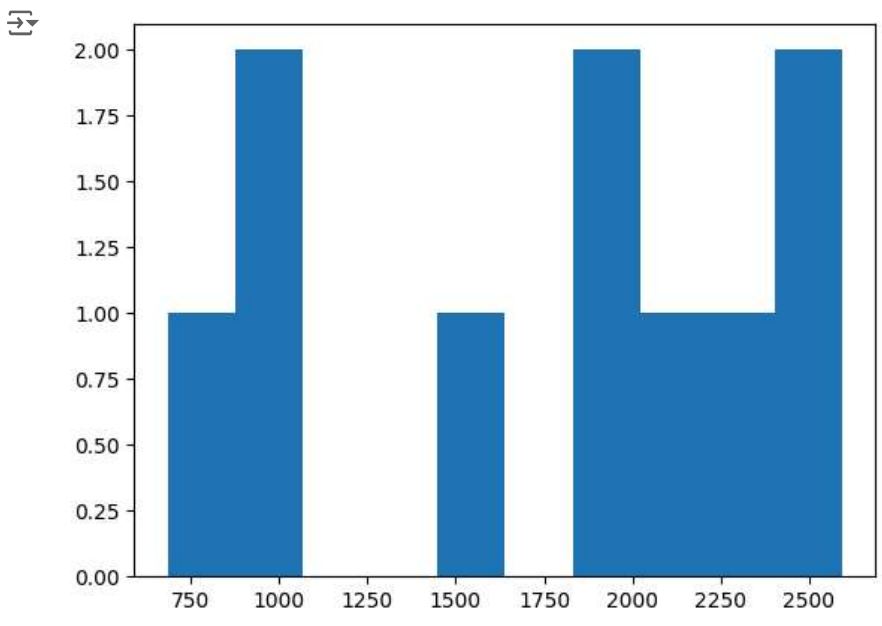

```
# Bar Height

x=np.array(Seasons)
y=np.array(Points[Pdict['Dhoni']])
plt.figure(figsize=(6,3))
plt.barh(x,y, height=0.1)
plt.show()
```



9. Histogram

Shows the frequency distribution. graph showing the number of observations within each given interval.

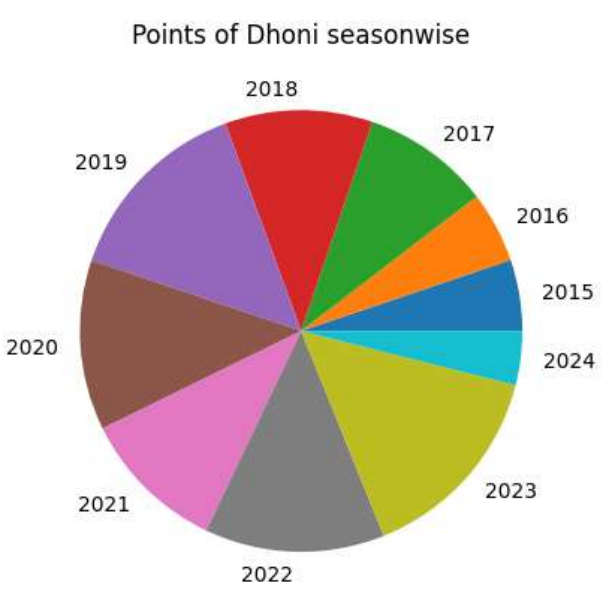
```
x=np.array(Points[Pdict['Dhoni']])

plt.hist(x)
plt.show()
```



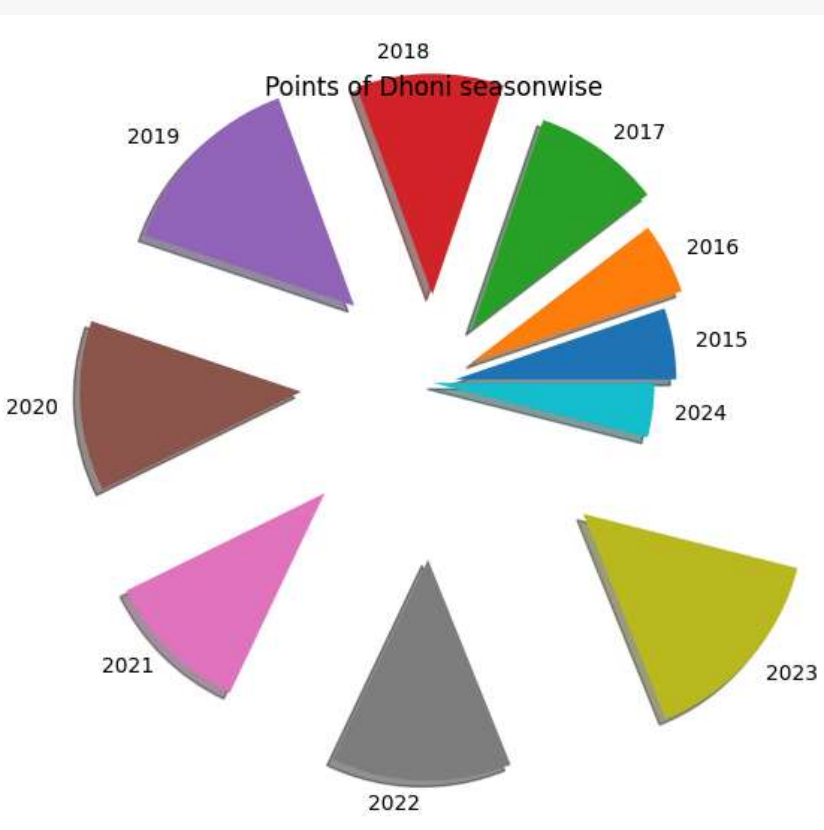
10. Pie Charts

```
x=np.array(Points[Pdict['Dhoni']])

plt.pie(x)

plt.show()
```

```
# Start angle and Labels

x=np.array(Points[Pdict['Dhoni']])
y=np.array(Seasons)

plt.pie(x, labels=y,startangle=360)
plt.title('Points of Dhoni seasonwise')
plt.show()
```



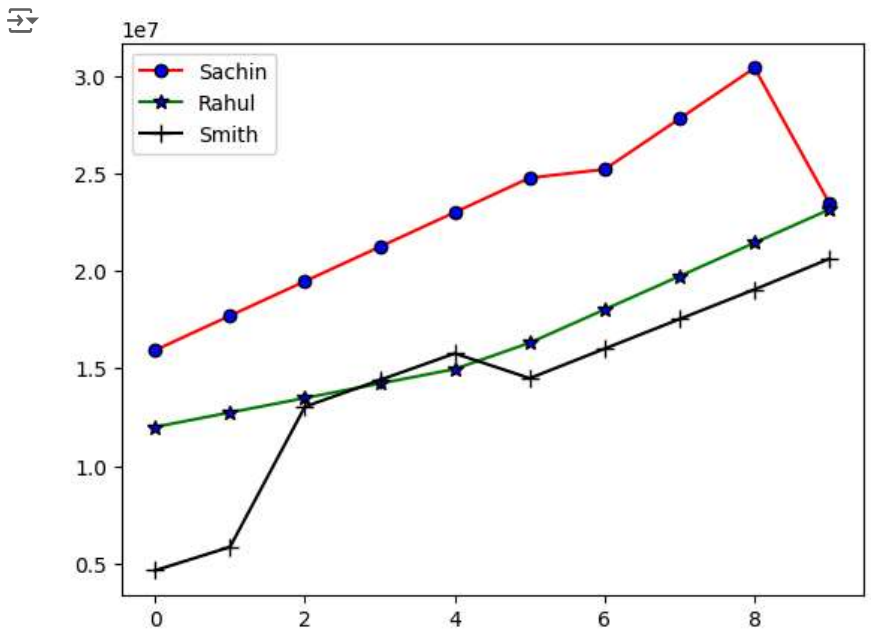

```
# Explode and Shadow

x=np.array(Points[Pdict['Dhoni']])
y=np.array(Seasons)
myexplode = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,0]
plt.pie(x, labels=y,startangle=360,explode=myexplode, shadow=True)
plt.title('Points of Dhoni seasonwise')
plt.show()
```



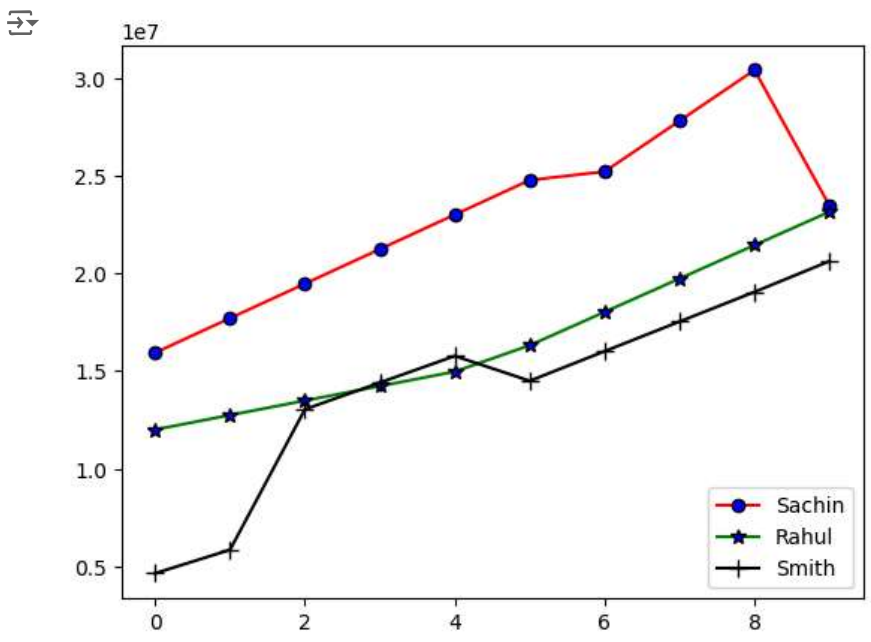
11. legend

It is an area describing the elements of the graph

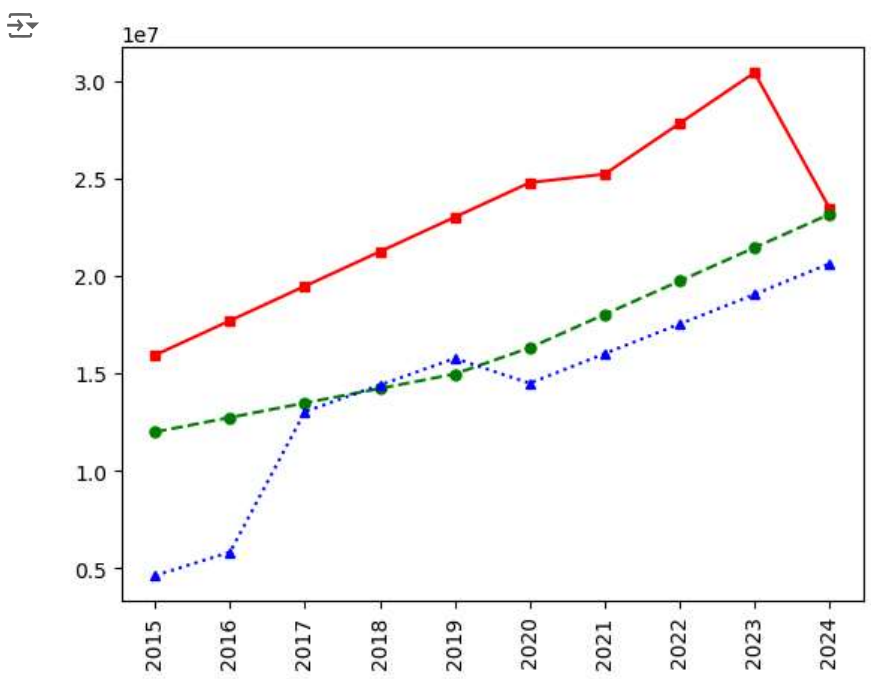
```
plt.plot(Salary[Pdict['Sachin']],color='r',marker='o',ms=6,mec='k',mfc='b',label=Players[Pdict['Sachin']])
plt.plot(Salary[Pdict['Rahul']],color='g',marker='*',ms=7,mec='k',mfc='b',label=Players[Pdict['Rahul']])
plt.plot(Salary[Pdict['Smith']],color='k',marker='+',ms=8,mec='k',mfc='b',label=Players[Pdict['Smith']])

plt.legend(loc='upper left')
plt.show()
```




```
plt.plot(Salary[Pdict['Sachin']],color='r',marker='o',ms=6,mec='k',mfc='b',label=Players[Pdict['Sachin']])
plt.plot(Salary[Pdict['Rahul']],color='g',marker='*',ms=7,mec='k',mfc='b',label=Players[Pdict['Rahul']])
plt.plot(Salary[Pdict['Smith']],color='k',marker='+',ms=8,mec='k',mfc='b',label=Players[Pdict['Smith']])
plt.legend(loc='lower right')
plt.show()
```

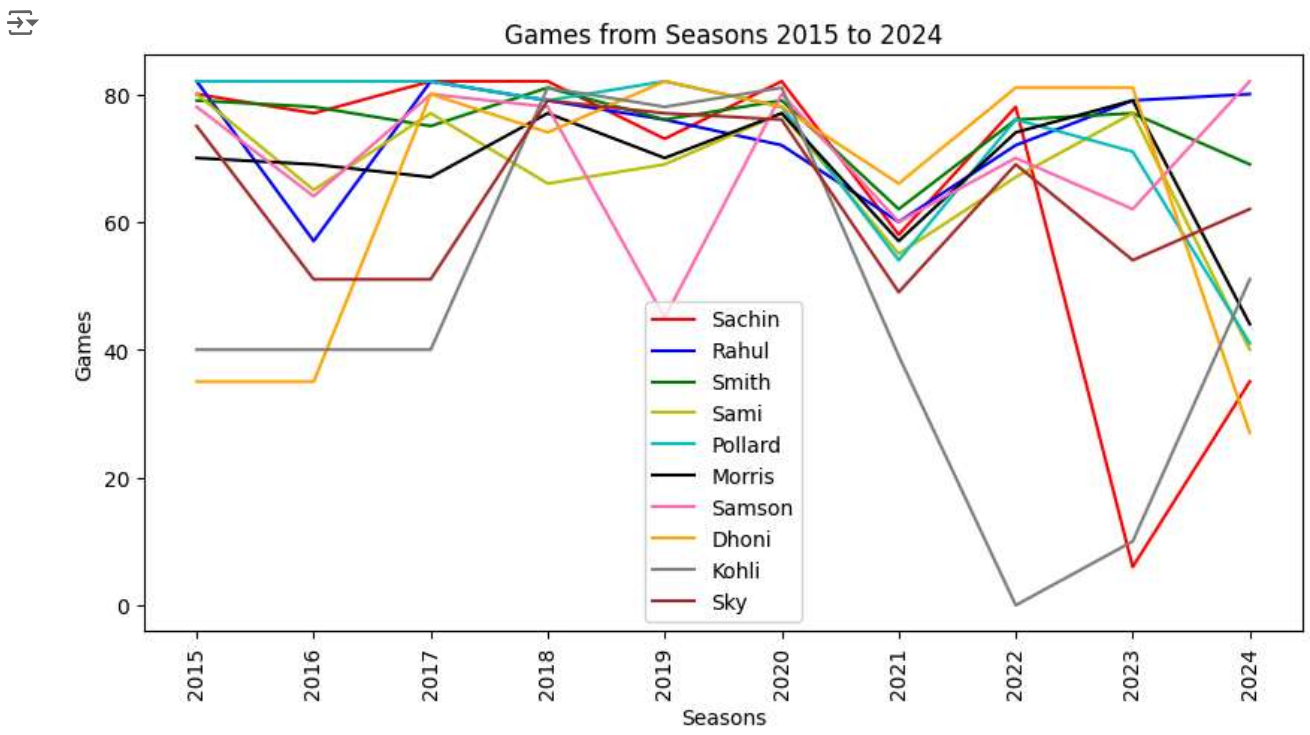



```
plt.plot(Salary[Pdict['Sachin']],color='red',ls='-',marker='s',ms='5',label=Players[Pdict['Sachin']])
plt.plot(Salary[Pdict['Rahul']],color='green',ls='--',marker='o',ms='5',label=Players[Pdict['Rahul']])
plt.plot(Salary[Pdict['Smith']],color='blue',ls=':',marker='^',ms='5',label=Players[Pdict['Smith']])

plt.xticks(list(range(0,10)),Seasons,rotation='vertical')
plt.show()
```

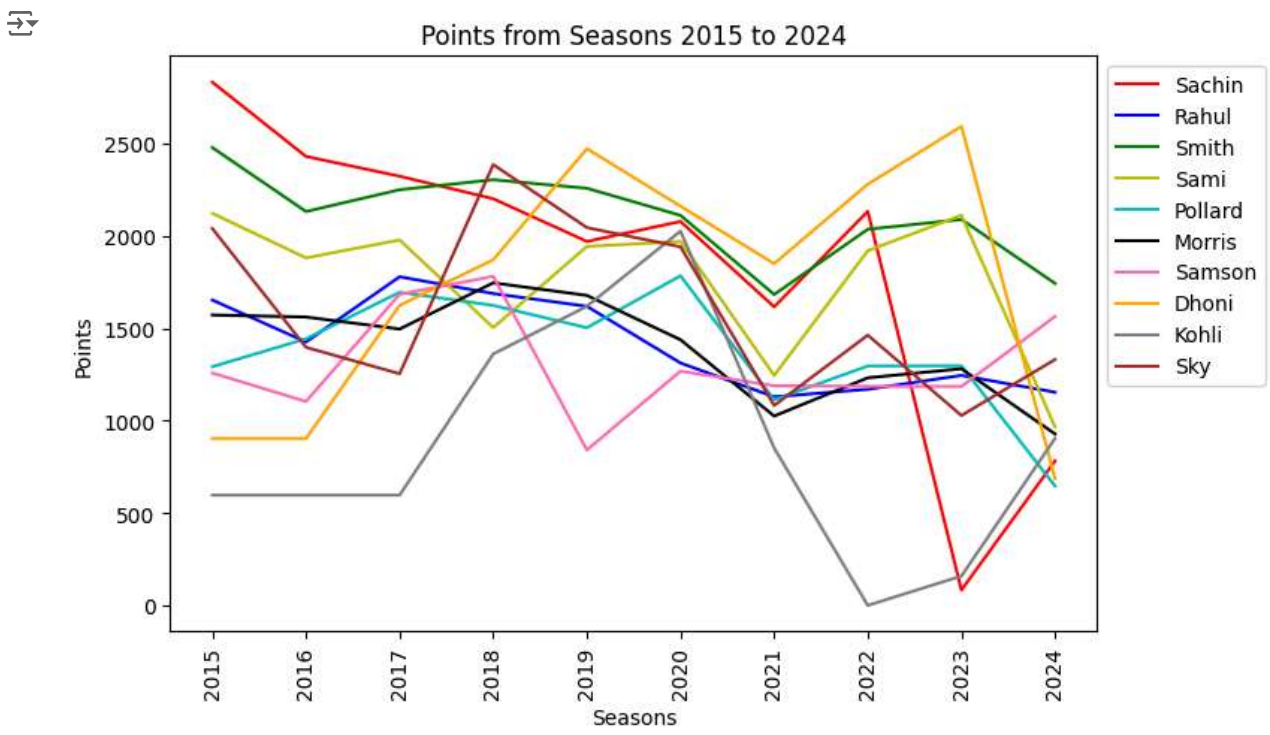



```
plt.figure(figsize=(10,5))
plt.plot(Games[Pdict['Sachin']],color='r',label=Players[Pdict['Sachin']])
plt.plot(Games[Pdict['Rahul']],color='b',label=Players[Pdict['Rahul']])
plt.plot(Games[Pdict['Smith']],color='g',label=Players[Pdict['Smith']])
plt.plot(Games[Pdict['Sami']],color='y',label=Players[Pdict['Sami']])
plt.plot(Games[Pdict['Pollard']],color='c',label=Players[Pdict['Pollard']])
plt.plot(Games[Pdict['Morris']],color='k',label=Players[Pdict['Morris']])
plt.plot(Games[Pdict['Samson']],color='hotpink',label=Players[Pdict['Samson']])
plt.plot(Games[Pdict['Dhoni']],color='orange',label=Players[Pdict['Dhoni']])
plt.plot(Games[Pdict['Kohli']],color='gray',label=Players[Pdict['Kohli']])
plt.plot(Games[Pdict['Sky']],color='brown',label=Players[Pdict['Sky']])


plt.xticks(list(range(0,10)),Seasons,rotation='vertical')
plt.legend()
```

```
plt.title("Games from Seasons 2015 to 2024")
plt.xlabel("Seasons")
plt.ylabel("Games")


plt.show()
```

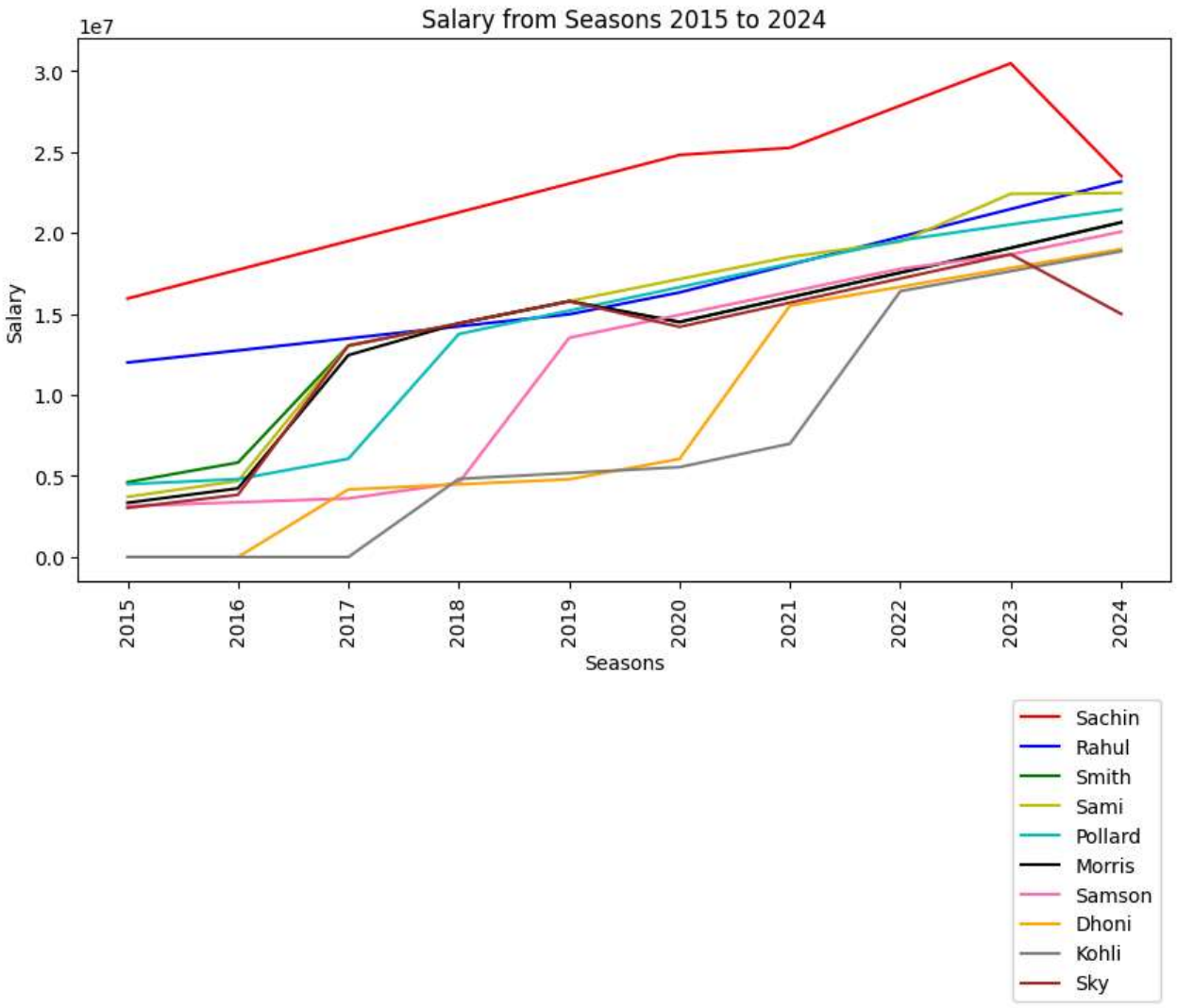



```
plt.figure(figsize=(10,5))
plt.plot(Points[Pdict['Sachin']],color='r',label=Players[Pdict['Sachin']])
plt.plot(Points[Pdict['Rahul']],color='b',label=Players[Pdict['Rahul']])
plt.plot(Points[Pdict['Smith']],color='g',label=Players[Pdict['Smith']])
plt.plot(Points[Pdict['Sami']],color='y',label=Players[Pdict['Sami']])
plt.plot(Points[Pdict['Pollard']],color='c',label=Players[Pdict['Pollard']])
plt.plot(Points[Pdict['Morris']],color='k',label=Players[Pdict['Morris']])
plt.plot(Points[Pdict['Samson']],color='hotpink',label=Players[Pdict['Samson']])
plt.plot(Points[Pdict['Dhoni']],color='orange',label=Players[Pdict['Dhoni']])
plt.plot(Points[Pdict['Kohli']],color='gray',label=Players[Pdict['Kohli']])
plt.plot(Points[Pdict['Sky']],color='brown',label=Players[Pdict['Sky']])


plt.xticks(list(range(0,10)),Seasons,rotation='vertical')
plt.legend(bbox_to_anchor=(1.0, 1.0))
plt.title("Points from Seasons 2015 to 2024")
plt.xlabel("Seasons")
plt.ylabel("Points")

plt.show()
```




```
plt.figure(figsize=(10,5))
plt.plot(Salary[Pdict['Sachin']],color='r',label=Players[Pdict['Sachin']])
plt.plot(Salary[Pdict['Rahul']],color='b',label=Players[Pdict['Rahul']])
plt.plot(Salary[Pdict['Smith']],color='g',label=Players[Pdict['Smith']])
plt.plot(Salary[Pdict['Sami']],color='y',label=Players[Pdict['Sami']])
plt.plot(Salary[Pdict['Pollard']],color='c',label=Players[Pdict['Pollard']])
plt.plot(Salary[Pdict['Morris']],color='k',label=Players[Pdict['Morris']])
plt.plot(Salary[Pdict['Samson']],color='hotpink',label=Players[Pdict['Samson']])
plt.plot(Salary[Pdict['Dhoni']],color='orange',label=Players[Pdict['Dhoni']])
plt.plot(Salary[Pdict['Kohli']],color='gray',label=Players[Pdict['Kohli']])
plt.plot(Salary[Pdict['Sky']],color='brown',label=Players[Pdict['Sky']])


plt.xticks(list(range(0,10)),Seasons,rotation='vertical')
plt.legend(bbox_to_anchor=(1.0, -0.2))
plt.title("Salary from Seasons 2015 to 2024")
plt.xlabel("Seasons")
plt.ylabel("Salary")

plt.show()
```

[↓]



Seaborn

Its a powerful and user-friendly Python library built on top of Matplotlib. It's specifically designed for statistical data visualization, making it an excellent tool for data scientists to explore and visualize datasets effectively. Seaborn provides a high-level interface for drawing attractive and informative statistical graphics.

1.Importing Required Libraries

```
pip install seaborn
```

```
import seaborn as sns
import pandas as pd
```

```
Games_new = {
    "Sachin": [80, 77, 82, 82, 73, 82, 58, 78, 6, 35],
    "Rahul": [82, 57, 82, 79, 76, 72, 60, 72, 79, 80],
    "Smith": [79, 78, 75, 81, 76, 79, 62, 76, 77, 69],
    "Sami": [80, 65, 77, 66, 69, 77, 55, 67, 77, 40],
    "Pollard": [82, 82, 82, 79, 82, 78, 54, 76, 71, 41],
    "Morris": [70, 69, 67, 77, 70, 77, 57, 74, 79, 44],
    "Samson": [78, 64, 80, 78, 45, 80, 60, 70, 62, 82],
    "Dhoni": [35, 35, 80, 74, 82, 78, 66, 81, 81, 27],
    "Kohli": [40, 40, 40, 81, 78, 81, 39, 0, 10, 51],
    "Sky": [75, 51, 51, 79, 77, 76, 49, 69, 54, 62]
}
```

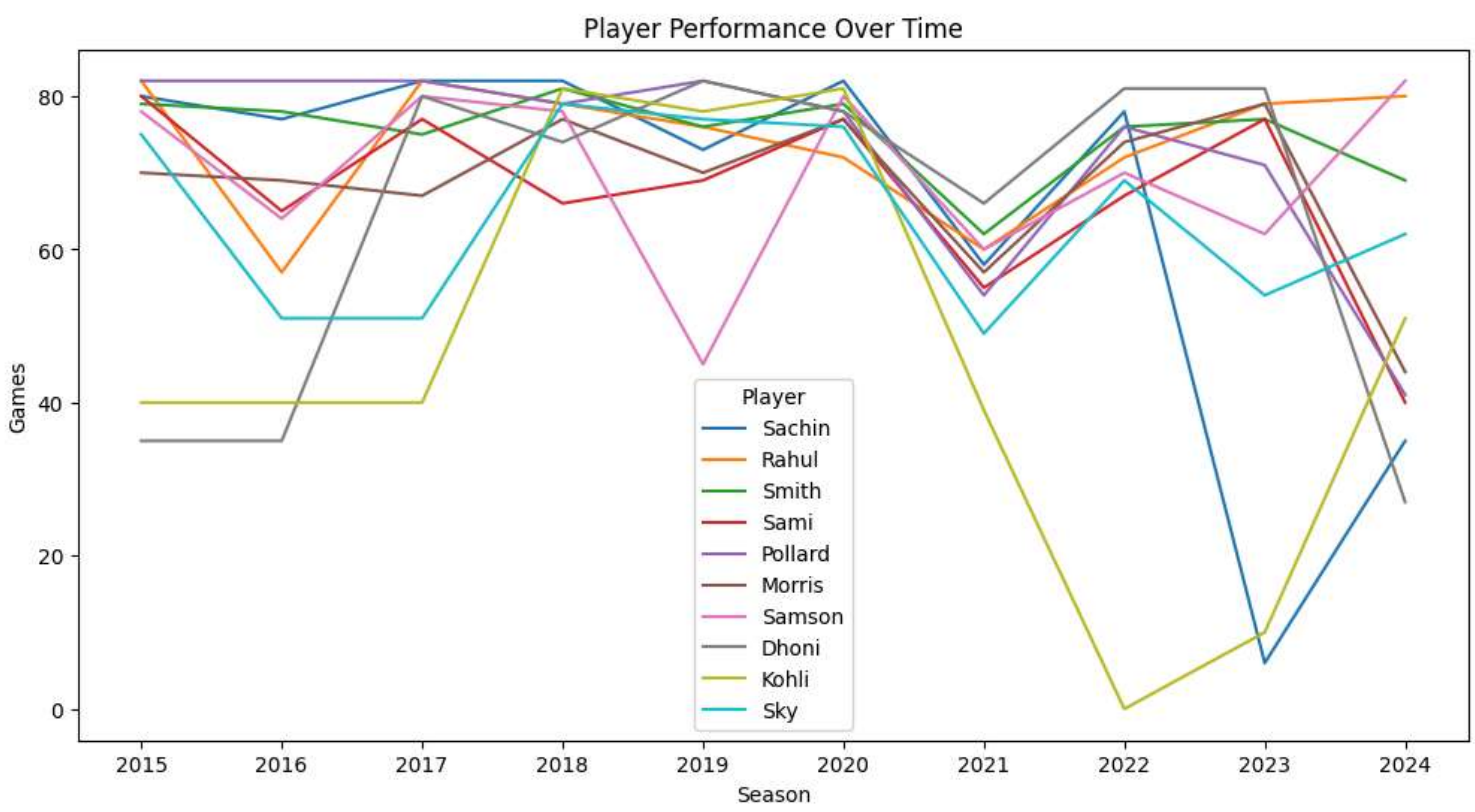
```
df=pd.DataFrame(Games_new, index=Seasons)

# Resetting index to include 'Seasons' as a column
df.reset_index(inplace=True)

# Resetting index to use 'Seasons' as a column
df = df.melt(id_vars=['index'], var_name='Player', value_name='Games')

# Renaming the columns appropriately
df = df.rename(columns={'index': 'Season'})


# Display the DataFrame
print(df.head(15))
```

```
   Season  Player  Games
0    2015  Sachin     80
1    2016  Sachin     77
2    2017  Sachin     82
3    2018  Sachin     82
4    2019  Sachin     73
5    2020  Sachin     82
6    2021  Sachin     58
7    2022  Sachin     78
8    2023  Sachin      6
9    2024  Sachin     35
10   2015   Rahul     82
11   2016   Rahul     57
12   2017   Rahul     82
13   2018   Rahul     79
14   2019   Rahul     76
```

2. Line plot (Games over Seasons of each player)

This plot will help to find the trend over time for each player.

```
plt.figure(figsize=(12,6))
sns.lineplot(data=df, x='Season', y='Games', hue='Player')
plt.title('Player Performance Over Time')
plt.xlabel('Season')
plt.ylabel('Games')
plt.show()
```

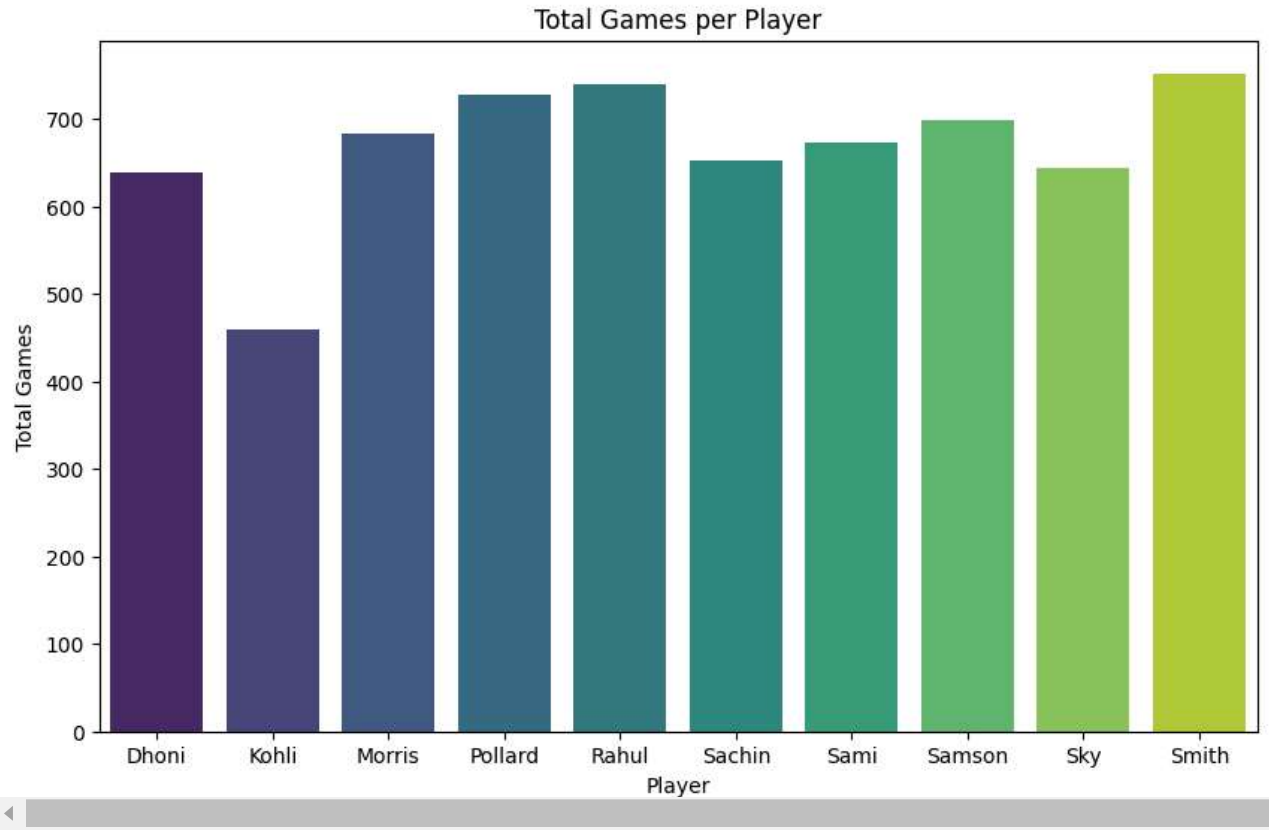


3. Bar Plot (Total Games per Player Across All Seasons)

This plot helps to compare the total games of each player

```
plt.figure(figsize=(10, 6))
total_games=df.groupby('Player')['Games'].sum().reset_index()
sns.barplot(data=total_games,x='Player', y='Games',palette='viridis')
plt.title('Total Games per Player')
plt.xlabel('Player')
plt.ylabel('Total Games')
plt.show()
```

```
<ipython-input-62-bb72a30d9512>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=total_games,x='Player', y='Games',palette='viridis')
```

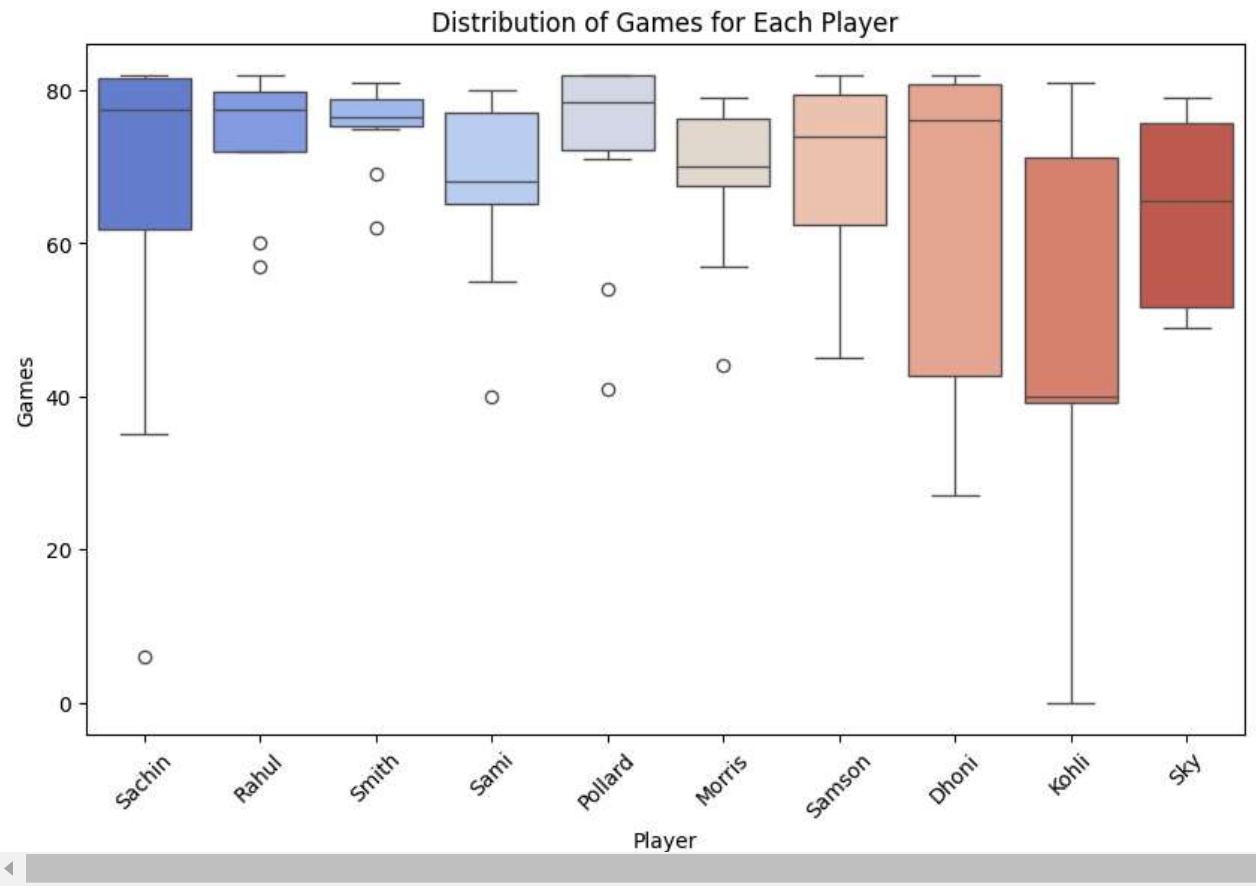


4. Box Plot (Distribution of Games for Each Player)

This plot shows the distribution of games, highlighting the median, quartiles, and outliers.

```
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Player', y='Games', palette='coolwarm')
plt.title('Distribution of Games for Each Player')
plt.xlabel('Player')
plt.ylabel('Games')
plt.xticks(rotation=45)
plt.show()
#
```

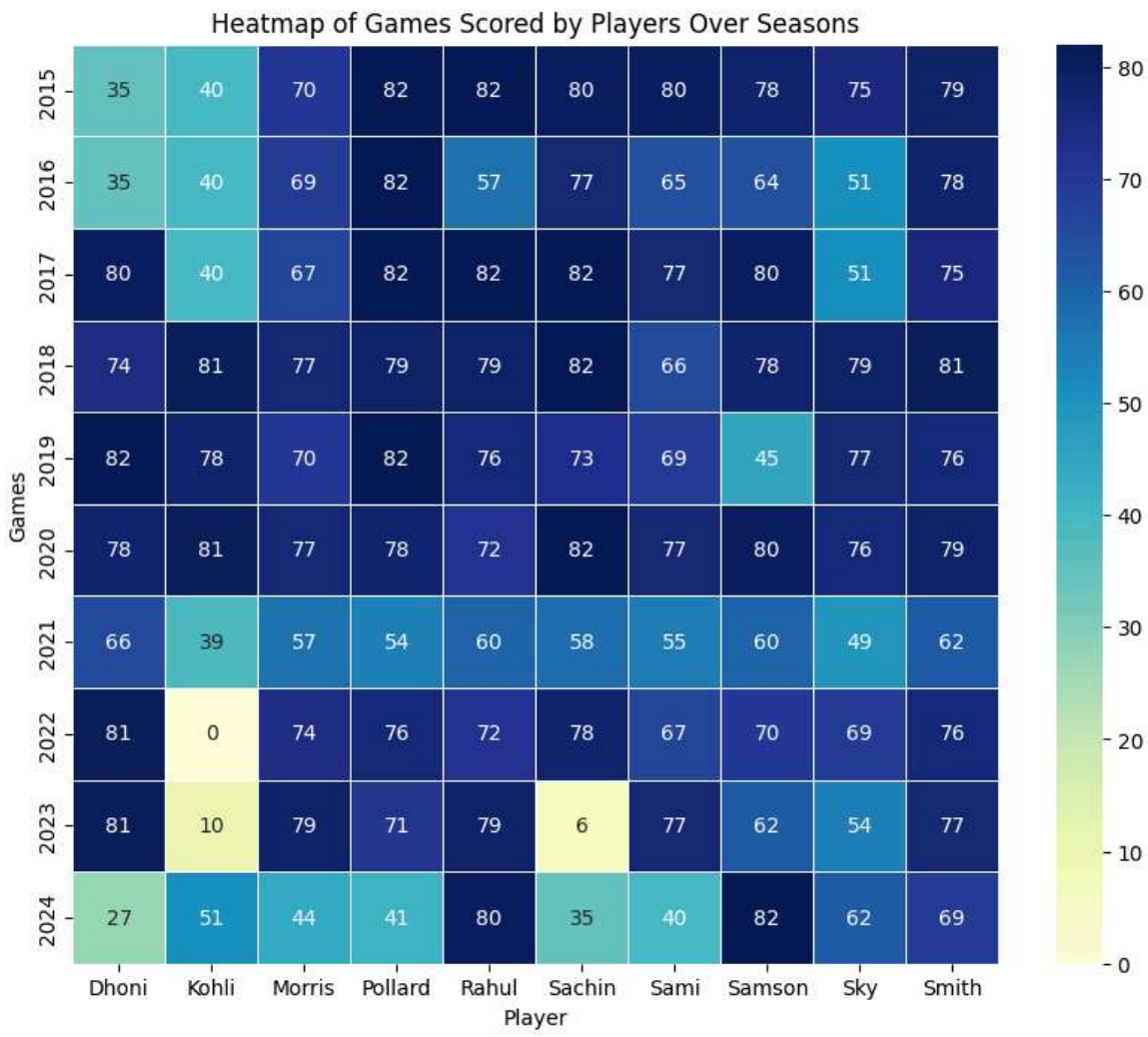
```
<ipython-input-63-49353617f655>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect

  sns.boxplot(data=df, x='Player', y='Games', palette='coolwarm')
```



5. Heatmap (Games played by Players Over Seasons)

This plot gives a clear overview of ho many games played in which seasons.

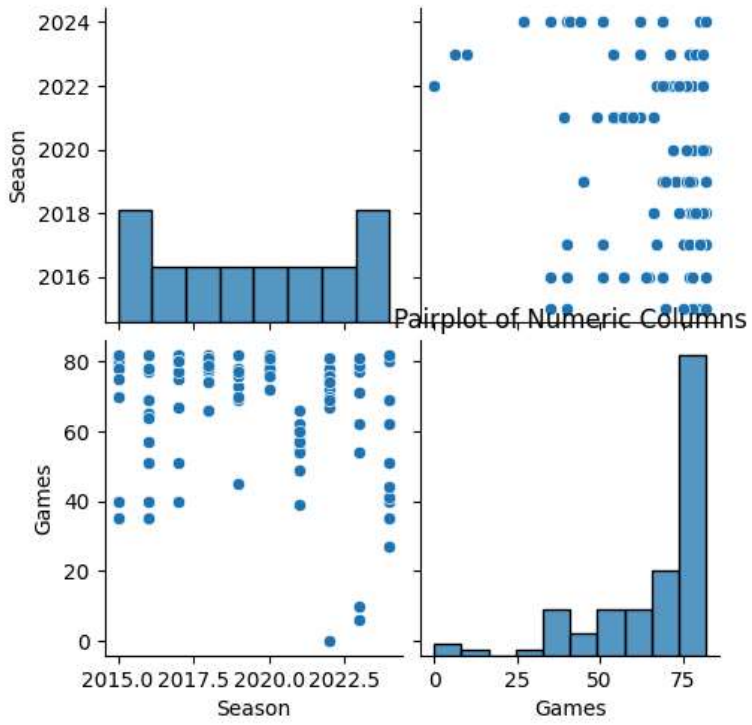
```
pivot_table = df.pivot(index='Season', columns='Player', values='Games')
plt.figure(figsize=(10, 8))
sns.heatmap(pivot_table, annot=True, cmap='YlGnBu', linewidths=0.5)
plt.title('Heatmap of Games Scored by Players Over Seasons')
plt.xlabel('Player')
plt.ylabel('Games')
plt.show()
```



6. Pairplot (Exploring Relationships)

```
plt.figure(figsize=(12,10))
# Step 1: Extract relevant numeric data (only 'Games' in this case)
numeric_df = df[['Season', 'Games']].copy()

# Step 2: Convert 'Season' to numeric (if needed)
numeric_df['Season'] = pd.to_numeric(numeric_df['Season'])

# Step 3: Apply pairplot
sns.pairplot(numeric_df)
plt.title('Pairplot of Numeric Columns')
plt.show()
```

```
<Figure size 1200x1000 with 0 Axes>
```



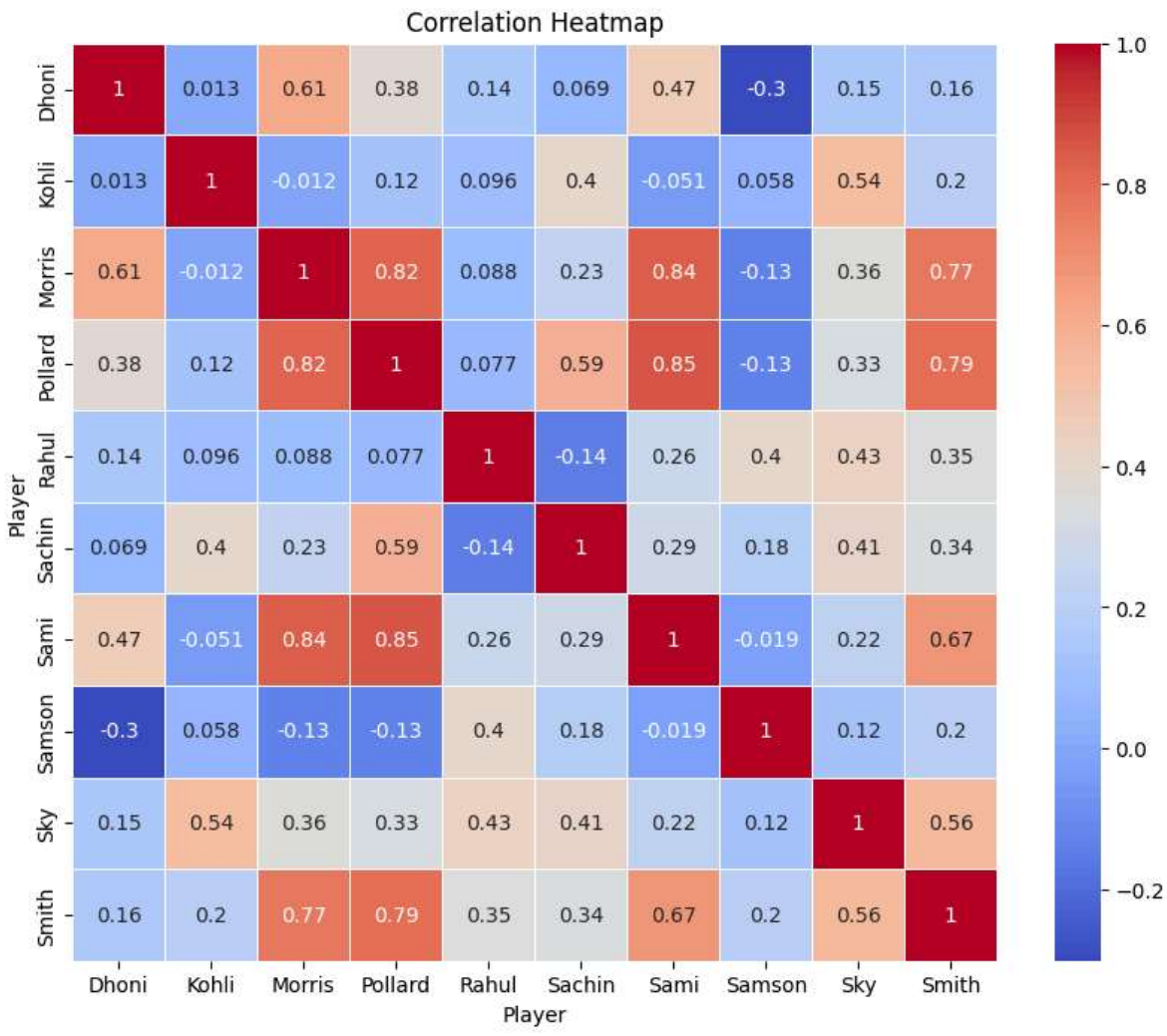
7. Statistical Analysis: Correlation

To check for correlations between players' performances:

```
correlation = df.pivot(index='Season', columns='Player', values='Games')
correlation_matrix = correlation.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show
```

```
matplotlib.pyplot.show
def show(*args, **kwargs)
```

```
/usr/local/lib/python3.10/dist-packages/matplotlib/pyplot.py
Display all open figures.

Parameters
----------
block : bool, optional
```



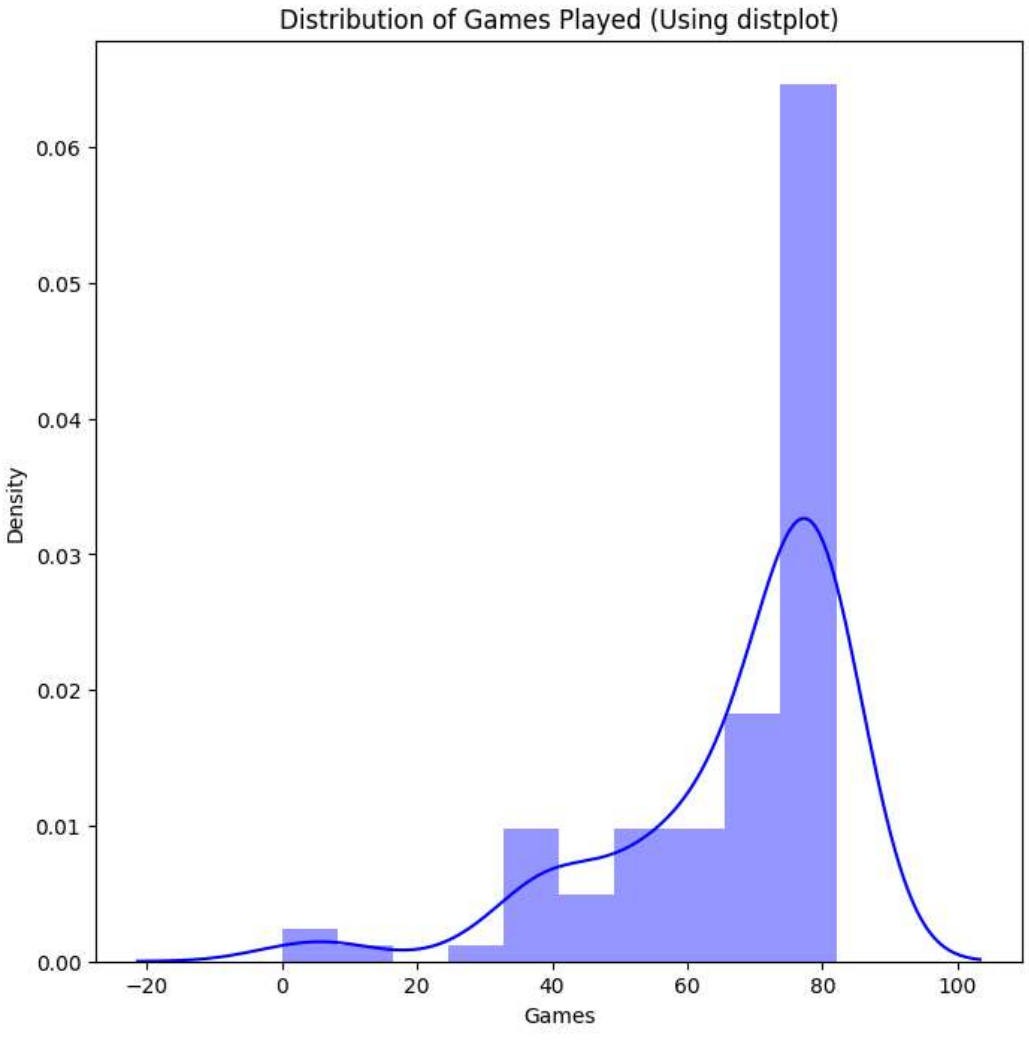
8. distplot() & displot()

both visualize the distribution of a dataset.

- distplot():
  - This was a commonly used function in earlier versions of Seaborn to plot histograms along with a KDE (Kernel Density Estimate) curve.
  - As of Seaborn v0.11.0, distplot() is deprecated and is no longer recommended for use.
  - It combines a histogram and a KDE plot in one figure.
- displot():
  - Introduced in Seaborn v0.11.0, displot() is a more powerful and flexible function for visualizing distributions.
  - It can create histograms, KDE plots, ECDF (Empirical CDF) plots, or combinations of these, based on the parameters you pass.
  - It works with Facets for multi-plot grids, allowing you to create complex visualizations easily.

```
# distplot

plt.figure(figsize=(8, 8))
sns.distplot(df['Games'], bins=10, kde=True, color='blue')
plt.title('Distribution of Games Played (Using distplot)')
plt.xlabel('Games')
plt.ylabel('Density')
plt.show()
```

```
<ipython-input-143-bdee107f612c>:4: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Games'], bins=10, kde=True, color='blue')
```




```
# Visualizing distribution of 'Games' using displot
plt.figure(figsize=(8, 8))
sns.displot(df['Games'], bins=10, kde=True, color='green')
plt.title('Distribution of Games Played (Using displot)')
plt.xlabel('Games')
plt.ylabel('Density')
plt.show()
```

```
<Figure size 800x800 with 0 Axes>
```