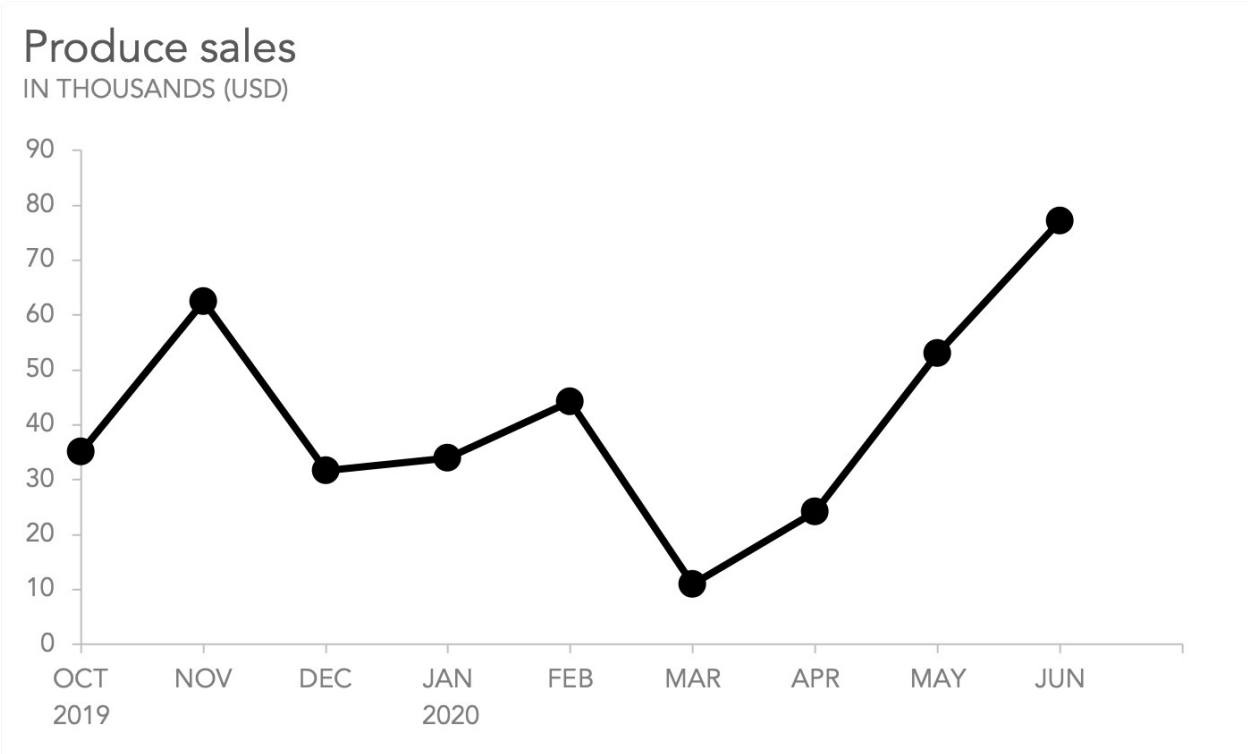
Matplotlib

Matplotlib is one of the most popular Python packages used for data visualization.
Matplotlib was originally written by John D. Hunter in 2003

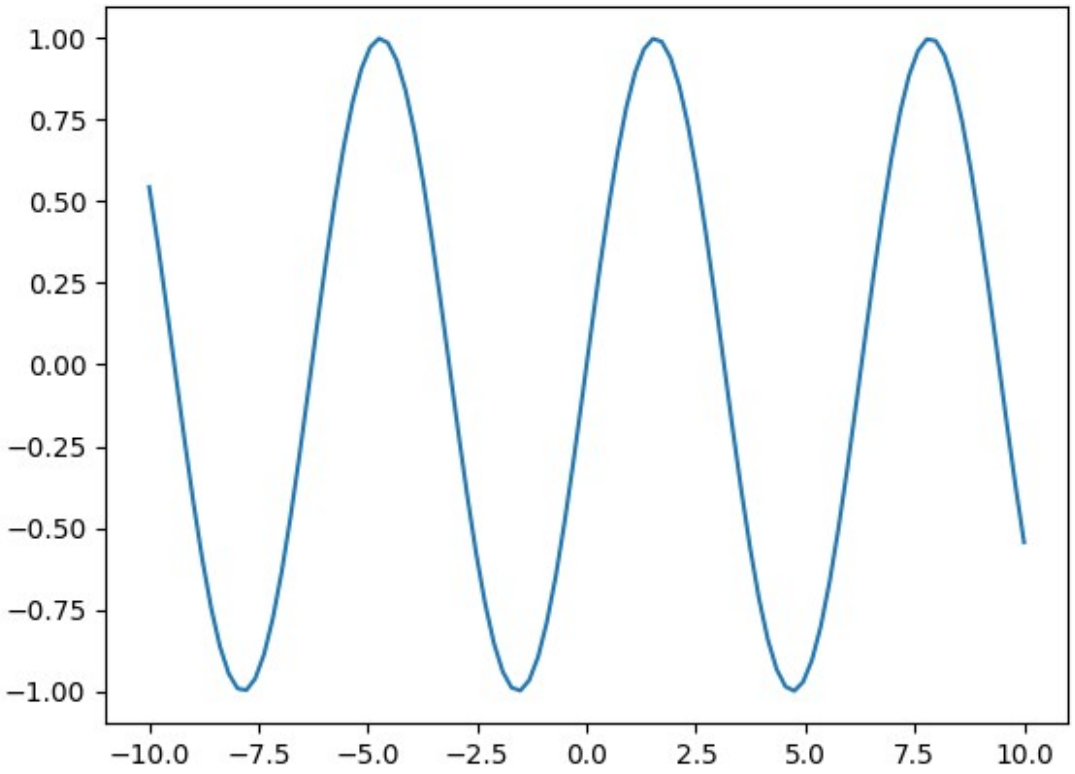
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2D Line plot

- Bivariate Analysis
- categorical - numerical and numerical - numerical
- Use case - Time series data




```
x = np.linspace(-10, 10, 100)
y = np.sin(x)
plt.plot(x, y)
plt.show()
```

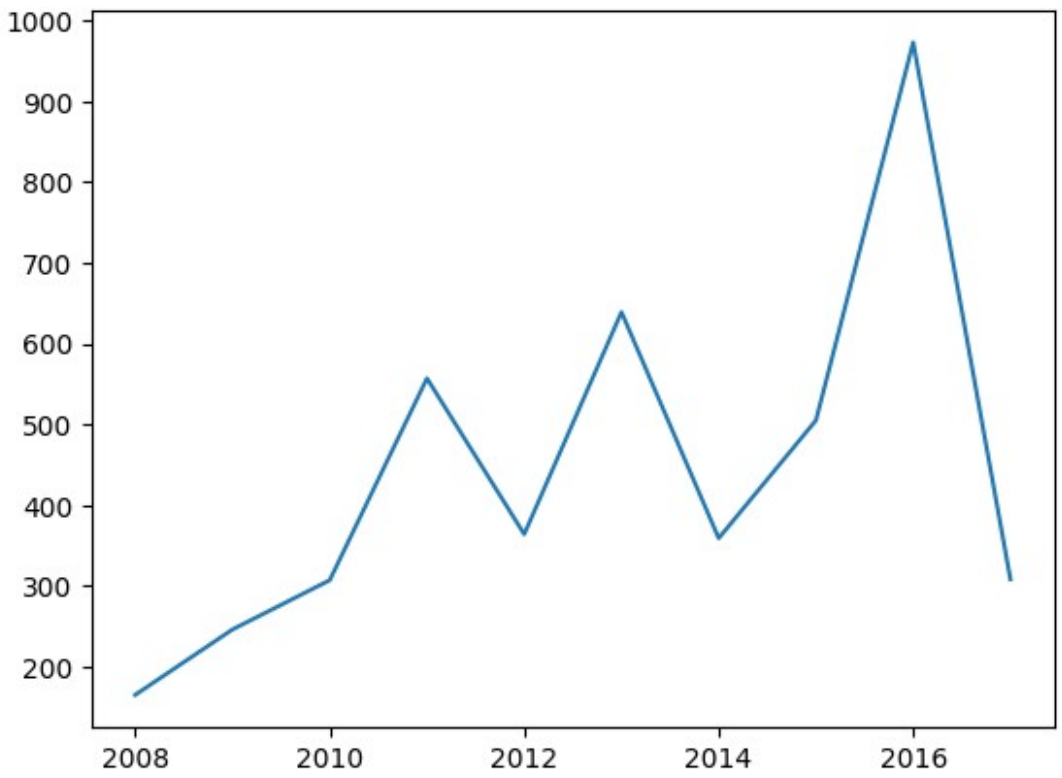



```
batsman = pd.read_csv('Datasets/sharma-kohli.csv')
batsman.head(2)
```

	index	RG Sharma	V Kohli
0	2008	404	165
1	2009	362	246

```
plt.plot(batsman['index'], batsman['V Kohli'])
```

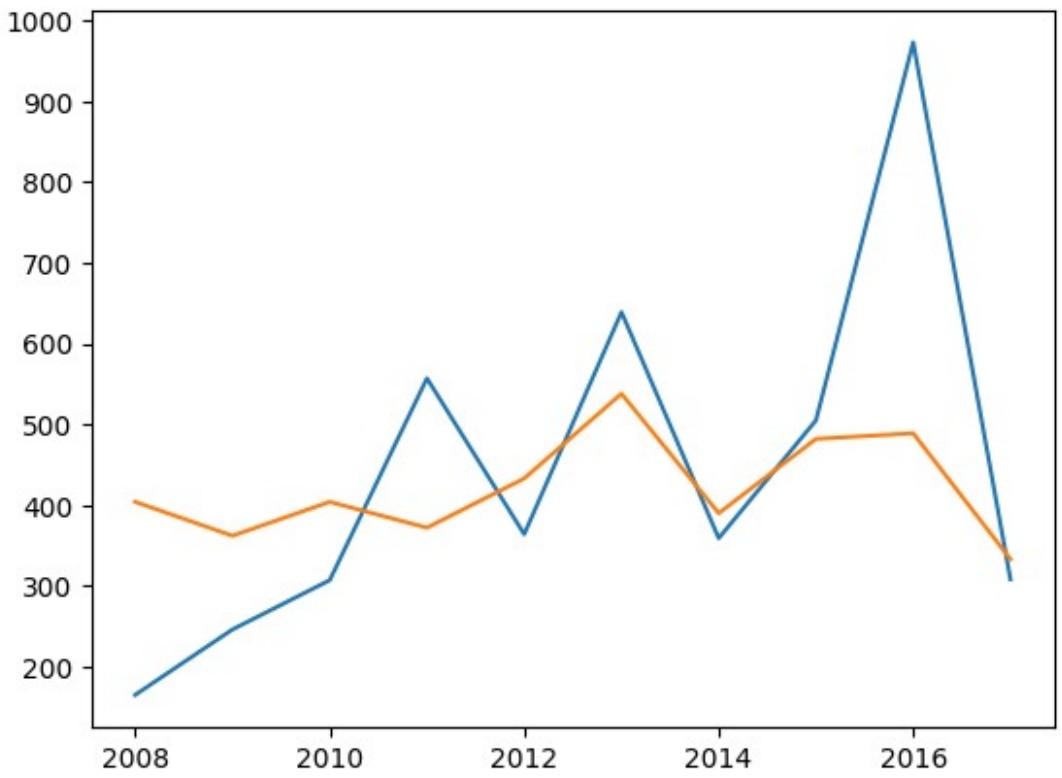
```
[<matplotlib.lines.Line2D at 0x18dbd90d610>]
```



Plotting multiple plots

```
plt.plot(batsman['index'], batsman['V Kohli'])
plt.plot(batsman['index'], batsman['RG Sharma'])
```

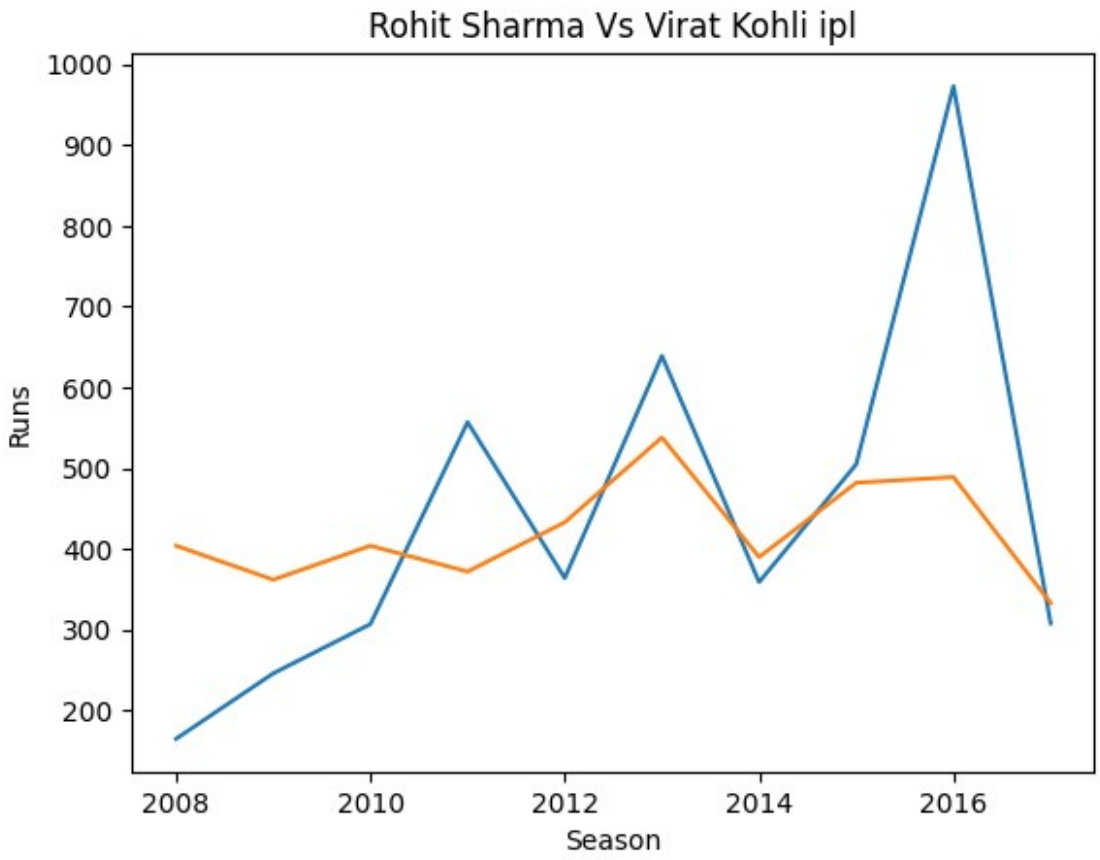
```
[<matplotlib.lines.Line2D at 0x1ab0330cbc0>]
```



Labels and Title

```
plt.plot(batsman['index'], batsman['V Kohli'])
plt.plot(batsman['index'], batsman['RG Sharma'])

plt.title('Rohit Sharma Vs Virat Kohli ipl')
plt.xlabel('Season')
plt.ylabel('Runs')
```

```
Text(0, 0.5, 'Runs')
```

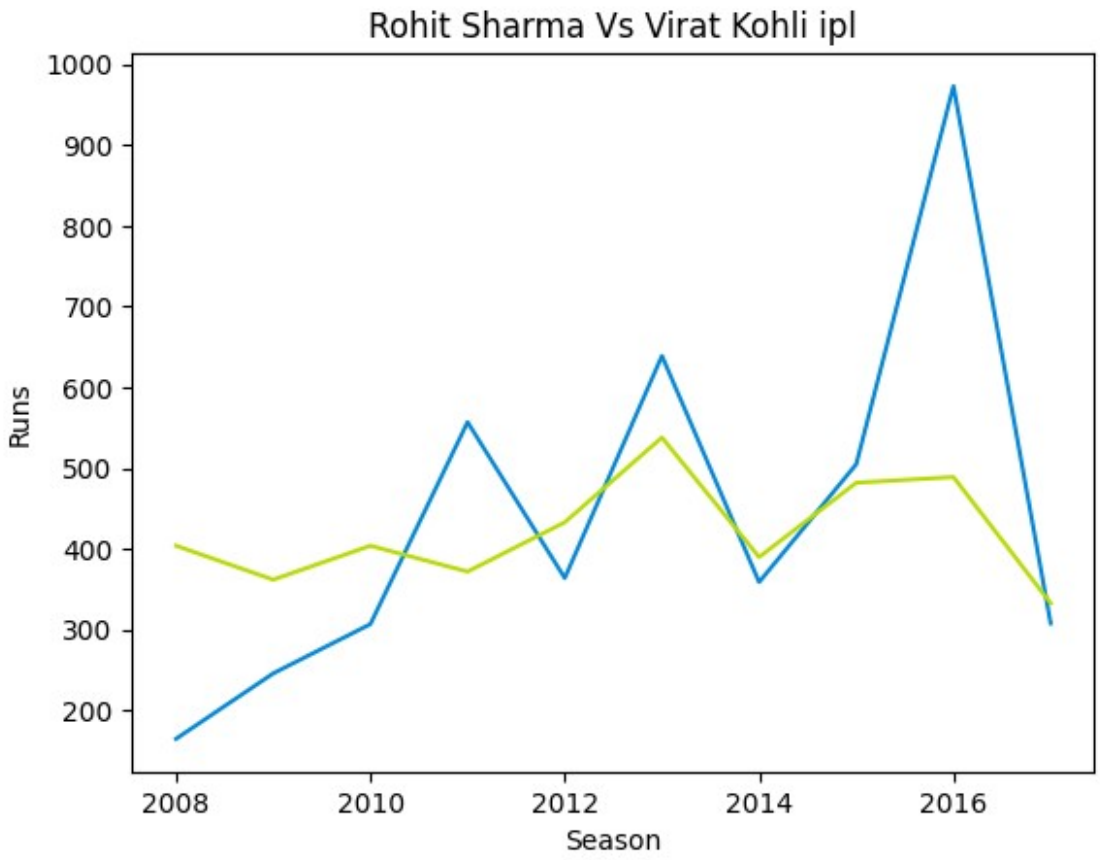


Colors(hex) and line(width and style)

- `linewidth` - specified in pixels.

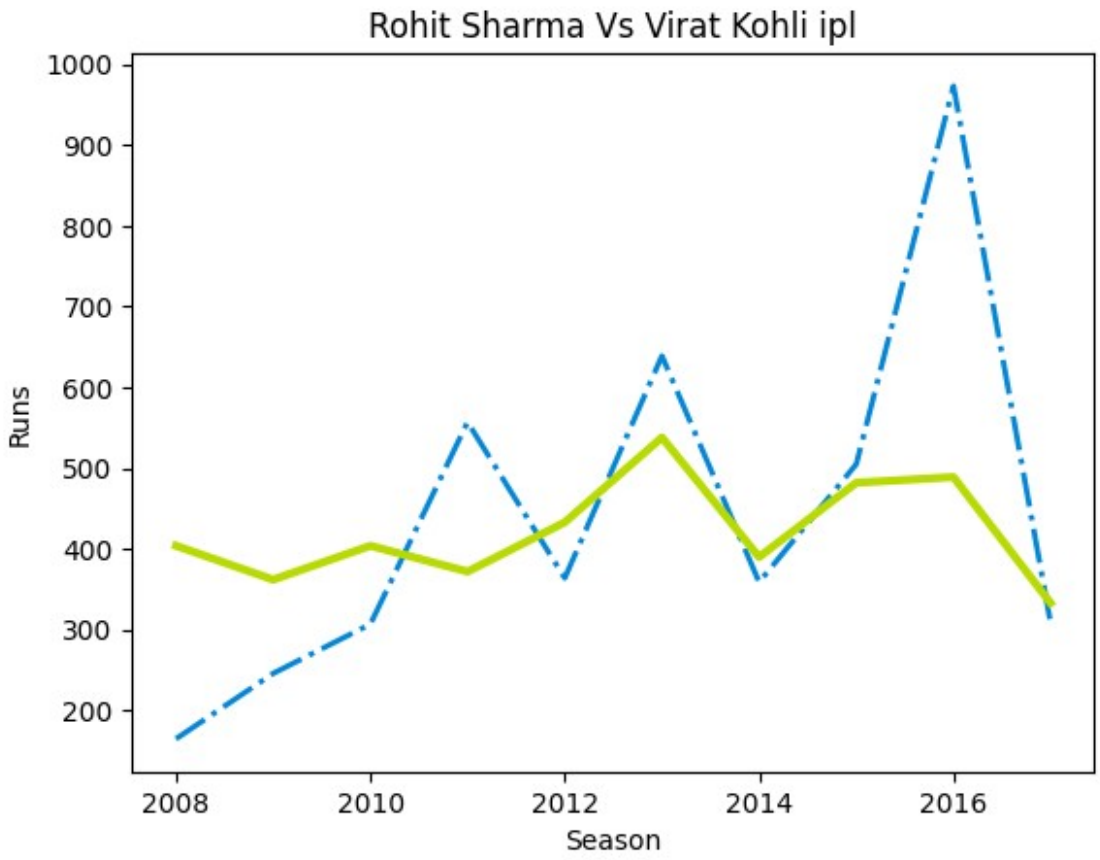
Character	Colour
`b`	blue
`g`	green
`r`	red
`c`	cyan
`m`	magenta
`y`	yellow
`k`	black
`w`	white

```
plt.plot(batsman['index'], batsman['V Kohli'], color='#0087db')
plt.plot(batsman['index'], batsman['RG Sharma'], color='#badb00')

plt.title('Rohit Sharma Vs Virat Kohli ipl')
plt.xlabel('Season')
plt.ylabel('Runs')

Text(0, 0.5, 'Runs')
```




```
plt.plot(batsman['index'], batsman['V Kohli'], color='#0087db',
linestyle='dashdot', linewidth=2)
plt.plot(batsman['index'], batsman['RG Sharma'], color='#badb00',
linestyle='solid', linewidth=3)

plt.title('Rohit Sharma Vs Virat Kohli ipl')
plt.xlabel('Season')
plt.ylabel('Runs')
```

```
Text(0, 0.5, 'Runs')
```



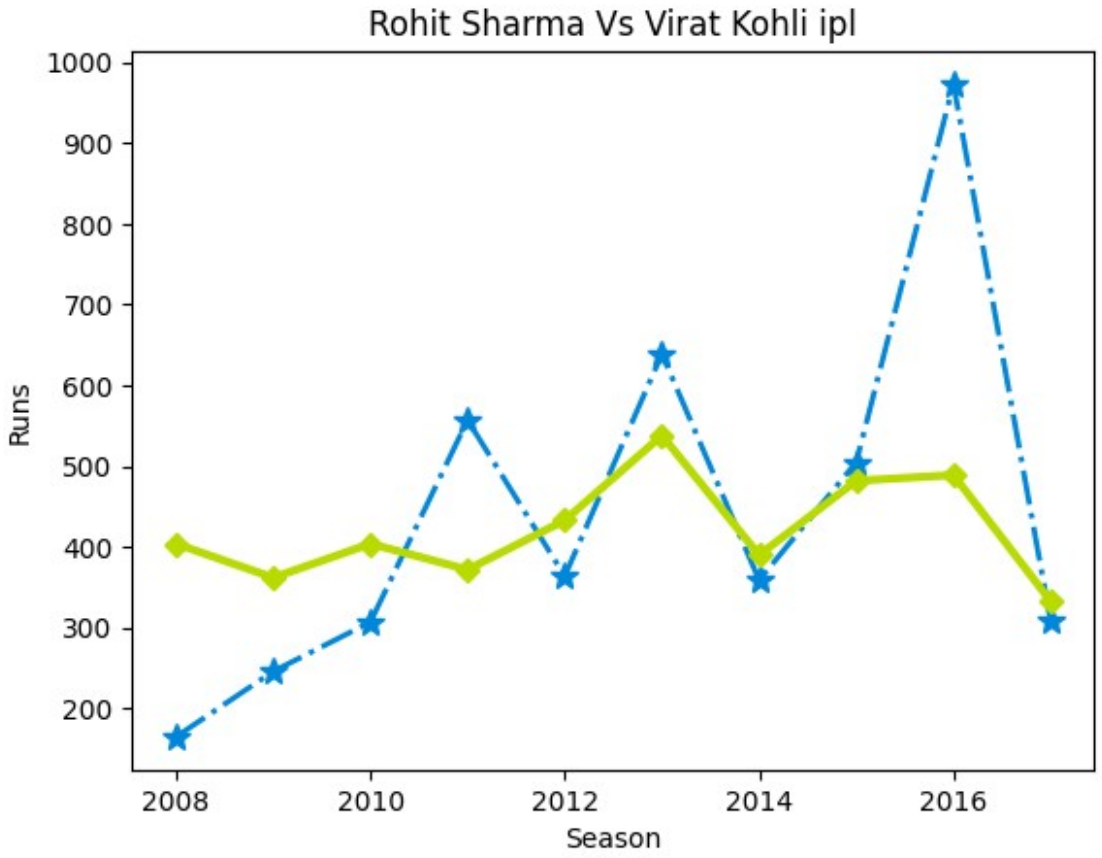
Markers

Marker	Symbol	Description	Marker	Symbol	Description
`.`		Point	`8`		Octagon
`,`		Pixel	`s`		Square
`o`		Circle	`p`		Pentagon
`v`		Triangle Down	`P`		Plus (filled)
`^`		Triangle Up	`*`		Star
`<`		Triangle Left	`h`		Hexagon1
`>`		Triangle Right	`H`		Hexagon2
`1`		Tri Down	`+`		Plus
`2`		Tri Up	`x`		X
`3`		Tri Left	`X`		X (filled)
`4`		Tri Right	`D`		Diamond

```
plt.plot(batsman['index'], batsman['V Kohli'], color='#0087db',
linestyle='dashdot', linewidth=2, marker='*', markersize=10)
plt.plot(batsman['index'], batsman['RG Sharma'], color='#badb00',
linestyle='solid', linewidth=3, marker='D')
```

```
plt.title('Rohit Sharma Vs Virat Kohli ipl')
plt.xlabel('Season')
plt.ylabel('Runs')
```

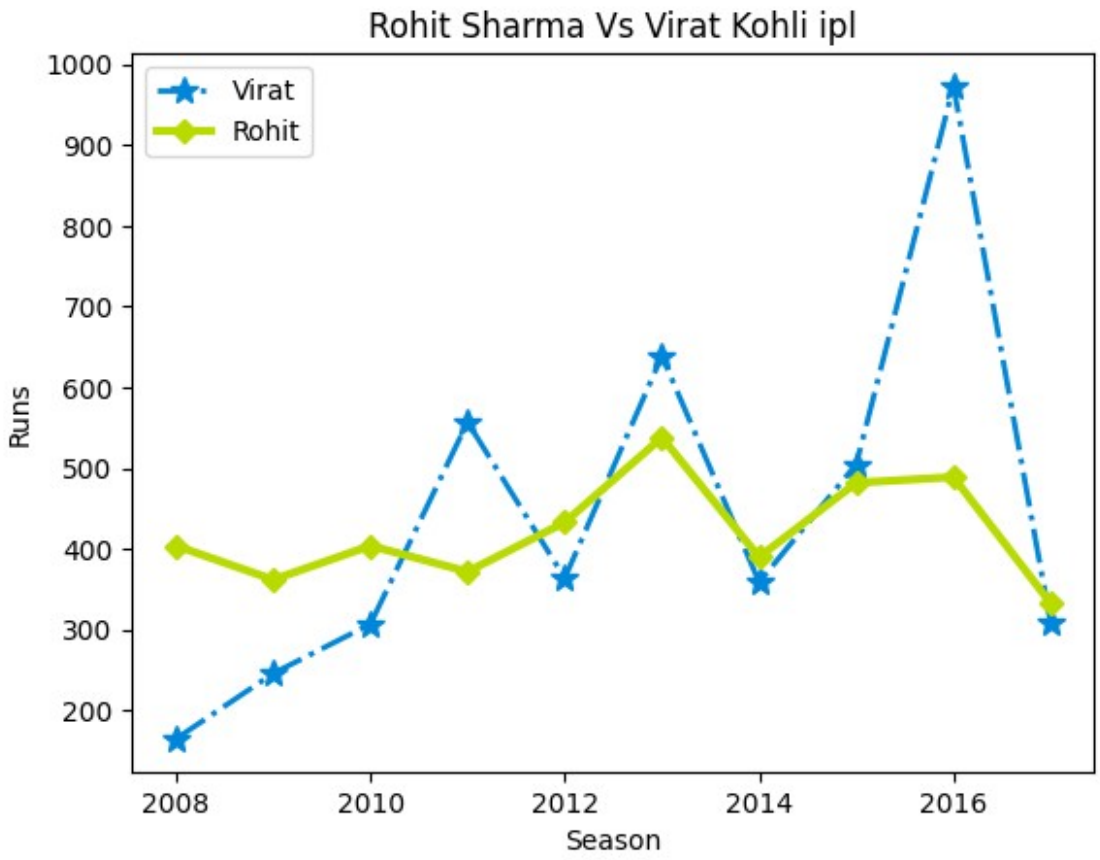
```
Text(0, 0.5, 'Runs')
```



Legend and Location

```
plt.plot(batsman['index'], batsman['V Kohli'], color='#0087db',
linestyle='dashdot', linewidth=2, marker='*', markersize=10,
label='Virat')
plt.plot(batsman['index'], batsman['RG Sharma'], color='#badb00',
linestyle='solid', linewidth=3, marker='D', label='Rohit')

plt.title('Rohit Sharma Vs Virat Kohli ipl')
plt.xlabel('Season')
plt.ylabel('Runs')

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

```
<matplotlib.legend.Legend at 0x1ab03657a10>
```

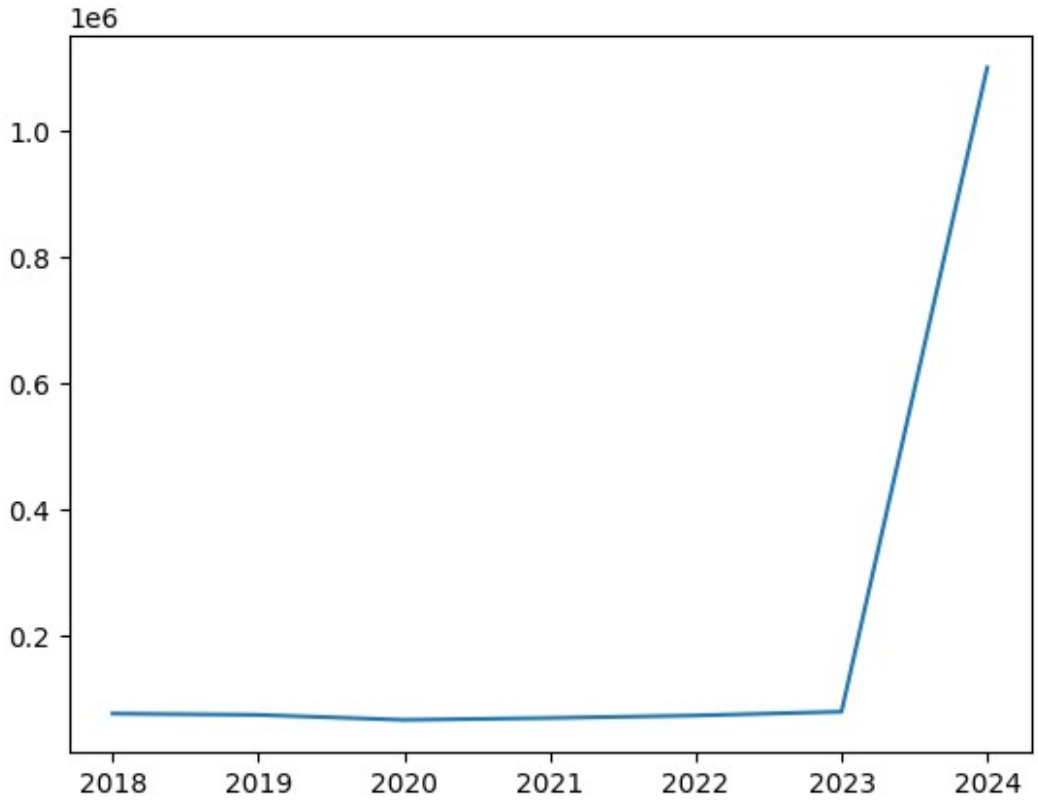


Limiting axes

- Sometime any outlier or unexpected value comes into data which on plotting ruins the appearance of graph and lost the imp info
- In such cases limiting axes is used.

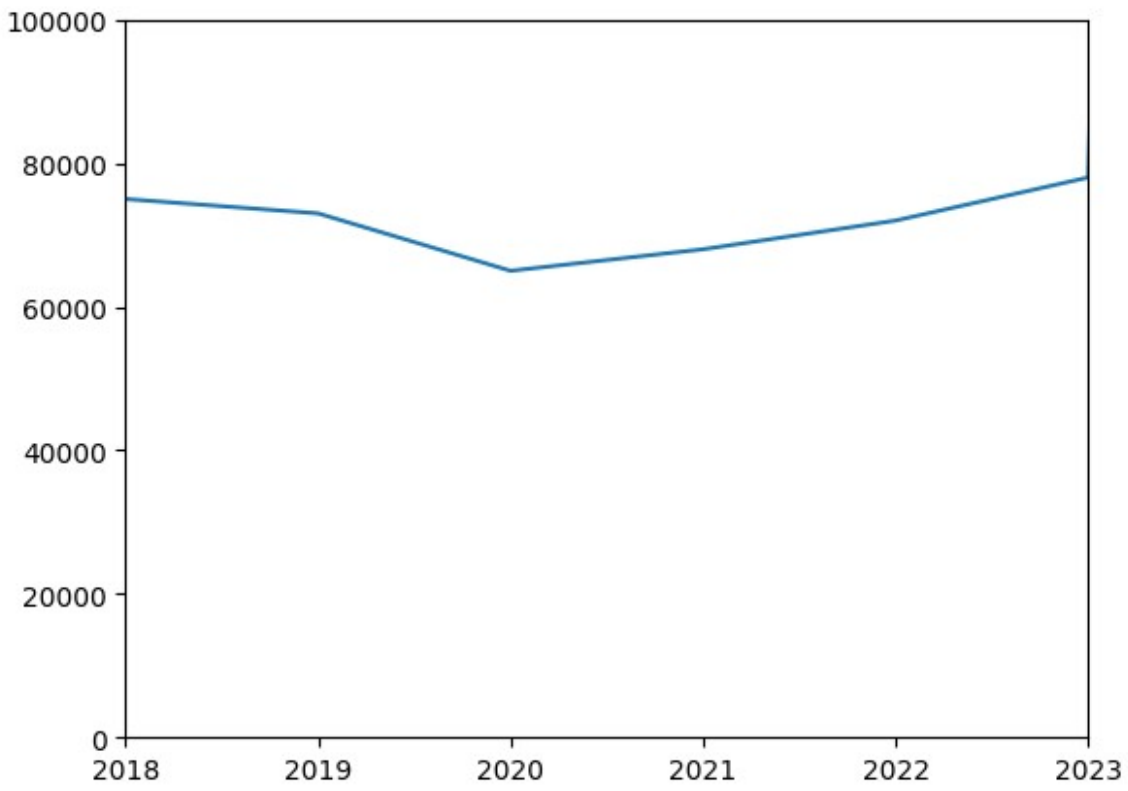
```
price = [75000, 73000, 65000, 68000, 72000, 78000, 1100000]
year = [2018, 2019, 2020, 2021, 2022, 2023, 2024]

plt.plot(year, price)
```

```
[<matplotlib.lines.Line2D at 0x1ab03654f50>]
```



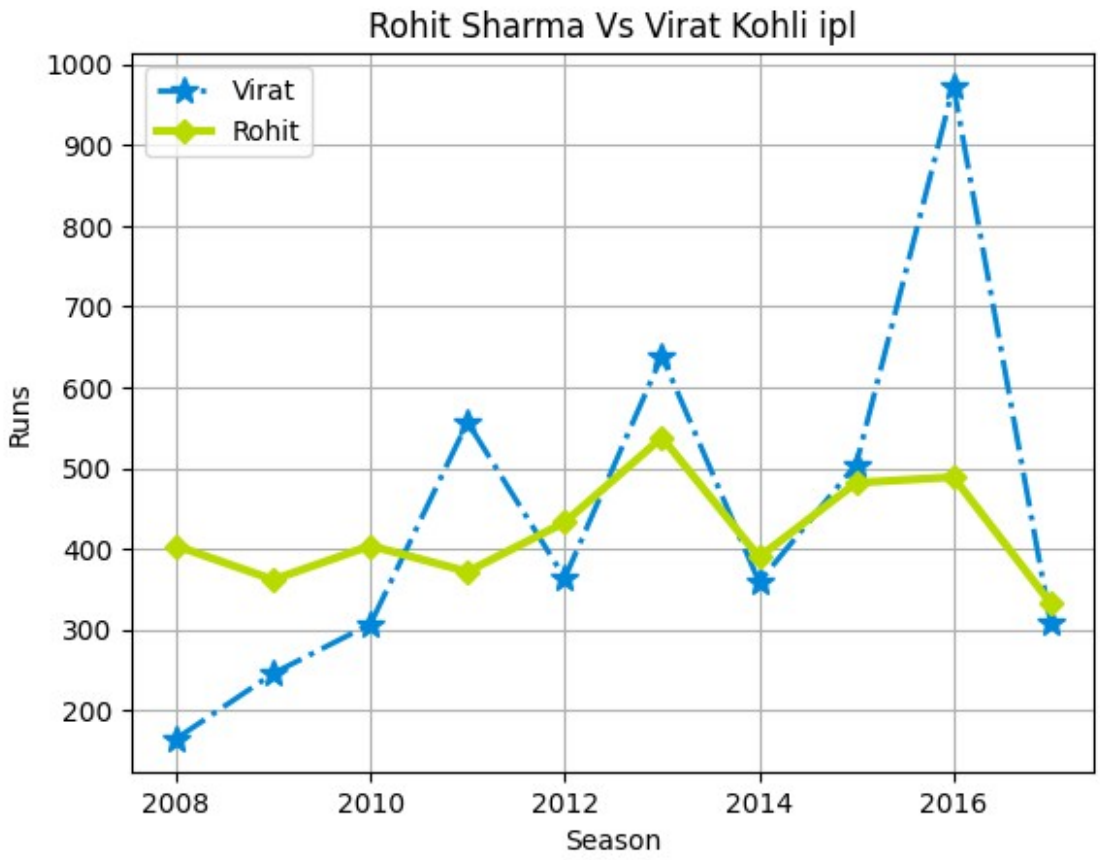

```
price = [75000, 73000, 65000, 68000, 72000, 78000, 1100000]
year = [2018, 2019, 2020, 2021, 2022, 2023, 2024]

plt.plot(year, price)

plt.ylim(0, 100000)
plt.xlim(2018, 2023)
```

```
(2018.0, 2023.0)
```



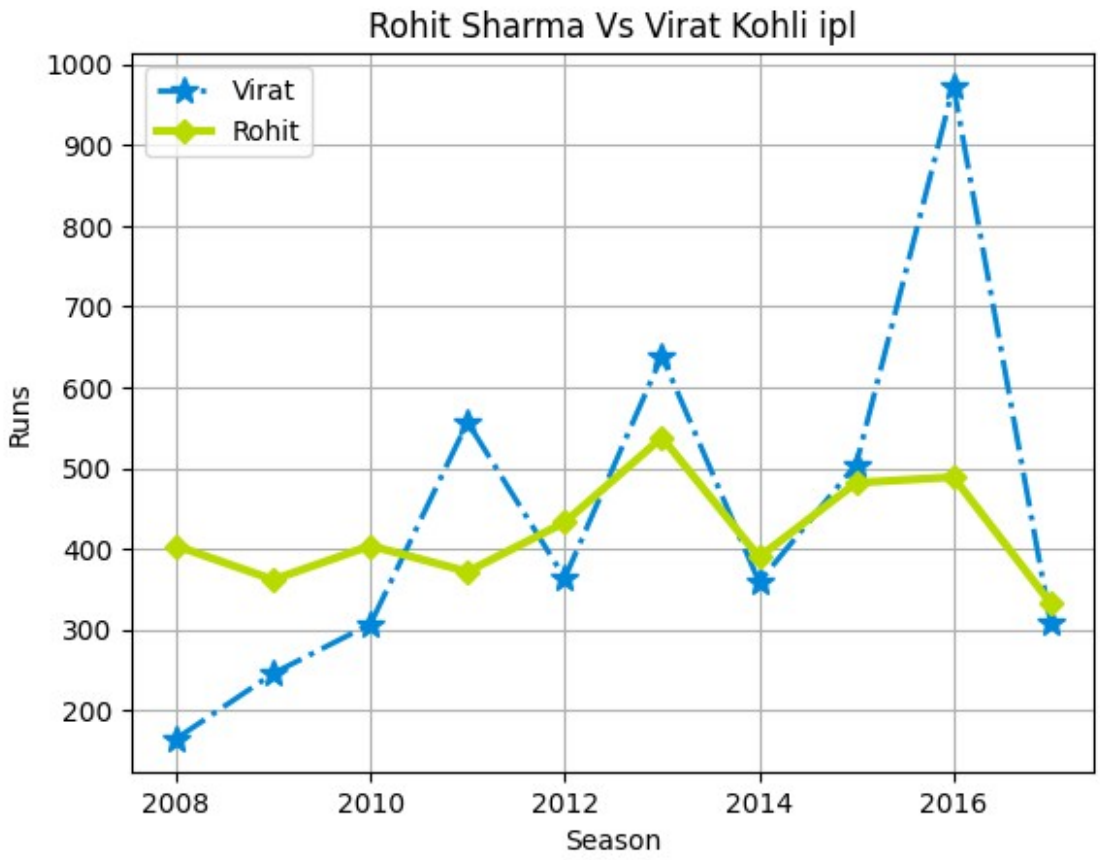
Grid

```
plt.plot(batsman['index'], batsman['V Kohli'], color='#0087db',
linestyle='dashdot', linewidth=2, marker='*', markersize=10,
label='Virat')
plt.plot(batsman['index'], batsman['RG Sharma'], color='#badb00',
linestyle='solid', linewidth=3, marker='D', label='Rohit')

plt.title('Rohit Sharma Vs Virat Kohli ipl')
plt.xlabel('Season')
plt.ylabel('Runs')

plt.legend(loc='upper left')

plt.grid()
```



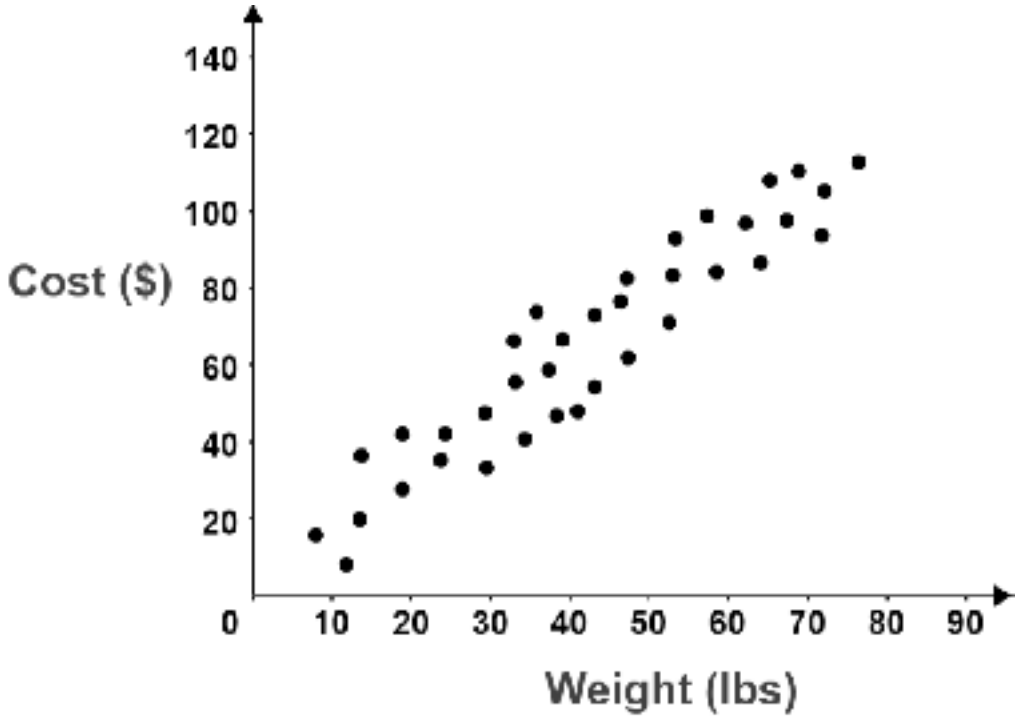
Show

```
plt.plot(batsman['index'], batsman['V Kohli'], color='#0087db',
linestyle='dashdot', linewidth=2, marker='*', markersize=10,
label='Virat')
plt.plot(batsman['index'], batsman['RG Sharma'], color='#badb00',
linestyle='solid', linewidth=3, marker='D', label='Rohit')

plt.title('Rohit Sharma Vs Virat Kohli ipl')
plt.xlabel('Season')
plt.ylabel('Runs')

plt.legend(loc='upper left')

plt.grid()

plt.show()
```



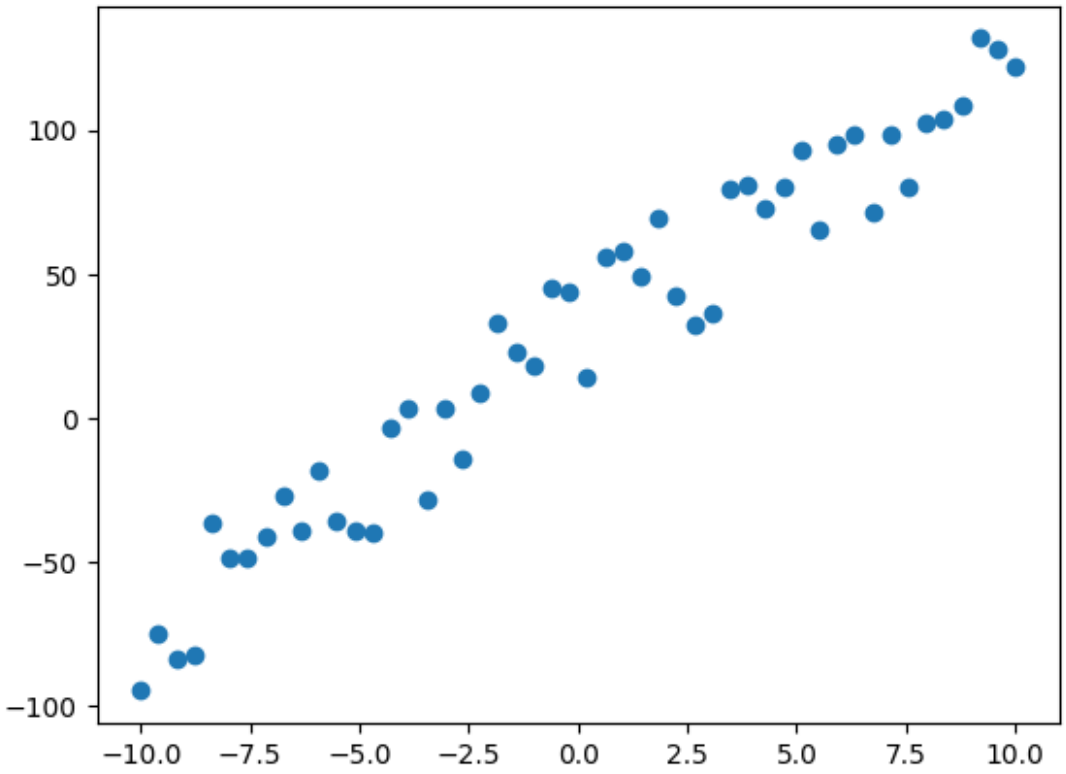
Scatter Plots

- Bivariate Analysis
- numerical vs numerical
- Use case - Finding Correlation




```
x = np.linspace(-10, 10, 50)
y = x*10 + 3 + np.random.randint(1,50,50)

plt.scatter(x, y)
```

```
<matplotlib.collections.PathCollection at 0x1ab03c78da0>
```

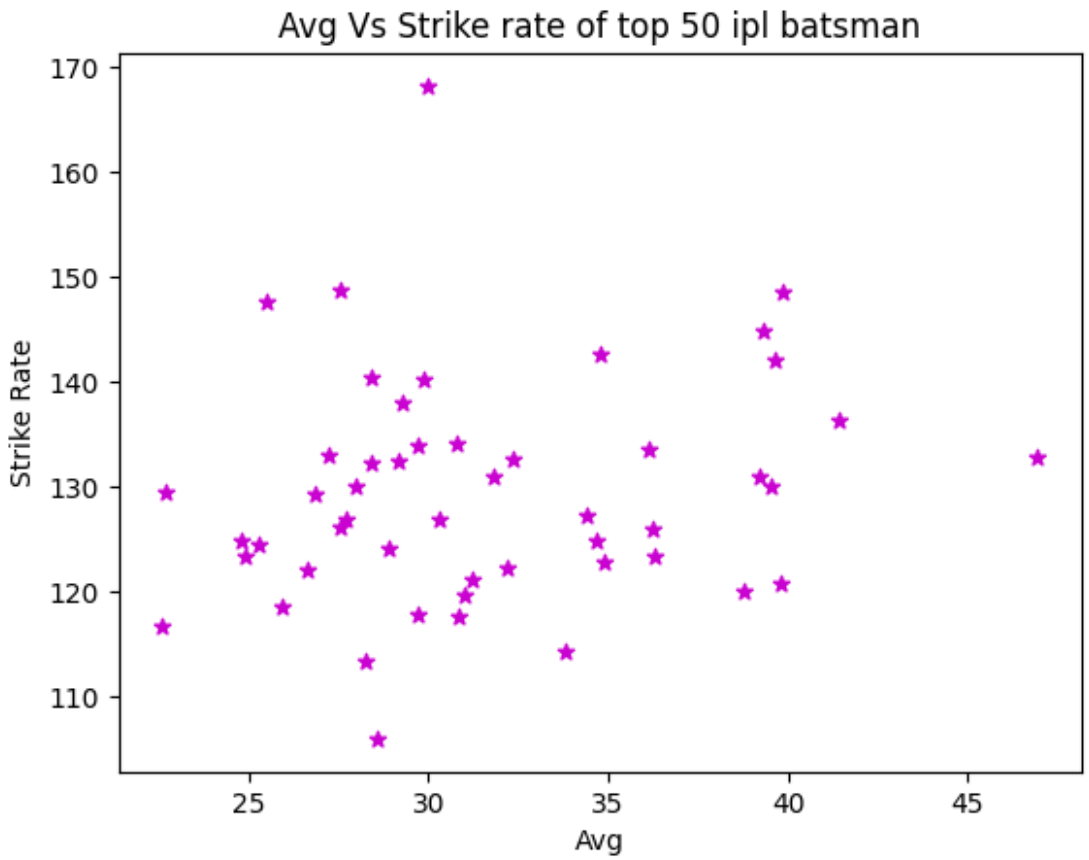



```
df = pd.read_csv('Datasets/batter.csv')
df = df.head(50)
df
```

	batter	runs	avg	strike_rate
0	V Kohli	6634	36.251366	125.977972
1	S Dhawan	6244	34.882682	122.840842
2	DA Warner	5883	41.429577	136.401577
3	RG Sharma	5881	30.314433	126.964594
4	SK Raina	5536	32.374269	132.535312
5	AB de Villiers	5181	39.853846	148.580442
6	CH Gayle	4997	39.658730	142.121729
7	MS Dhoni	4978	39.196850	130.931089
8	RV Uthappa	4954	27.522222	126.152279
9	KD Karthik	4377	26.852761	129.267572
10	G Gambhir	4217	31.007353	119.665153
11	AT Rayudu	4190	28.896552	124.148148
12	AM Rahane	4074	30.863636	117.575758
13	KL Rahul	3895	46.927711	132.799182
14	SR Watson	3880	30.793651	134.163209
15	MK Pandey	3657	29.731707	117.739858
16	SV Samson	3526	29.140496	132.407060
17	KA Pollard	3437	28.404959	140.457703
18	F du Plessis	3403	34.373737	127.167414
19	YK Pathan	3222	29.290909	138.046272

```
20     BB McCullum  2882  27.711538   126.848592
21         RR Pant  2851  34.768293   142.550000
22        PA Patel  2848  22.603175   116.625717
23      JC Buttler  2832  39.333333   144.859335
24         SS Iyer  2780  31.235955   121.132898
25       Q de Kock  2767  31.804598   130.951254
26    Yuvraj Singh  2754  24.810811   124.784776
27        V Sehwag  2728  27.555556   148.827059
28        SA Yadav  2644  29.707865   134.009123
29         M Vijay  2619  25.930693   118.614130
30       RA Jadeja  2502  26.617021   122.108346
31       SPD Smith  2495  34.652778   124.812406
32        SE Marsh  2489  39.507937   130.109775
33       DA Miller  2455  36.102941   133.569097
34       JH Kallis  2427  28.552941   105.936272
35         WP Saha  2427  25.281250   124.397745
36        DR Smith  2385  28.392857   132.279534
37      MA Agarwal  2335  22.669903   129.506378
38    SR Tendulkar  2334  33.826087   114.187867
39      GJ Maxwell  2320  25.494505   147.676639
40          N Rana  2181  27.961538   130.053667
41        R Dravid  2174  28.233766   113.347237
42   KS Williamson  2105  36.293103   123.315759
43        AJ Finch  2092  24.904762   123.349057
44    AC Gilchrist  2069  27.223684   133.054662
45      AD Russell  2039  29.985294   168.234323
46       JP Duminy  2029  39.784314   120.773810
47      MEK Hussey  1977  38.764706   119.963592
48       HH Pandya  1972  29.878788   140.256046
49    Shubman Gill  1900  32.203390   122.186495
```

```
plt.scatter(df['avg'], df['strike_rate'], color='#ca02d1', marker='*')

plt.title('Avg Vs Strike rate of top 50 ipl batsman')
plt.xlabel('Avg')
plt.ylabel('Strike Rate')
plt.show()
```

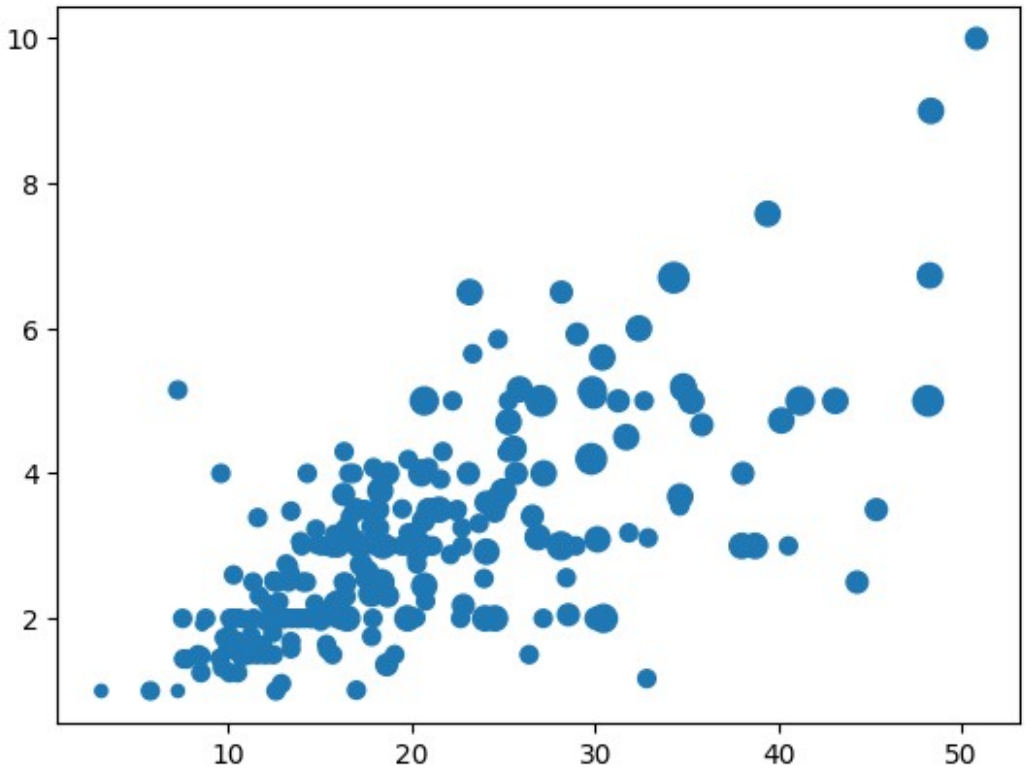


Size

- Size of marker based on values of third column from dataset
- 3D data

```
tips = sns.load_dataset('tips')

plt.scatter(tips['total_bill'], tips['tip'], s=tips['size']*20)
```

```
<matplotlib.collections.PathCollection at 0x1ab03bbe030>
```

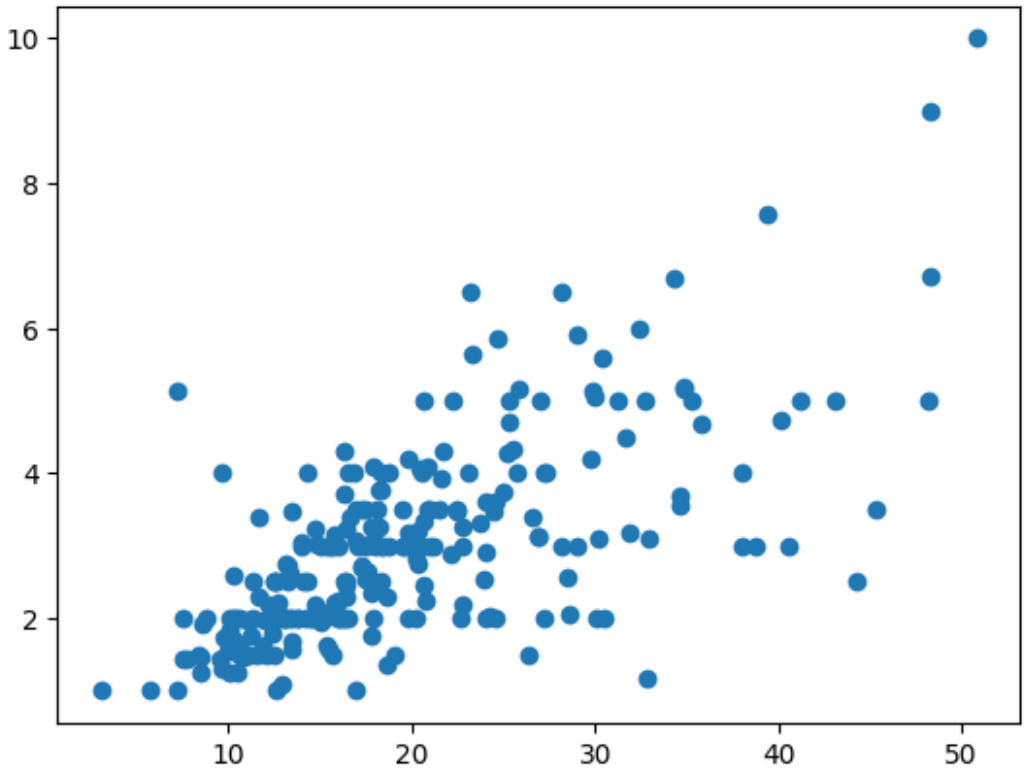


Scatter plot using plt.plot

- Some limitations with customization but highly faster

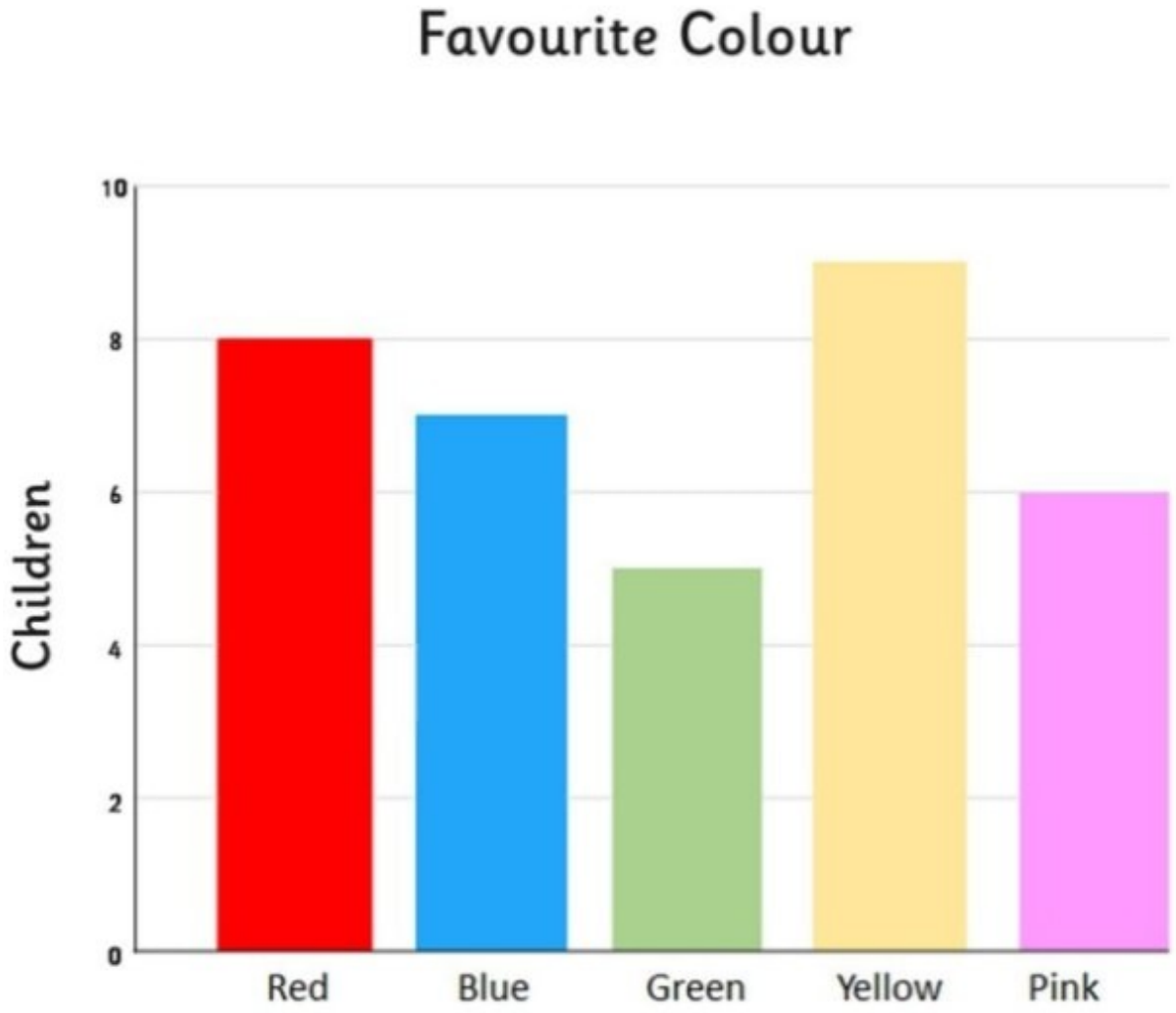
```
plt.plot(tips['total_bill'], tips['tip'], 'o')
```

```
[<matplotlib.lines.Line2D at 0x1ab03c49310>]
```

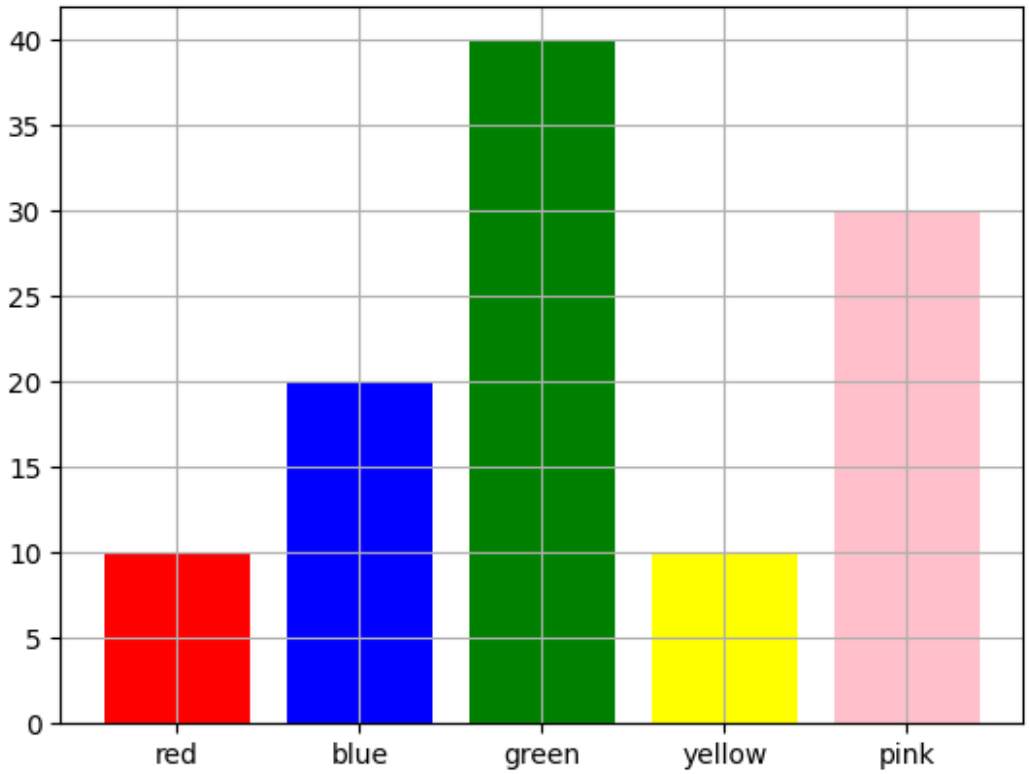


Bar Chart

- Univariate/Bivariate Analysis
- numerical - categorical
- Use case - Aggregate analysis of groups



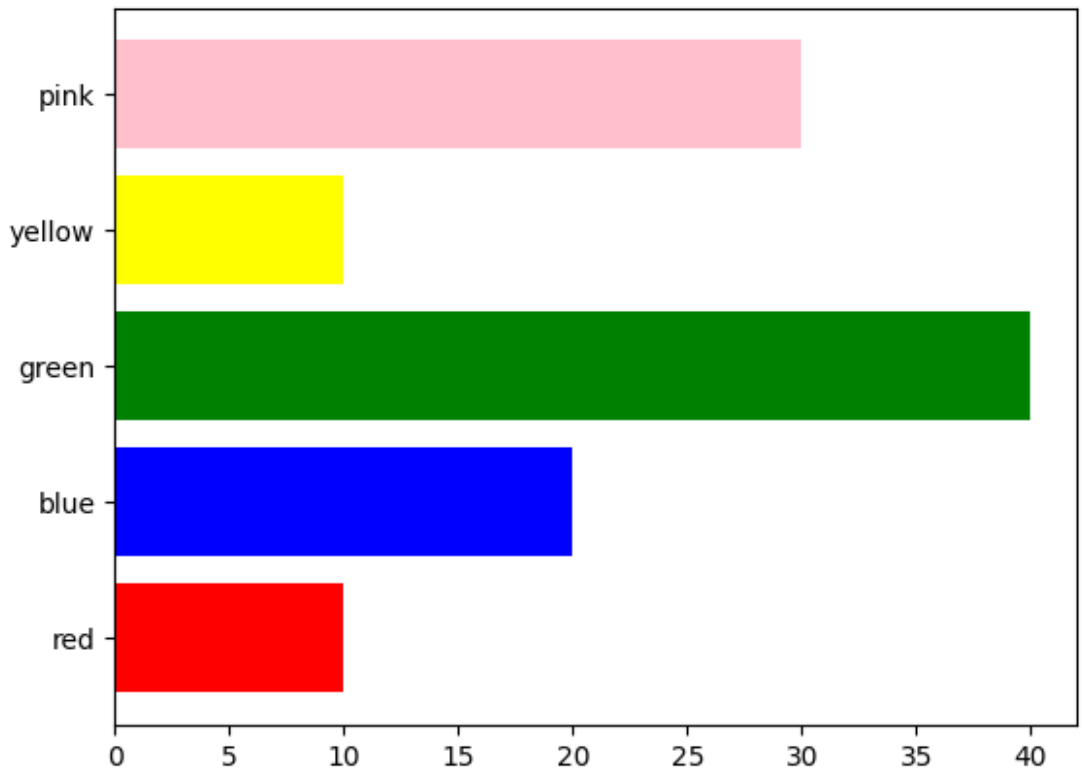

```
children = [10, 20, 40, 10, 30]
colors = ['red', 'blue', 'green', 'yellow', 'pink']

plt.bar(colors, children, color=['red', 'blue', 'green', 'yellow',
'pink'])
plt.grid()
```



Horizontal Bar Chart

```
plt.barh(colors, children, color=['red', 'blue', 'green', 'yellow',
'pink'])
```

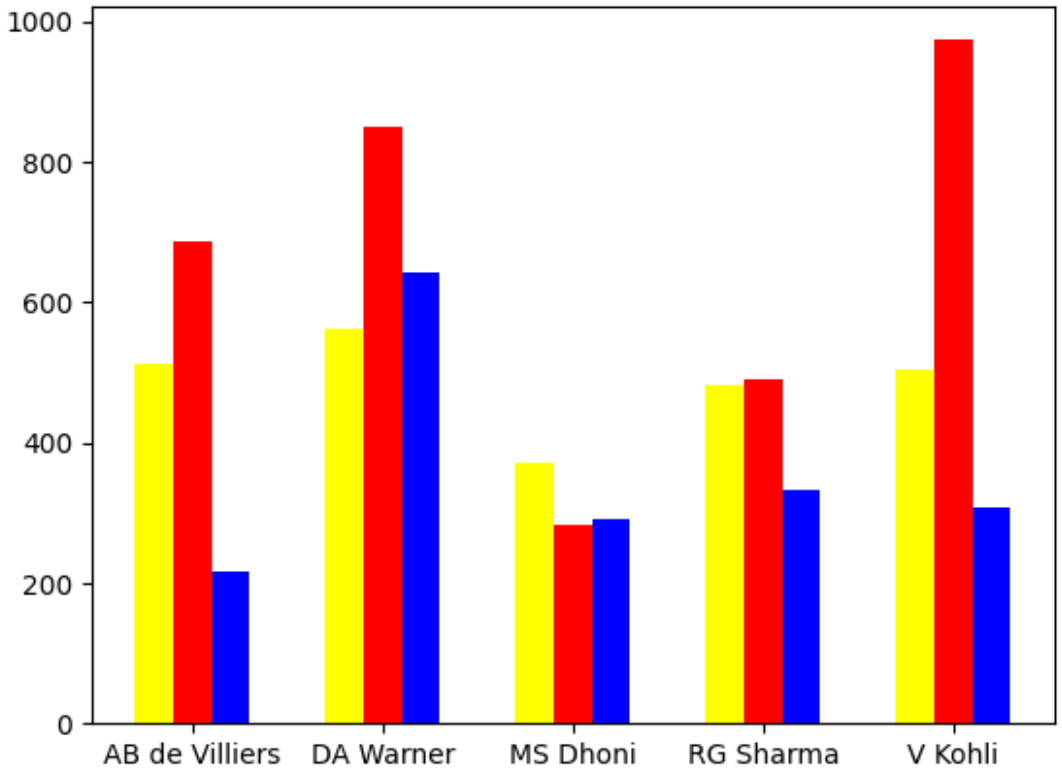
```
<BarContainer object of 5 artists>
```




```
df = pd.read_csv('Datasets/batsman_season_record.csv')
df
```

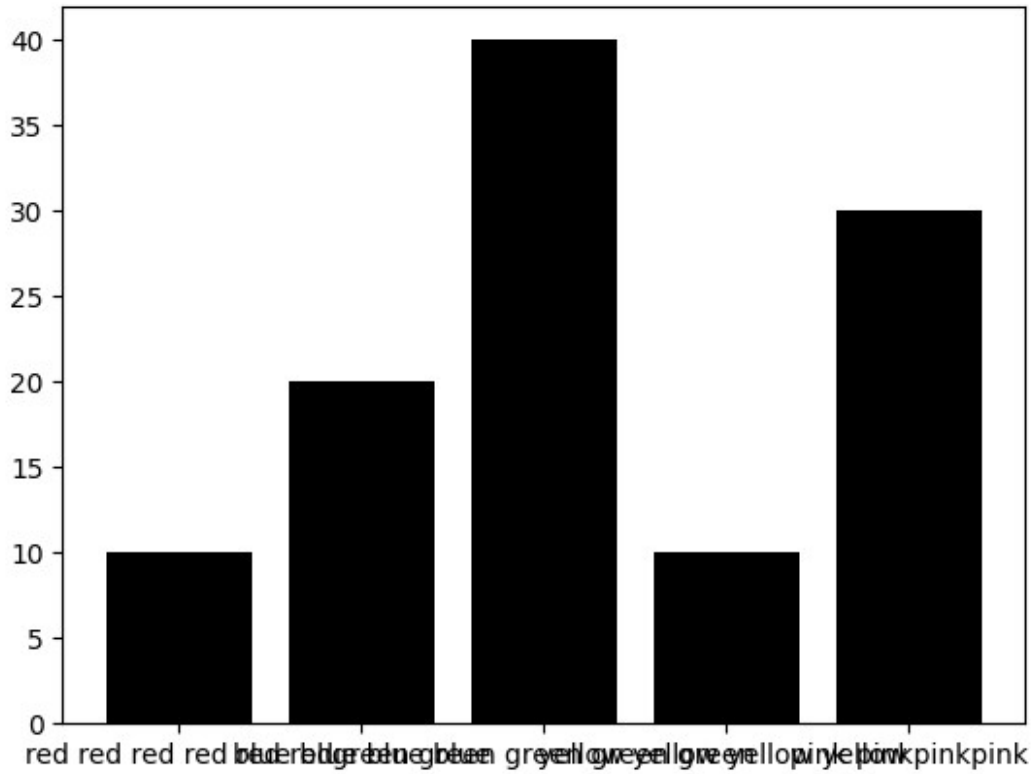
	batsman	2015	2016	2017
0	AB de Villiers	513	687	216
1	DA Warner	562	848	641
2	MS Dhoni	372	284	290
3	RG Sharma	482	489	333
4	V Kohli	505	973	308

```
plt.bar(np.arange(df.shape[0])-0.2, df['2015'], width=0.2,
color='yellow')
plt.bar(np.arange(df.shape[0]), df['2016'], width=0.2, color='red')
plt.bar(np.arange(df.shape[0])+0.2, df['2017'], width=0.2,
color='blue')

plt.xticks(np.arange(df.shape[0]), df['batsman'])

plt.show()
```

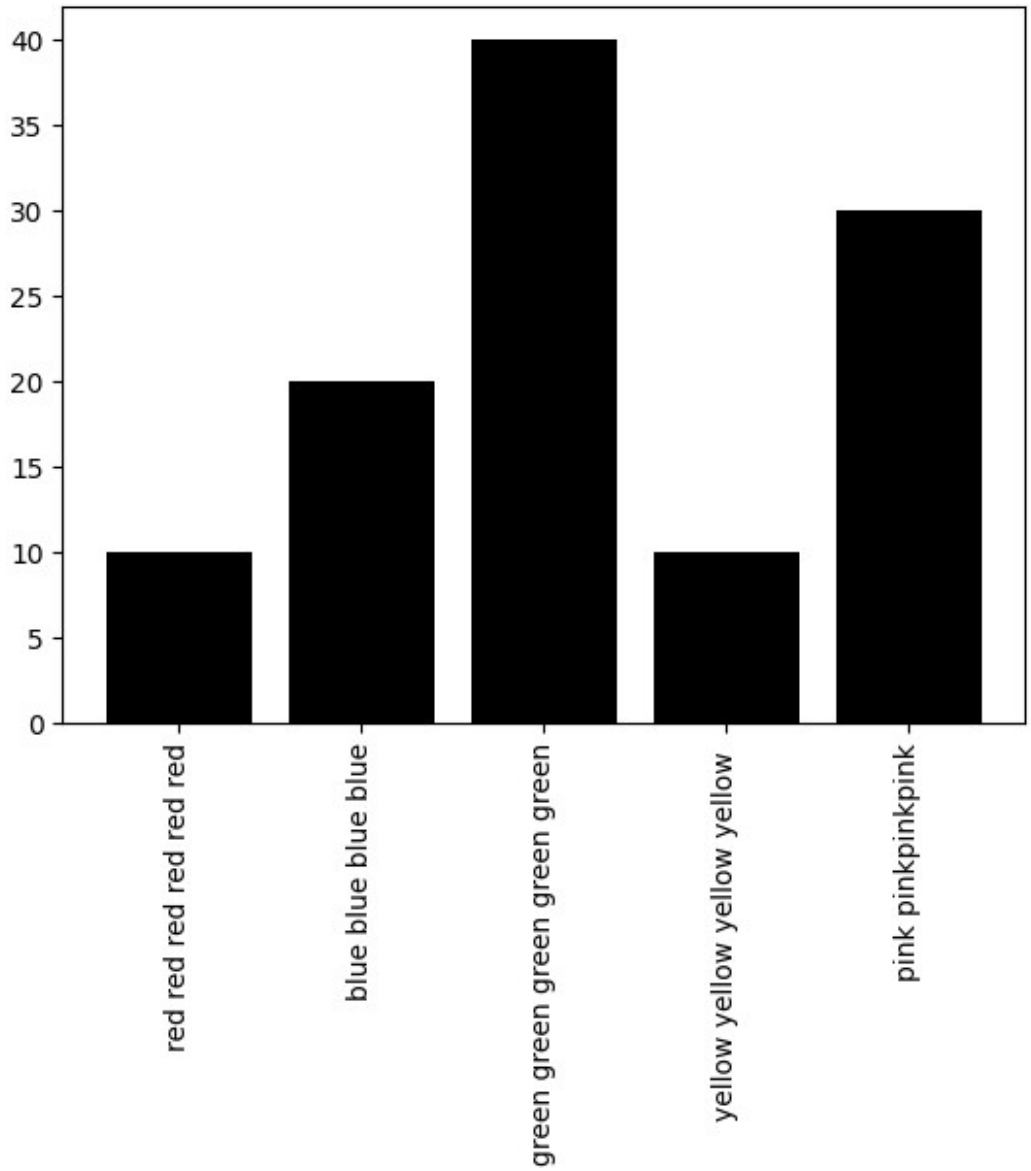


The Problem of longer category name

```
children = [10,20,40,10,30]
colors = ['red red red red red red','blue blue blue blue','green green
green green green','yellow yellow yellow yellow ','pink pinkpinkpink']

plt.bar(colors,children,color='black')

plt.show()
```

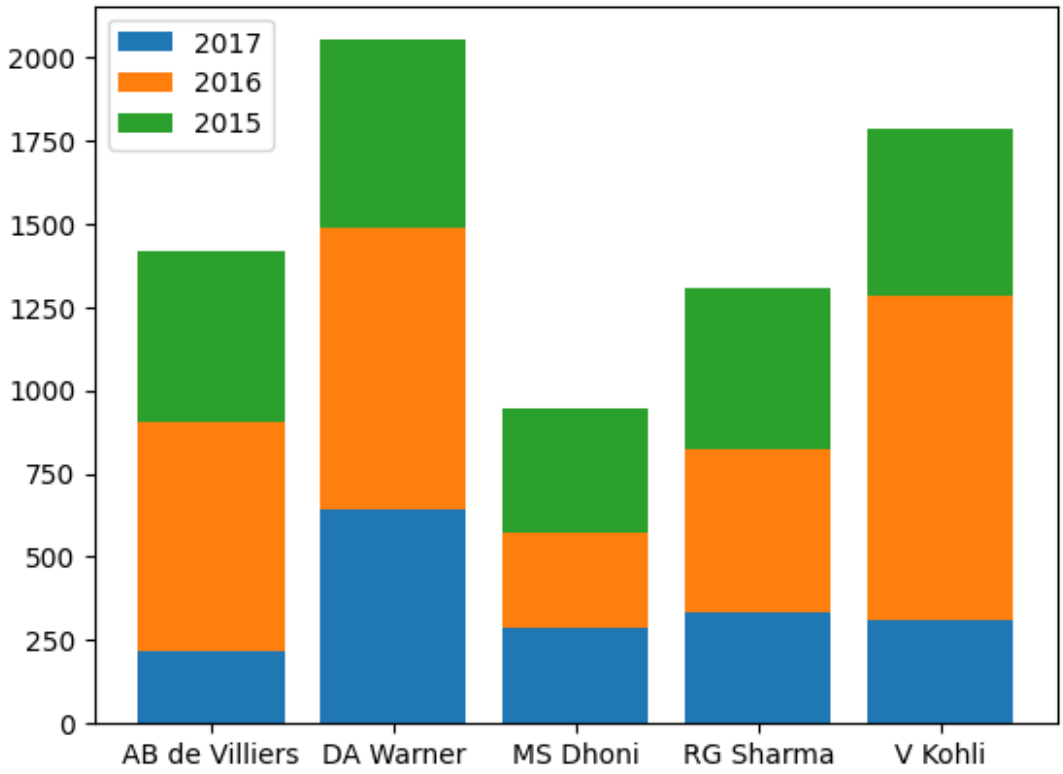



```
children = [10,20,40,10,30]
colors = ['red red red red red red','blue blue blue blue','green green
green green green','yellow yellow yellow yellow ','pink pinkpinkpink']

plt.bar(colors,children,color='black')
plt.xticks(rotation='vertical')

plt.show()
```



Stacked Bar Chart

```
plt.bar(df['batsman'],df['2017'],label='2017')
plt.bar(df['batsman'],df['2016'],bottom=df['2017'],label='2016')
plt.bar(df['batsman'],df['2015'],bottom=(df['2016'] +
df['2017']),label='2015')

plt.legend()
plt.show()
```

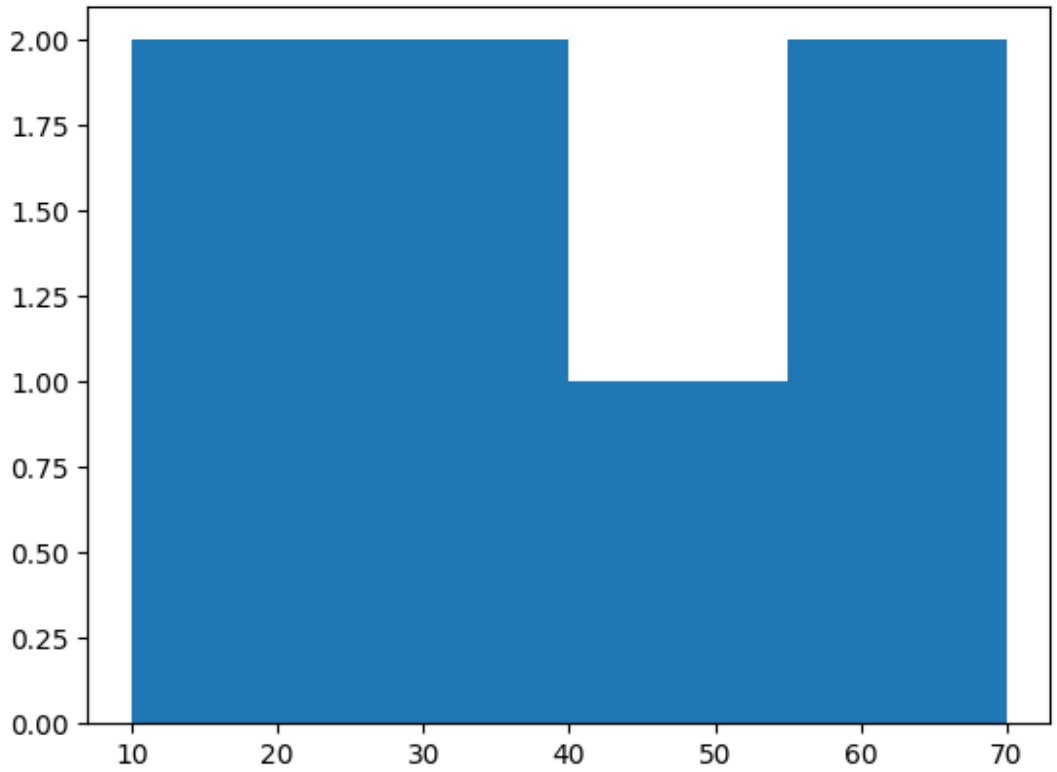


Histogram

- Univariate Analysis
- numerical col
- Use case - Frequency count

```
data = [32, 45, 56, 10, 15, 27, 61]
plt.hist(data, bins=[10, 25, 40, 55, 70])
```

```
(array([2., 2., 1., 2.]),
 array([10., 25., 40., 55., 70.]),
 <BarContainer object of 4 artists>)
```

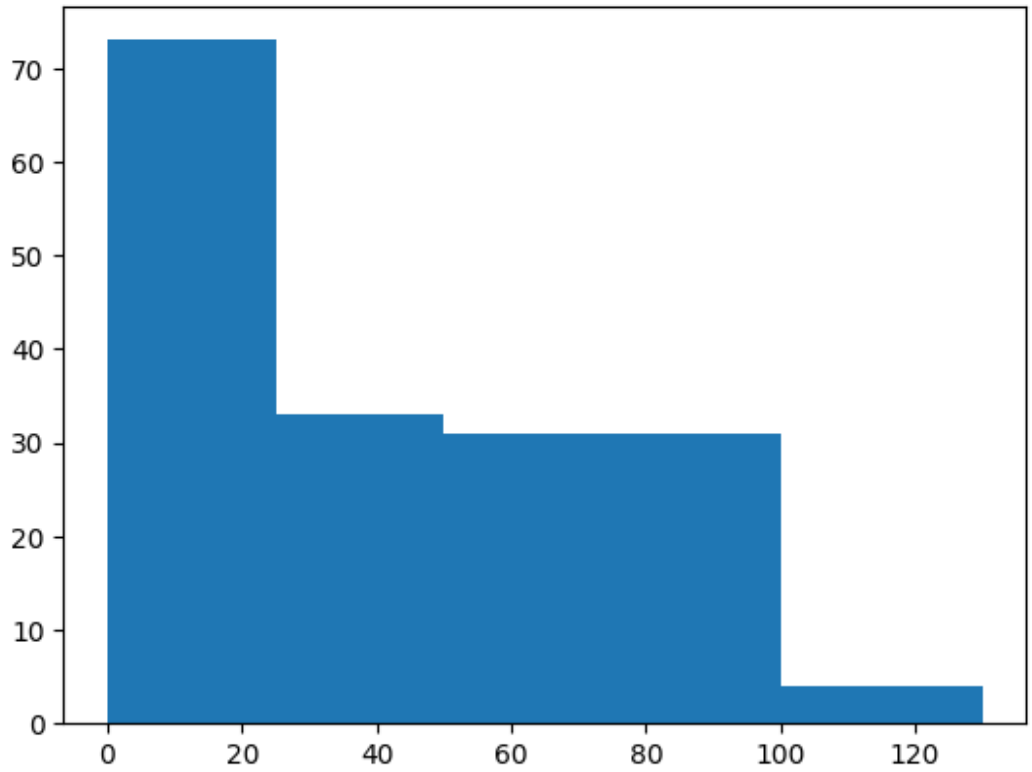



```
df = pd.read_csv('Datasets/vk.csv')
df
```

```
     match_id  batsman_runs
0          12            62
1          17            28
2          20            64
3          27             0
4          30            10
..        ...           ...
136       624            75
137       626           113
138       632            54
139       633             0
140       636            54

[141 rows x 2 columns]
```

```
plt.hist(df['batsman_runs'], bins=[0,25,50,100,130])
```

```
(array([73., 33., 31.,  4.]),
 array([  0.,  25.,  50., 100., 130.]),
 <BarContainer object of 4 artists>)
```

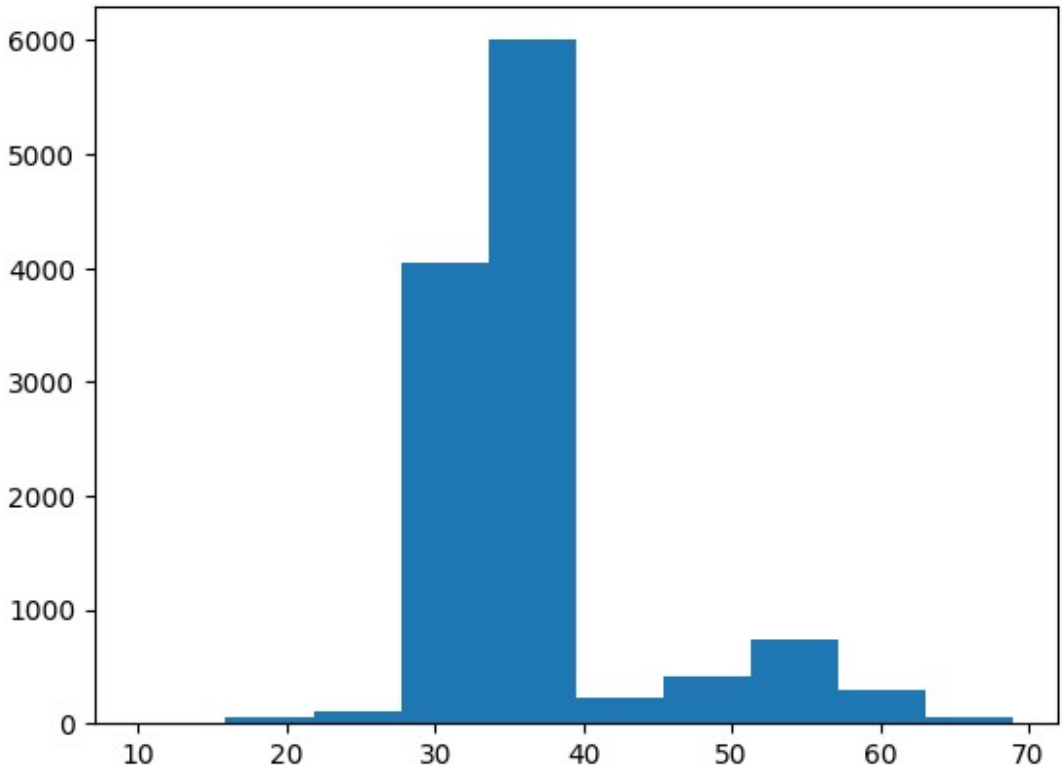


The Problem

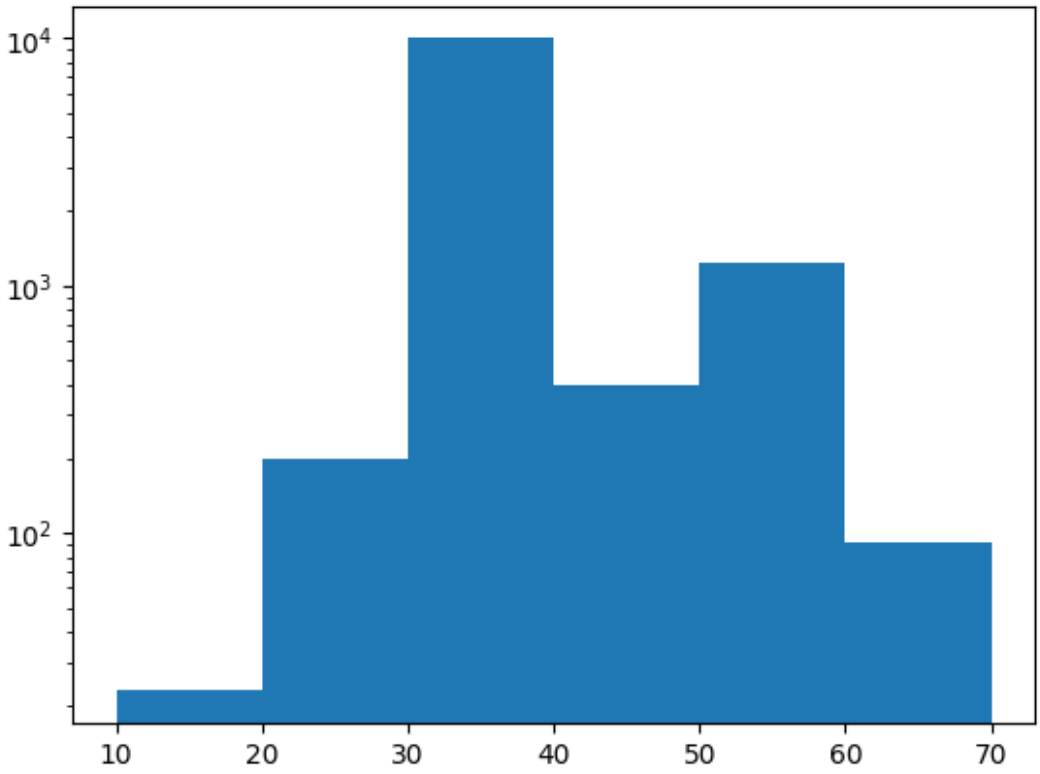
- When values in a range is too much in comparison with another range values.
- then information loss occur
- It can be solved using logarithmic function in which 0-10 is equal as 10-100 and 100-1000 and so on

```
arr = np.load('big-array.npy')
plt.hist(arr)
```

```
(array([  12.,   60.,  109., 4039., 6003.,  230.,  410.,  744.,  291.,
         51.]),
 array([10. , 15.9, 21.8, 27.7, 33.6, 39.5, 45.4, 51.3, 57.2, 63.1,
69. ]),
 <BarContainer object of 10 artists>)
```

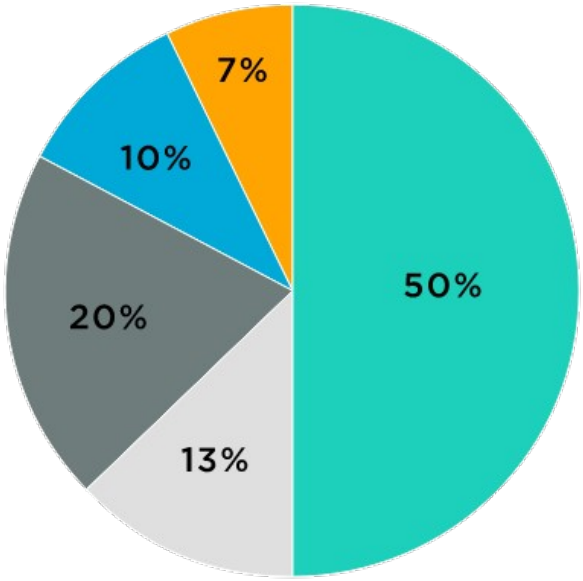



```
plt.hist(arr, bins=[10,20,30,40,50,60,70], log=True)
plt.show()
```

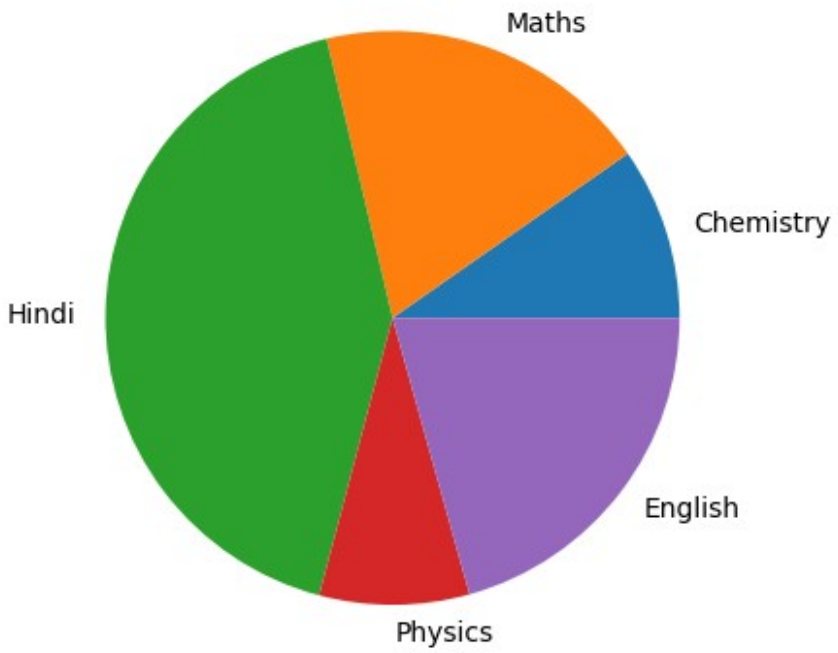


Pie Chart

- Univariate/Bivariate Analysis
- categorical vs numerical
- Use case - to find contribution on a standard scale



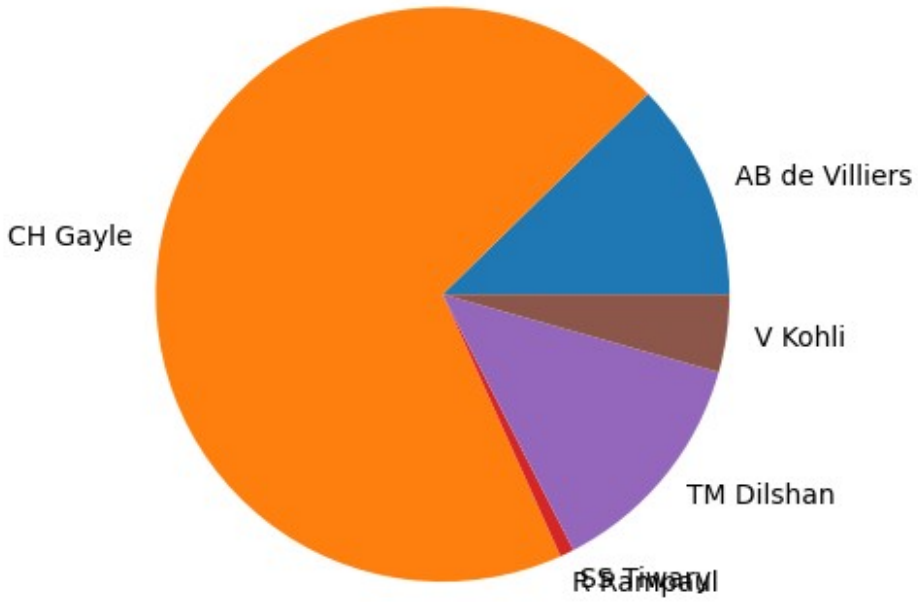
```
data = [23, 45, 100, 20, 49]
subjects = ['Chemistry', 'Maths', 'Hindi', 'Physics', 'English']

plt.pie(data, labels=subjects)

plt.show()
```




```
df = pd.read_csv('Datasets/gayle-175.csv')
df
```

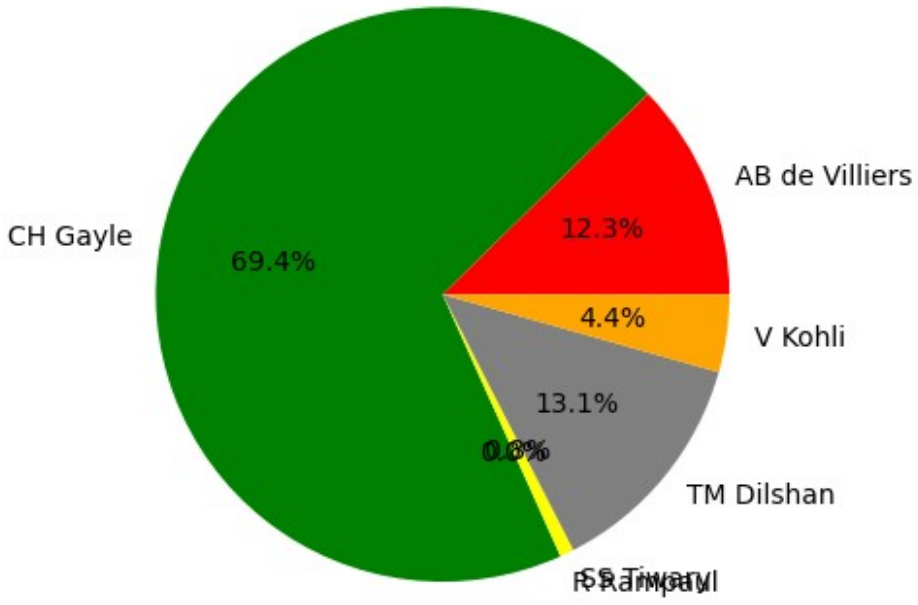
	batsman	batsman_runs
0	AB de Villiers	31
1	CH Gayle	175
2	R Rampaul	0
3	SS Tiwary	2
4	TM Dilshan	33
5	V Kohli	11

```
plt.pie(df['batsman_runs'], labels=df['batsman'])

plt.show()
```



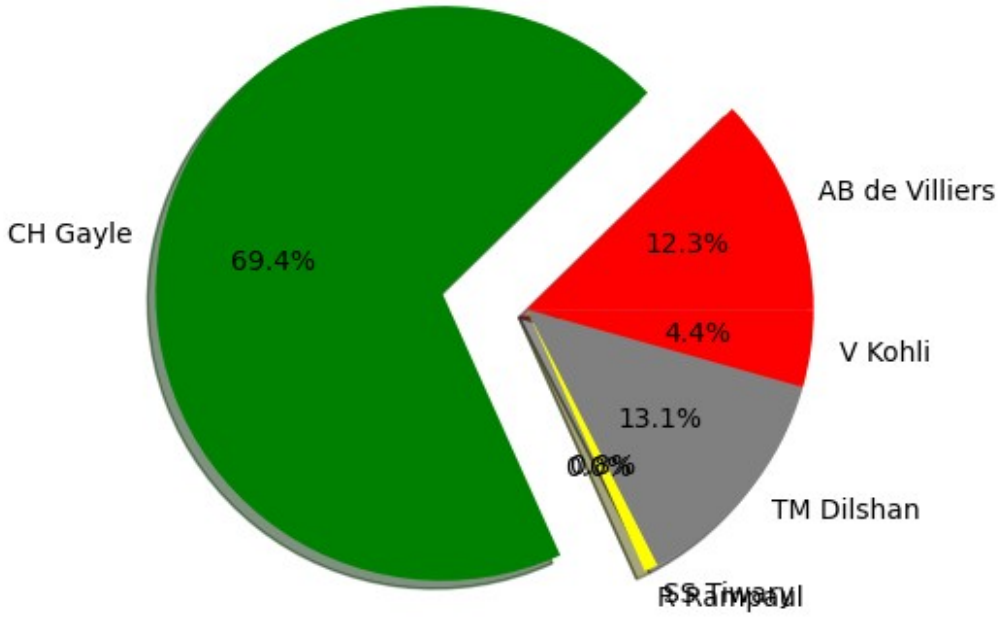
Percentage and Color

```
plt.pie(df['batsman_runs'], labels=df['batsman'], autopct='%0.1f%%',
colors=['red', 'green', 'blue', 'yellow', 'gray', 'orange', 'white'])

plt.show()
```



Explode and Shadow

```
plt.pie(df['batsman_runs'], labels=df['batsman'], autopct='%0.1f%%',
colors=['red', 'green', 'blue', 'yellow', 'gray'],
explode=[0,0.3,0,0,0,0], shadow=True)

plt.show()
```

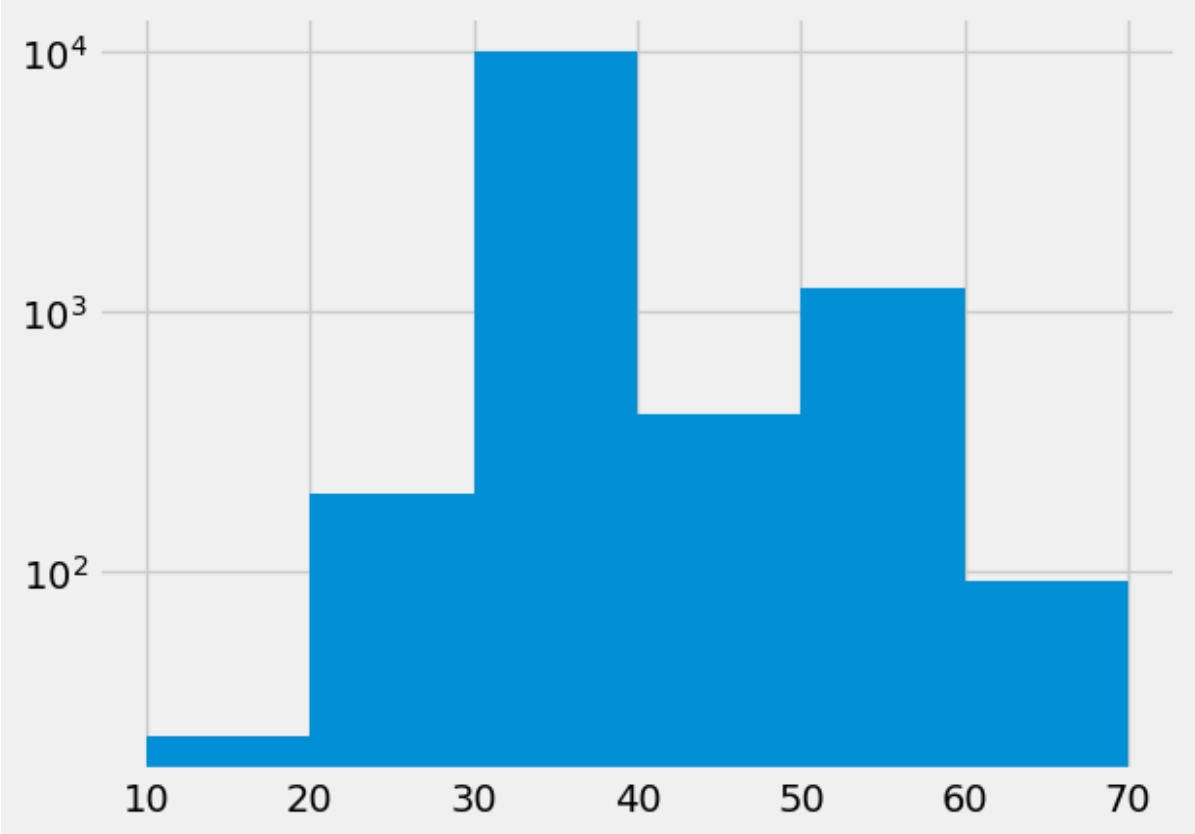


Changing Styles

```
plt.style.available
```

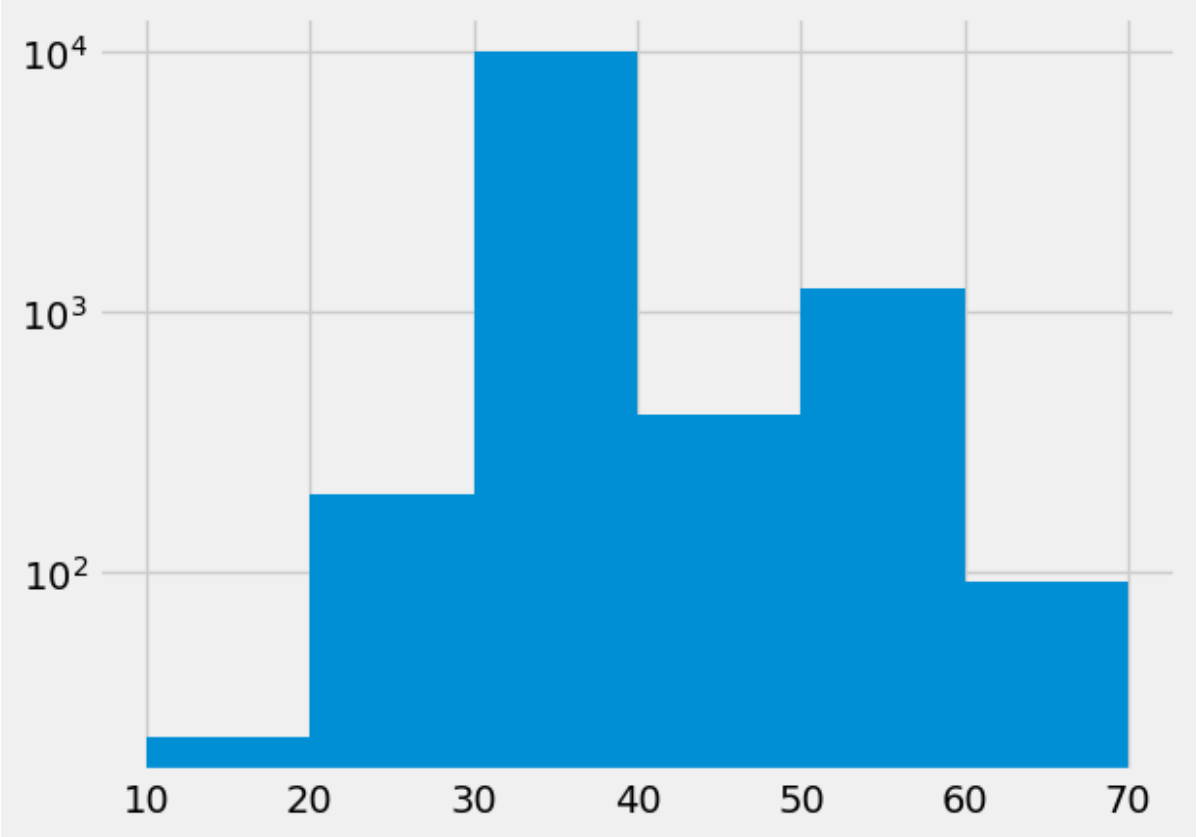
```
['Solarize_Light2',
 '_classic_test_patch',
 '_mpl-gallery',
 '_mpl-gallery-nogrid',
 'bmh',
 'classic',
 'dark_background',
 'fast',
 'fivethirtyeight',
 'ggplot',
 'grayscale',
 'seaborn-v0_8',
 'seaborn-v0_8-bright',
 'seaborn-v0_8-colorblind',
 'seaborn-v0_8-dark',
 'seaborn-v0_8-dark-palette',
 'seaborn-v0_8-darkgrid',
 'seaborn-v0_8-deep',
 'seaborn-v0_8-muted',
 'seaborn-v0_8-notebook',
 'seaborn-v0_8-paper',
 'seaborn-v0_8-pastel',
 'seaborn-v0_8-poster',
```

```
 'seaborn-v0_8-talk',
 'seaborn-v0_8-ticks',
 'seaborn-v0_8-white',
 'seaborn-v0_8-whitegrid',
 'tableau-colorblind10']

plt.style.use('fivethirtyeight')

arr = np.load('Datasets/big-array.npy')
plt.hist(arr, bins=[10,20,30,40,50,60,70], log=True)
plt.show()
```



Saving Figure

```
arr = np.load('Datasets/big-array.npy')
plt.hist(arr, bins=[10,20,30,40,50,60,70], log=True)
plt.savefig('Datasets/sample.png')
```




```
plt.style.use('default')
```

Colored Scatterplots

```
iris = sns.load_dataset('iris')
iris.sample(5)
```

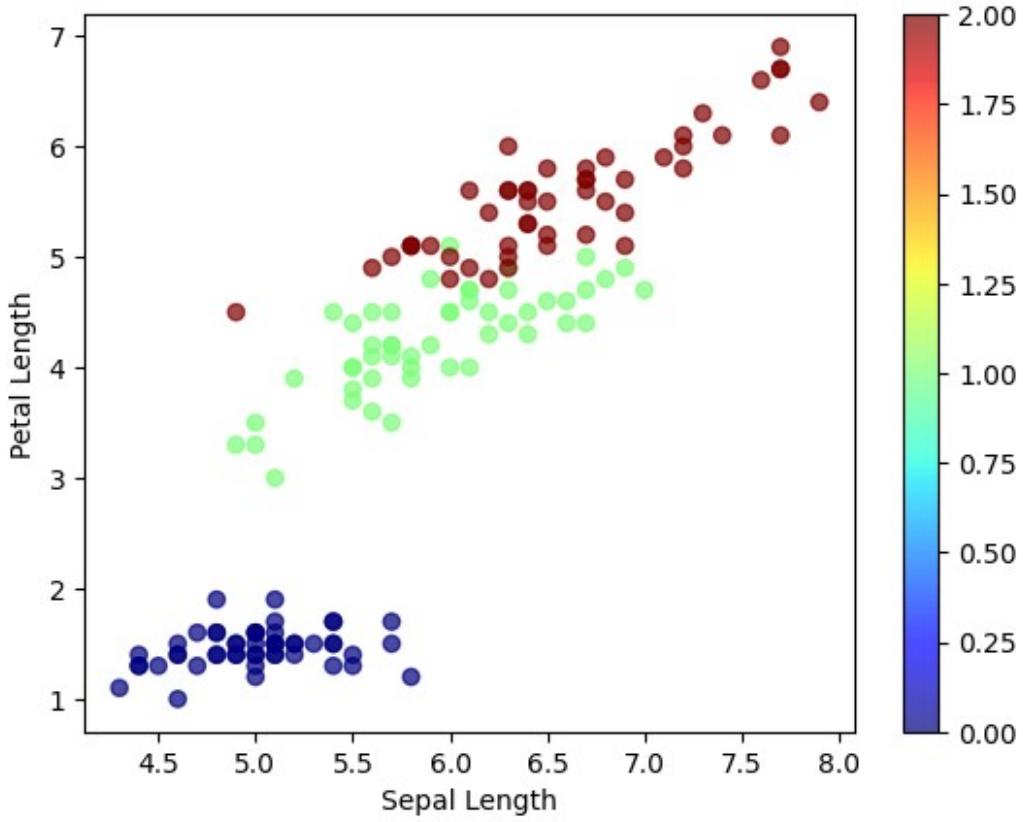
	sepal_length	sepal_width	petal_length	petal_width	species
46	5.1	3.8	1.6	0.2	setosa
125	7.2	3.2	6.0	1.8	virginica
26	5.0	3.4	1.6	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
145	6.7	3.0	5.2	2.3	virginica

```
iris['species'] = iris['species'].replace({'setosa': 0, 'versicolor':
1, 'virginica': 2}).infer_objects(copy=False)
iris.sample(5)
```

	sepal_length	sepal_width	petal_length	petal_width	species
103	6.3	2.9	5.6	1.8	2
24	4.8	3.4	1.9	0.2	0
141	6.9	3.1	5.1	2.3	2
135	7.7	3.0	6.1	2.3	2
146	6.3	2.5	5.0	1.9	2

```
plt.scatter(iris['sepal_length'], iris['petal_length'],
c=iris['species'], cmap='jet', alpha=0.7)

plt.xlabel('Sepal Length')
plt.ylabel('Petal Length')

plt.colorbar()
```

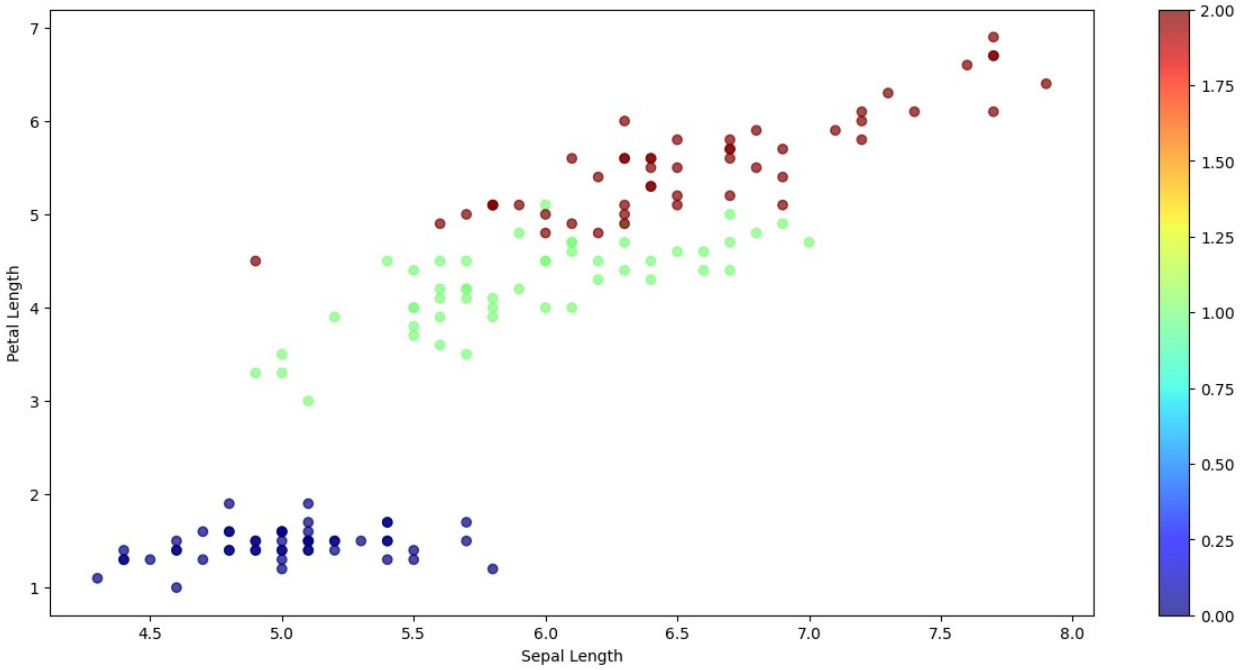
```
<matplotlib.colorbar.Colorbar at 0x1ab05906660>
```



Plot Size

```
plt.figure(figsize=(15,7))

plt.scatter(iris['sepal_length'], iris['petal_length'],
c=iris['species'], cmap='jet', alpha=0.7)

plt.xlabel('Sepal Length')
plt.ylabel('Petal Length')

plt.colorbar()
```

```
<matplotlib.colorbar.Colorbar at 0x1ab05764ef0>
```



Annotations

- Labeling plotted data points

```
batters = pd.read_csv('Datasets/batter.csv')
batters.head()
```

```
      batter  runs        avg  strike_rate
0    V Kohli  6634  36.251366   125.977972
1   S Dhawan  6244  34.882682   122.840842
2  DA Warner  5883  41.429577   136.401577
3  RG Sharma  5881  30.314433   126.964594
4   SK Raina  5536  32.374269   132.535312
```

```
batters.shape
```

```
(605, 4)
```

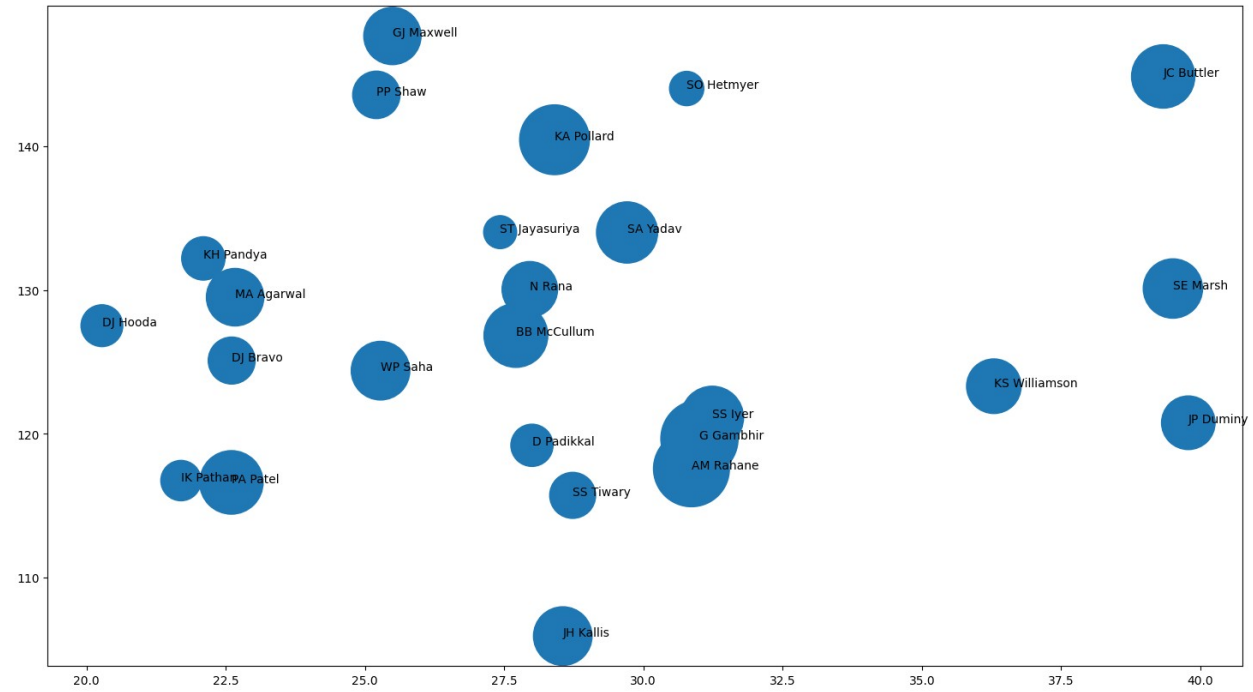
```
sample_df = batters.head(100).sample(25, random_state=5)
```

```
sample_df
```

```
           batter  runs        avg  strike_rate
66      KH Pandya  1326  22.100000   132.203390
32       SE Marsh  2489  39.507937   130.109775
46      JP Duminy  2029  39.784314   120.773810
28       SA Yadav  2644  29.707865   134.009123
74      IK Pathan  1150  21.698113   116.751269
23     JC Buttler  2832  39.333333   144.859335
10      G Gambhir  4217  31.007353   119.665153
20    BB McCullum  2882  27.711538   126.848592
```

```
17     KA Pollard  3437  28.404959   140.457703
35        WP Saha  2427  25.281250   124.397745
97  ST Jayasuriya   768  27.428571   134.031414
37     MA Agarwal  2335  22.669903   129.506378
70       DJ Hooda  1237  20.278689   127.525773
40          N Rana  2181  27.961538   130.053667
60      SS Tiwary  1494  28.730769   115.724245
34      JH Kallis  2427  28.552941   105.936272
42  KS Williamson  2105  36.293103   123.315759
57       DJ Bravo  1560  22.608696   125.100241
12      AM Rahane  4074  30.863636   117.575758
69     D Padikkal  1260  28.000000   119.205298
94     SO Hetmyer   831  30.777778   144.020797
56        PP Shaw  1588  25.206349   143.580470
22        PA Patel  2848  22.603175   116.625717
39     GJ Maxwell  2320  25.494505   147.676639
24        SS Iyer  2780  31.235955   121.132898
```

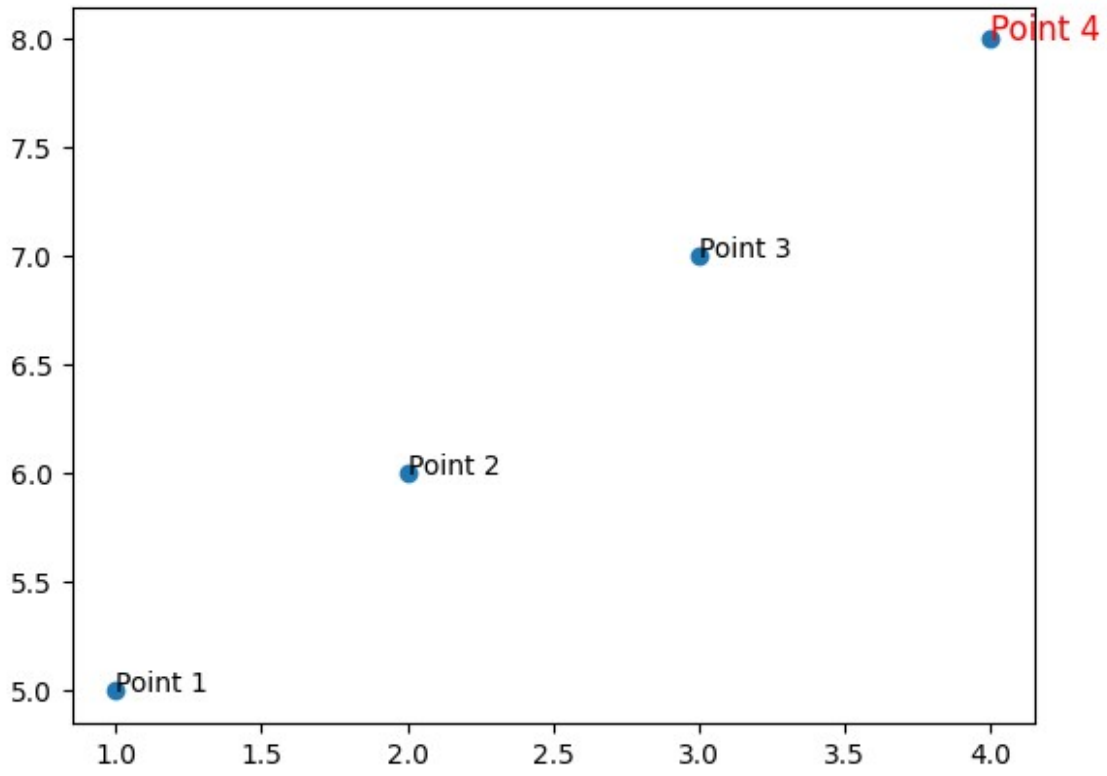
```
plt.figure(figsize=(18,10))
plt.scatter(sample_df['avg'], sample_df['strike_rate'],
s=sample_df['runs'])

for i in range(sample_df.shape[0]):
    plt.text(sample_df['avg'].values[i],
sample_df['strike_rate'].values[i], sample_df['batter'].values[i])
```




```
x = [1,2,3,4]
y = [5,6,7,8]
```

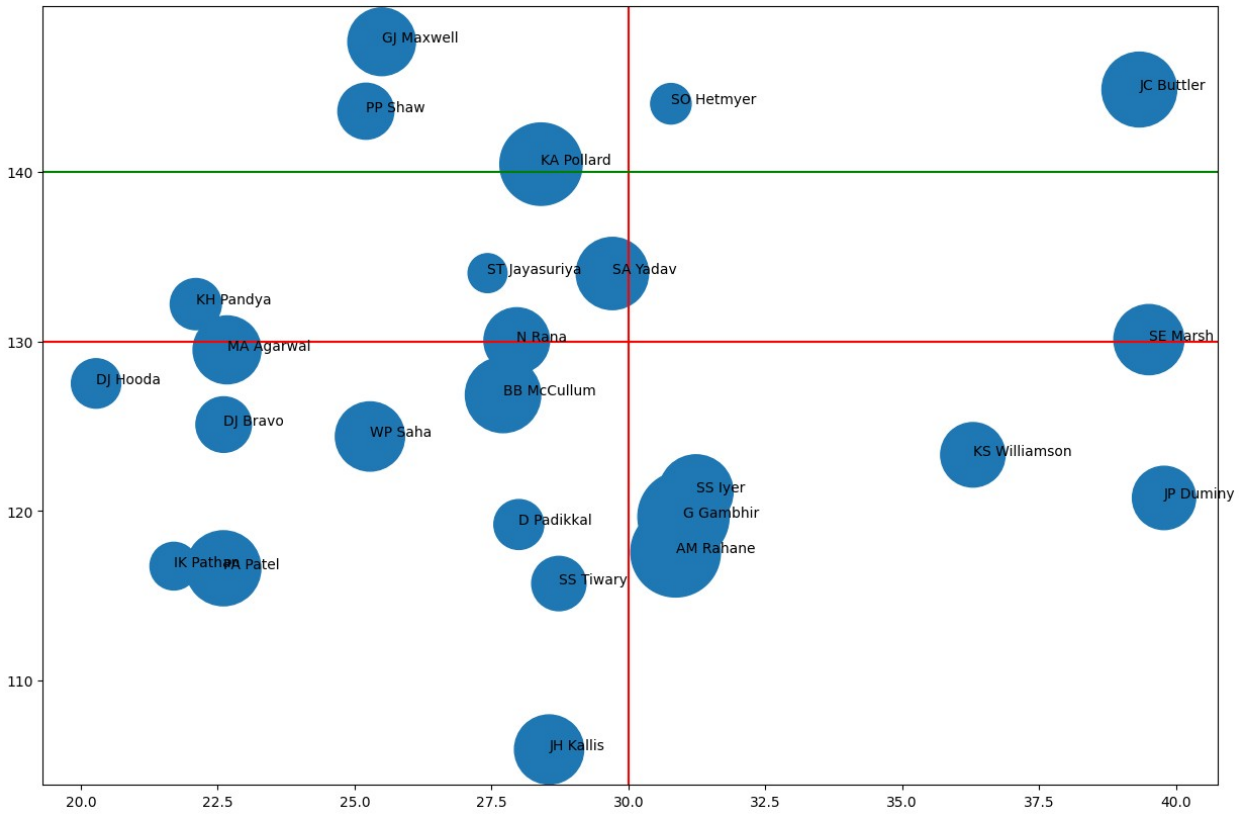
```
plt.scatter(x,y)
plt.text(1,5,'Point 1')
plt.text(2,6,'Point 2')
plt.text(3,7,'Point 3')
plt.text(4,8,'Point 4', fontdict={'size':12, 'color':'red'})
```

```
Text(4, 8, 'Point 4')
```



Horizontal and Vertical Lines

```
plt.figure(figsize=(15,10))
plt.scatter(sample_df['avg'], sample_df['strike_rate'],
s=sample_df['runs'])

plt.axhline(130, color='red')
plt.axvline(30, color='red')
plt.axhline(140, color='green')

for i in range(sample_df.shape[0]):

plt.text(sample_df['avg'].values[i],sample_df['strike_rate'].values[i]
,sample_df['batter'].values[i])
```



Subplots

- Used to plot multiple plots side by side.

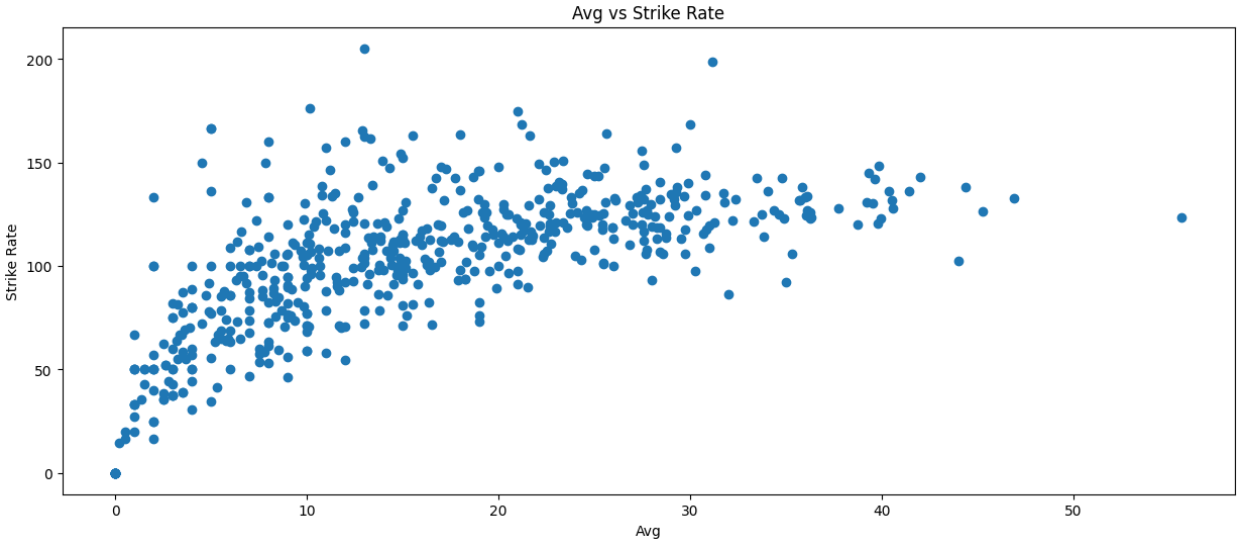
```
batters
```

	batter	runs	avg	strike_rate
0	V Kohli	6634	36.251366	125.977972
1	S Dhawan	6244	34.882682	122.840842
2	DA Warner	5883	41.429577	136.401577
3	RG Sharma	5881	30.314433	126.964594
4	SK Raina	5536	32.374269	132.535312
..	...	...	...	...
600	C Nanda	0	0.000000	0.000000
601	Akash Deep	0	0.000000	0.000000
602	S Ladda	0	0.000000	0.000000
603	V Pratap Singh	0	0.000000	0.000000
604	S Lamichhane	0	0.000000	0.000000

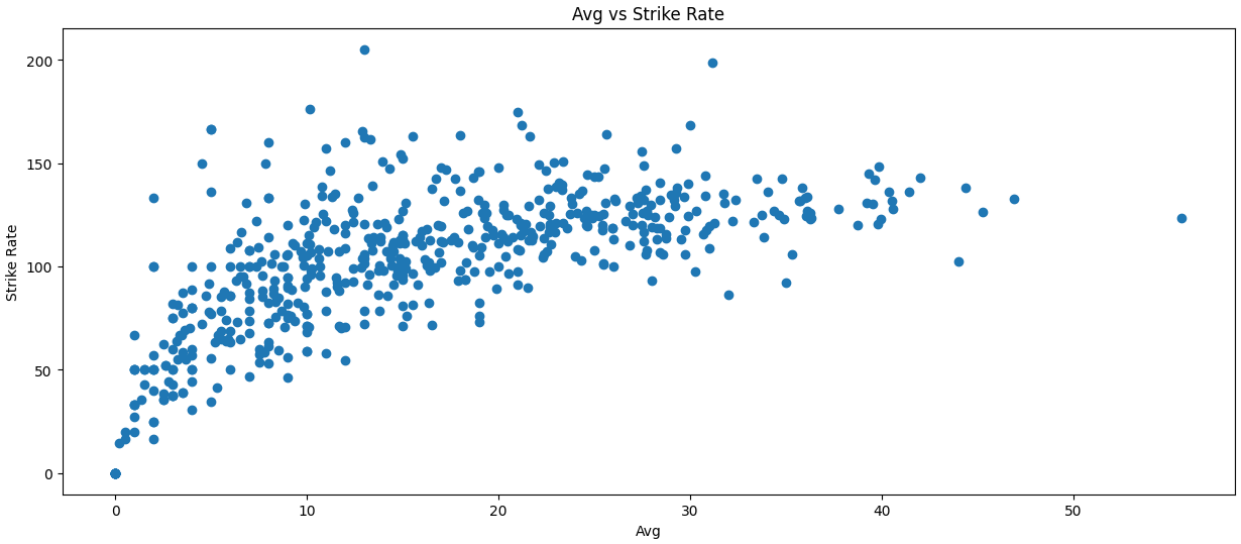
[605 rows x 4 columns]

```
plt.figure(figsize=(15,6))
plt.scatter(batters['avg'], batters['strike_rate'])
plt.title('Avg vs Strike Rate')
plt.xlabel('Avg')
plt.ylabel('Strike Rate')
```

```
plt.show()
```

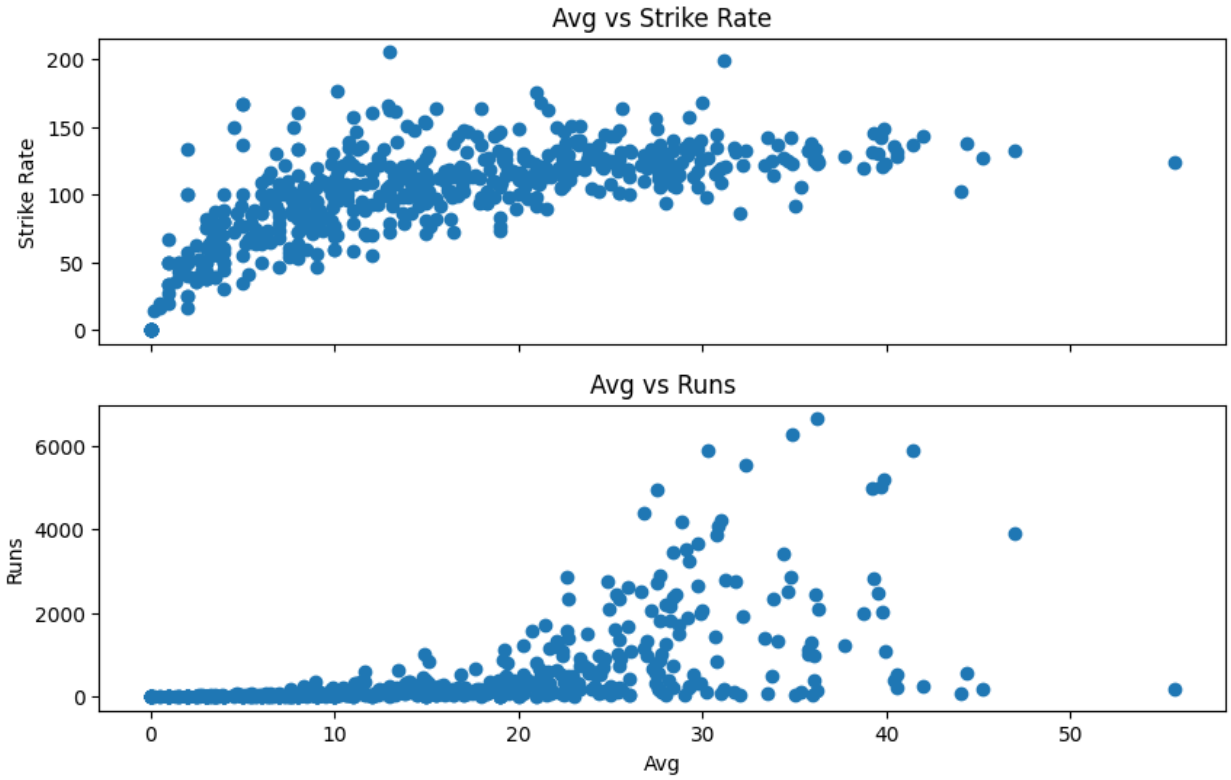



```
fig, ax = plt.subplots(figsize=(15,6))

ax.scatter(batters['avg'], batters['strike_rate'])
ax.set_title('Avg vs Strike Rate')
ax.set_xlabel('Avg')
ax.set_ylabel('Strike Rate')


Text(0, 0.5, 'Strike Rate')
```

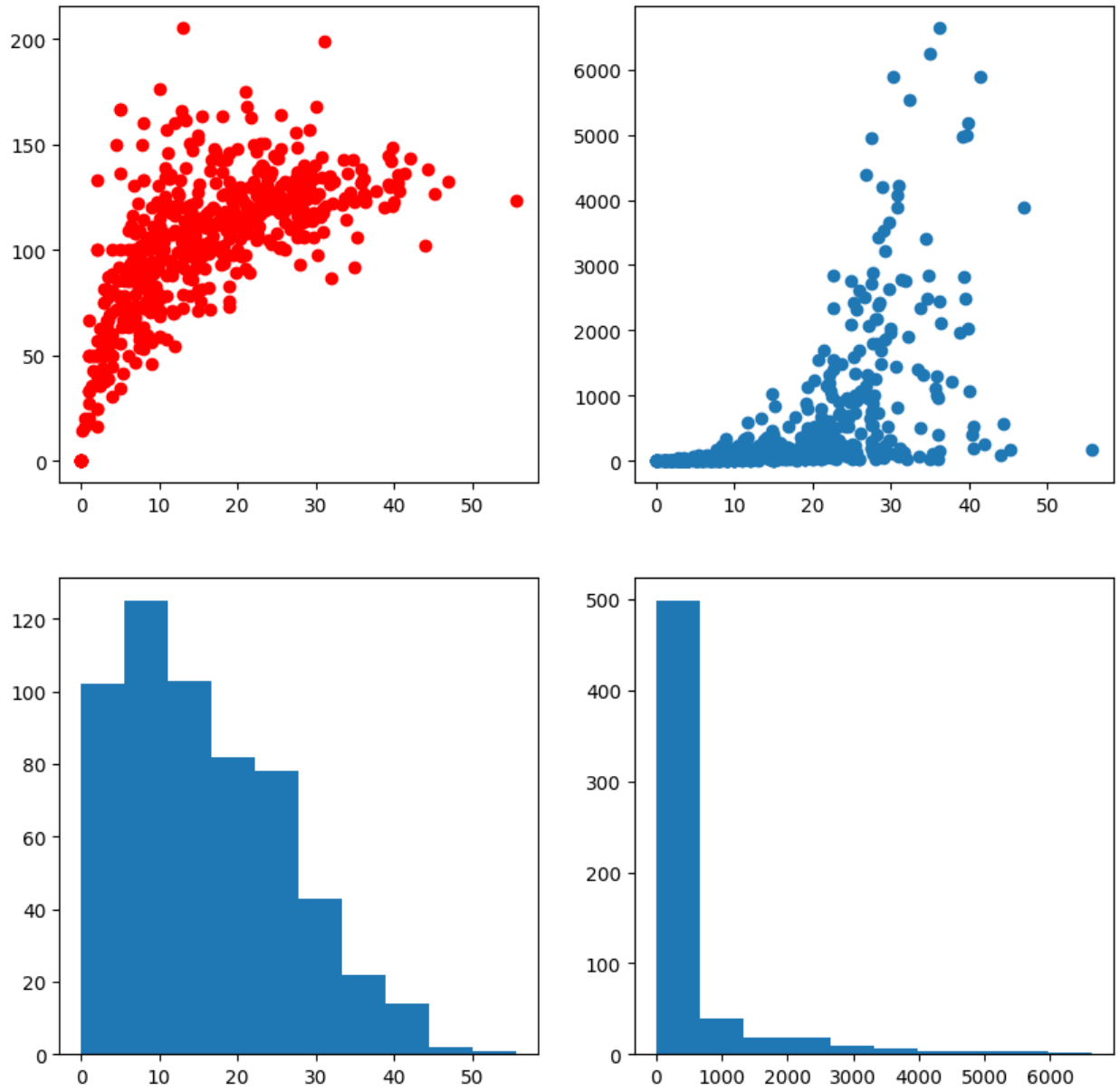



```
fig, ax = plt.subplots(nrows=2, ncols=1, figsize=(10,6), sharex=True)
```

```
ax[0].scatter(batters['avg'], batters['strike_rate'])
ax[1].scatter(batters['avg'], batters['runs'])

ax[0].set_title('Avg vs Strike Rate')
ax[0].set_ylabel('Strike Rate')

ax[1].set_title('Avg vs Runs')
ax[1].set_ylabel('Runs')
ax[1].set_xlabel('Avg')

Text(0.5, 0, 'Avg')
```



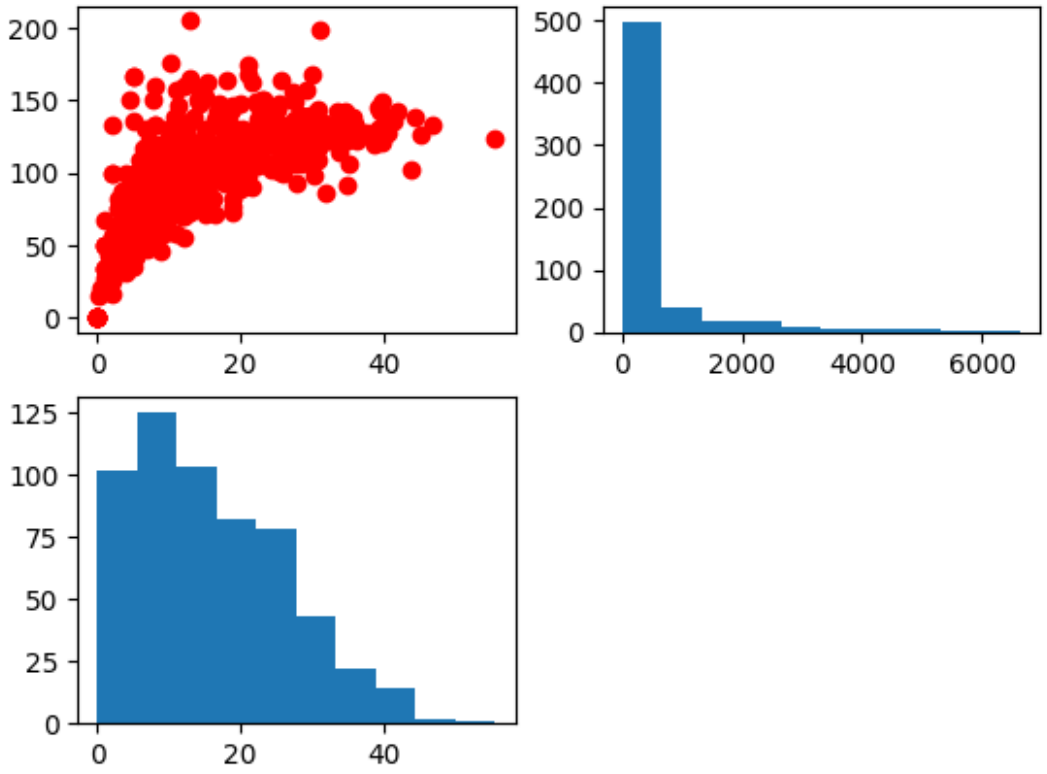

```
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(10,10))

ax[0,0].scatter(batters['avg'],batters['strike_rate'],color='red')
ax[0,1].scatter(batters['avg'],batters['runs'])
ax[1,0].hist(batters['avg'])
ax[1,1].hist(batters['runs'])

(array([499.,  40.,  19.,  19.,   9.,   6.,   4.,   4.,   3.,   2.]),
 array([   0. ,  663.4, 1326.8, 1990.2, 2653.6, 3317. , 3980.4,
4643.8,
        5307.2, 5970.6, 6634. ]),
 <BarContainer object of 10 artists>)
```



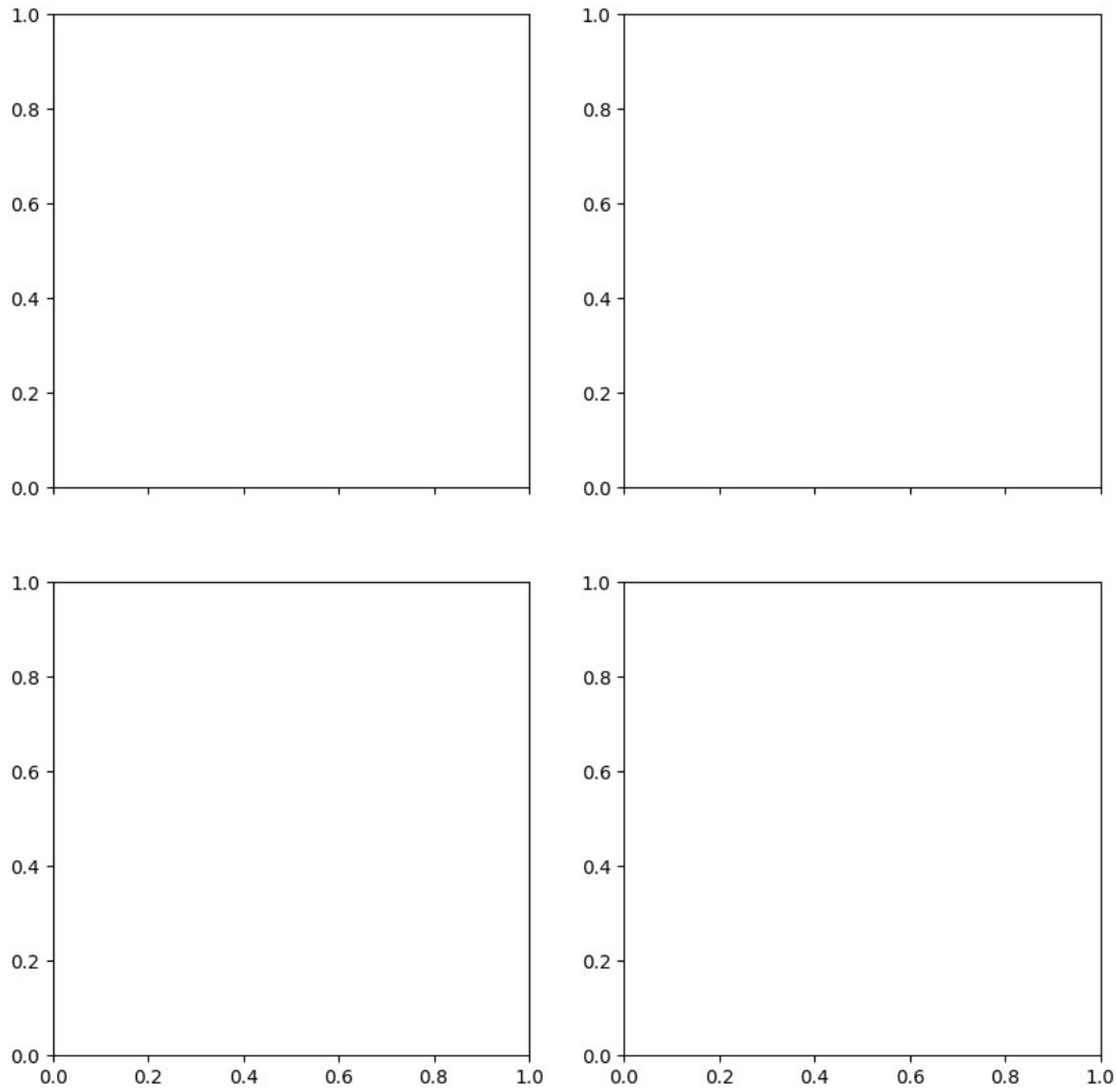

```
fig = plt.figure()

ax1 = fig.add_subplot(2,2,1)
ax1.scatter(batters['avg'], batters['strike_rate'], color='red')

ax2 = fig.add_subplot(2,2,2)
ax2.hist(batters['runs'])

ax3 = fig.add_subplot(2,2,3)
ax3.hist(batters['avg'])
```

```
(array([102., 125., 103.,  82.,  78.,  43.,  22.,  14.,   2.,   1.]),
 array([ 0.        ,  5.56666667, 11.13333333, 16.7       ,
```

```
22.26666667,
        27.83333333, 33.4        , 38.96666667, 44.53333333, 50.1
,
        55.66666667]),
 <BarContainer object of 10 artists>)
```



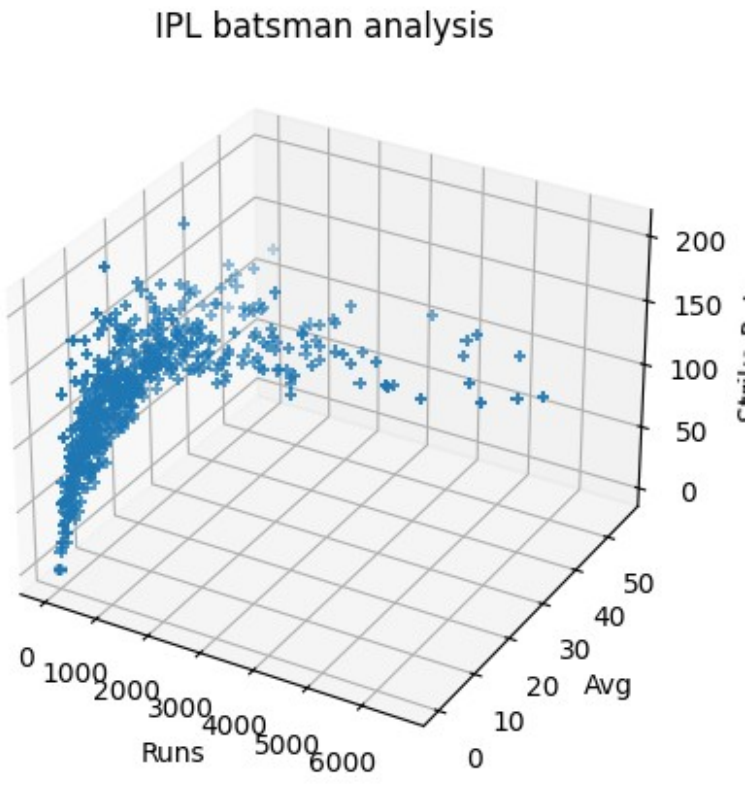

```
fig, ax = plt.subplots(nrows=2, ncols=2, sharex=True, figsize=(10,
10))
ax[1,1]

<Axes: >
```



3D Scatter Plots

```
fig = plt.figure()

ax = plt.subplot(projection='3d')

ax.scatter3D(batters['runs'], batters['avg'], batters['strike_rate'],
marker='+')

ax.set_title('IPL batsman analysis')
ax.set_xlabel('Runs')
ax.set_ylabel('Avg')
ax.set_zlabel('Strike Rate')
```

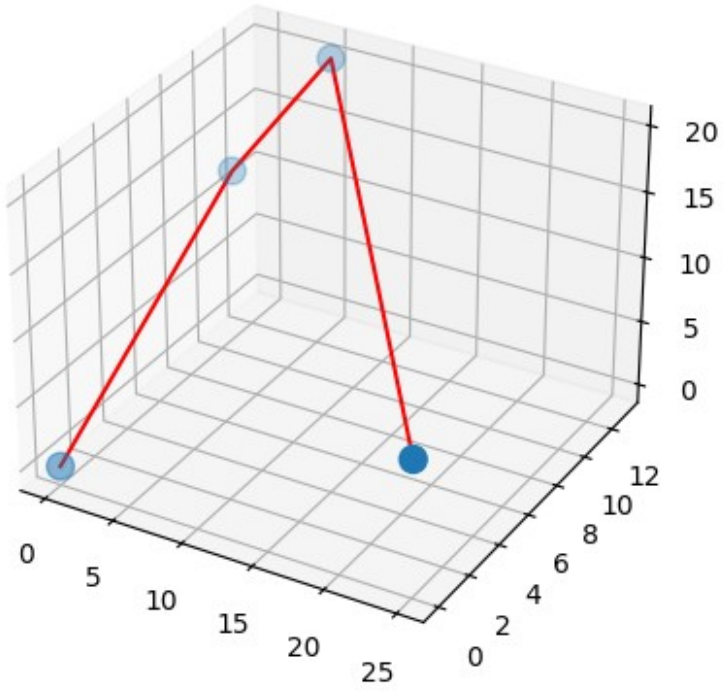
```
Text(0.5, 0, 'Strike Rate')
```



3D Line Plot

```
x = [0,1,5,25]
y = [0,10,13,0]
z = [0,13,20,9]

fig = plt.figure()

ax = plt.subplot(projection='3d')

ax.scatter3D(x, y, z, s=[100,100,100,100])
ax.plot3D(x,y,z,color='red')
```

```
[<mpl_toolkits.mplot3d.art3d.Line3D at 0x1ab059a3290>]
```



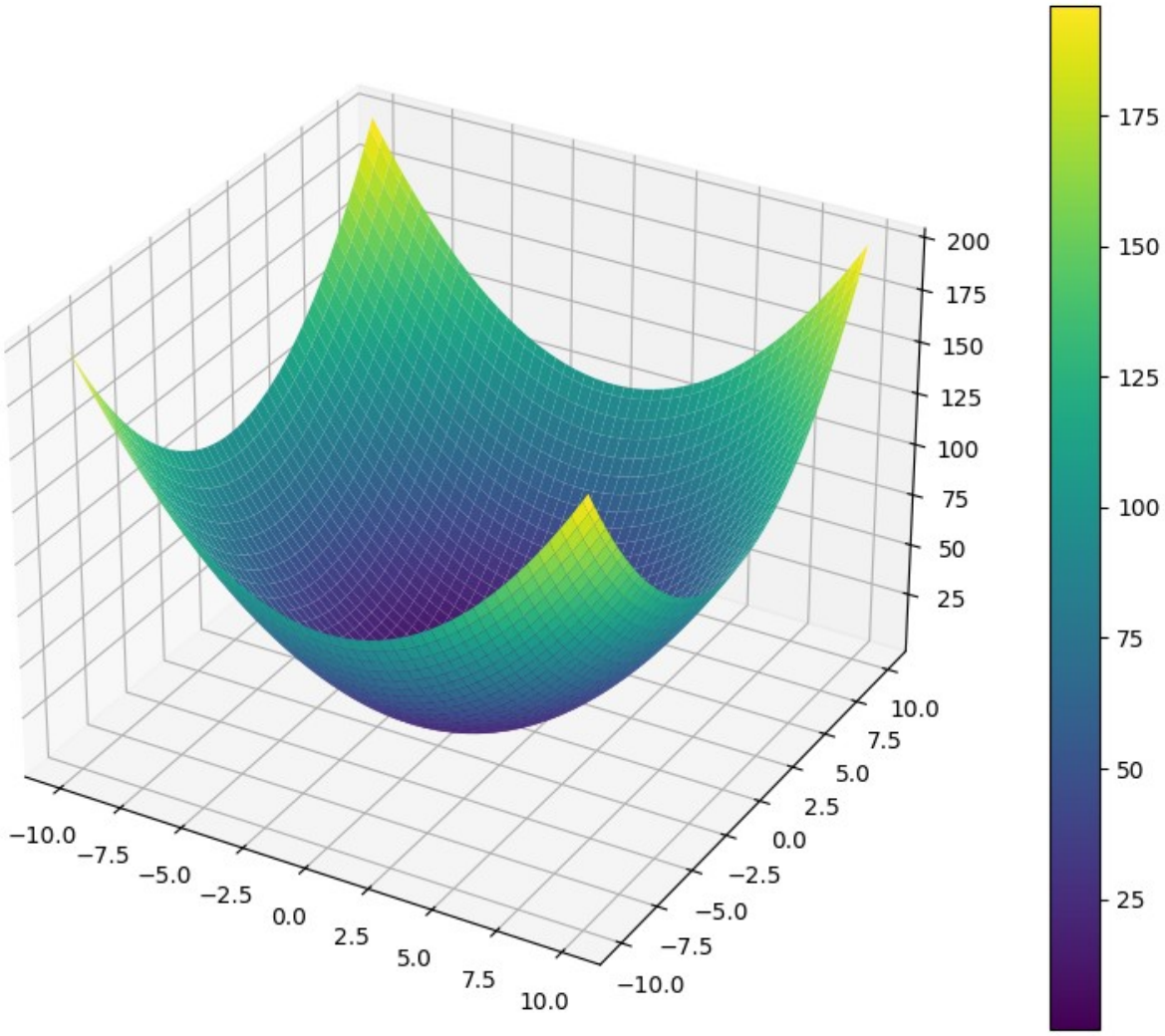
3D Surface Plots

```
x = np.linspace(-10, 10, 100)
y = np.linspace(-10, 10, 100)

xx, yy = np.meshgrid(x, y)

z = xx**2 + yy**2
z.shape
```

```
(100, 100)
```

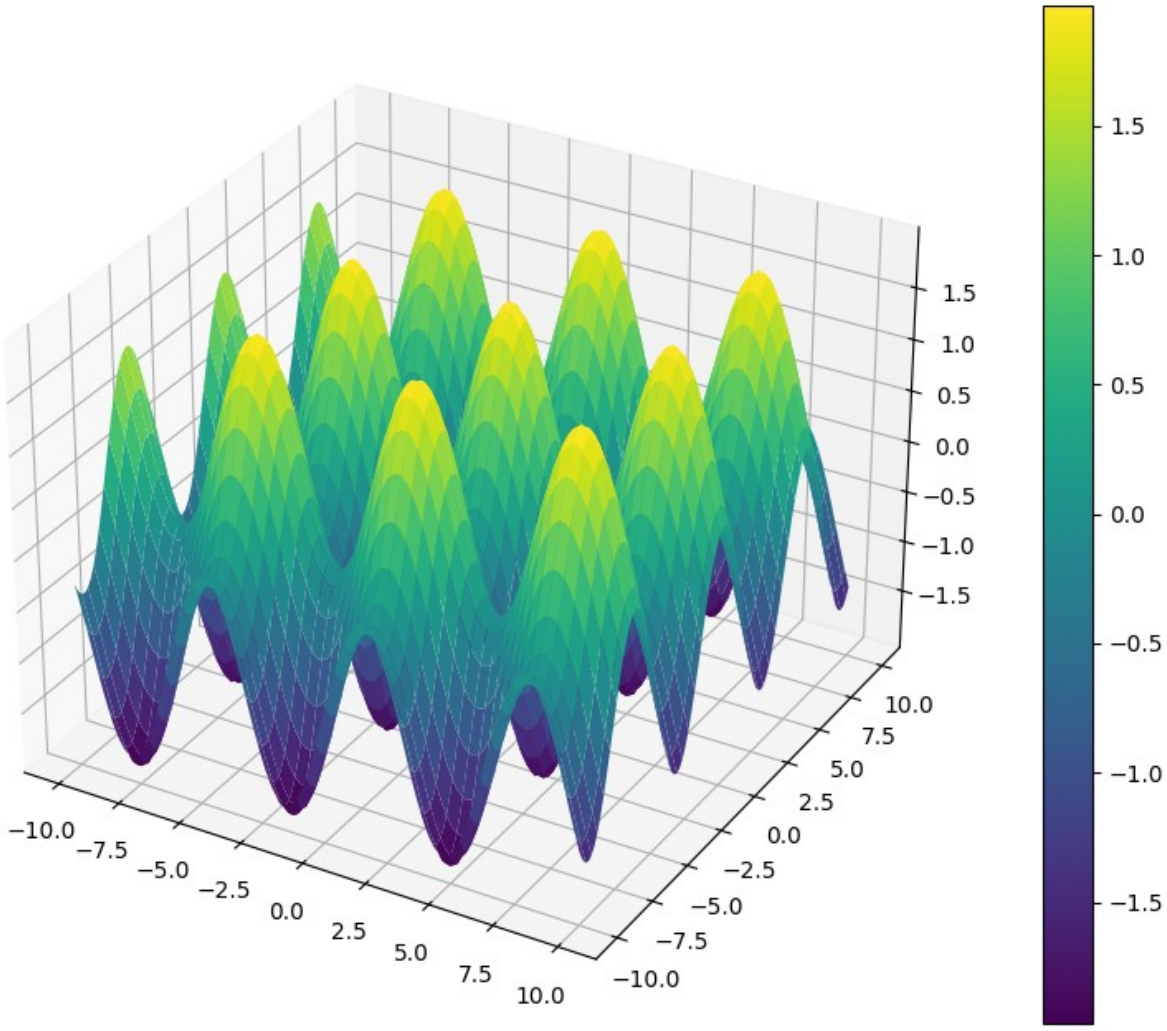
```
fig = plt.figure(figsize=(12,8))

ax = plt.subplot(projection='3d')

p = ax.plot_surface(xx, yy, z, cmap='viridis')

fig.colorbar(p)
```

```
<matplotlib.colorbar.Colorbar at 0x1ab055c0cb0>
```




```
z = np.sin(xx) + np.cos(yy)

fig = plt.figure(figsize=(12,8))

ax = plt.subplot(projection='3d')

p = ax.plot_surface(xx, yy, z, cmap='viridis')
fig.colorbar(p)
```

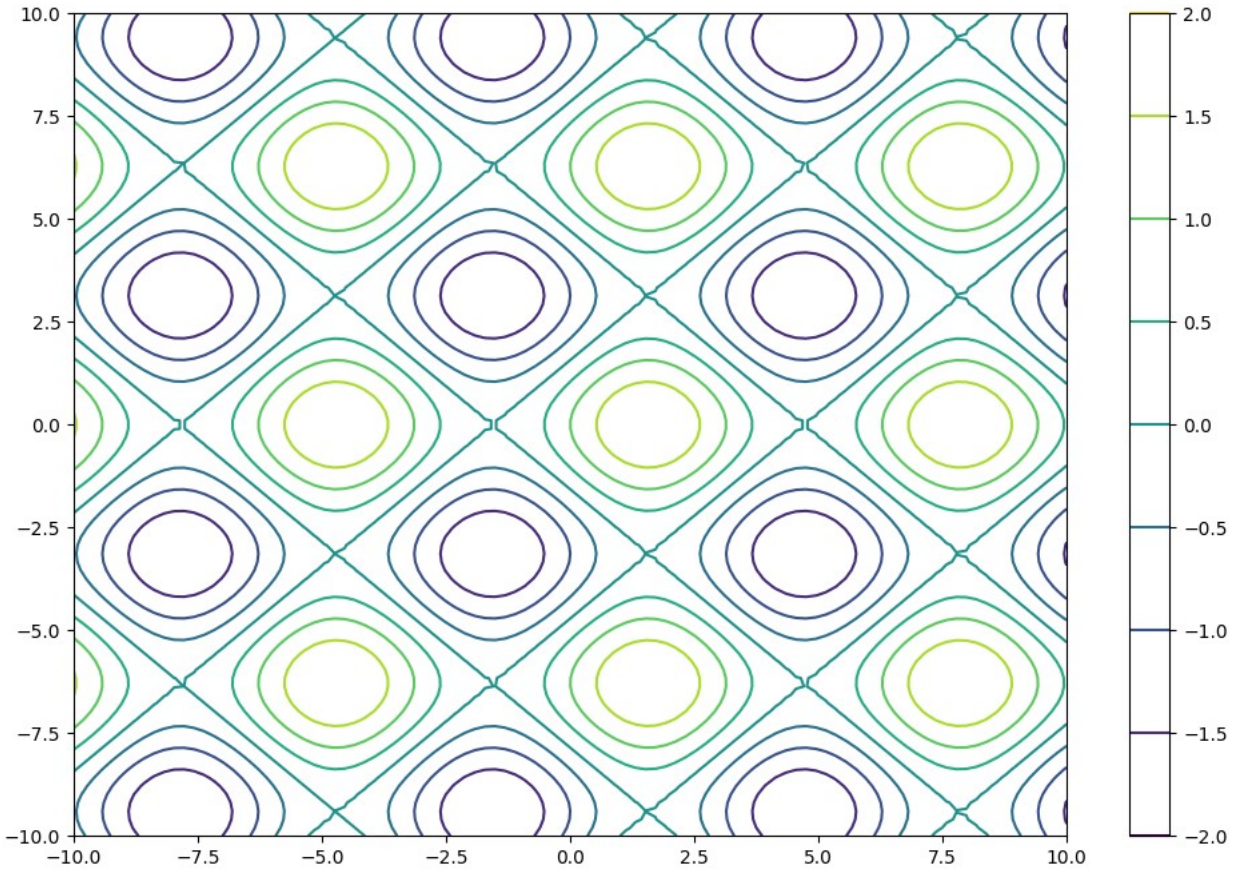
```
<matplotlib.colorbar.Colorbar at 0x1ab059a33b0>
```



Contour Plots

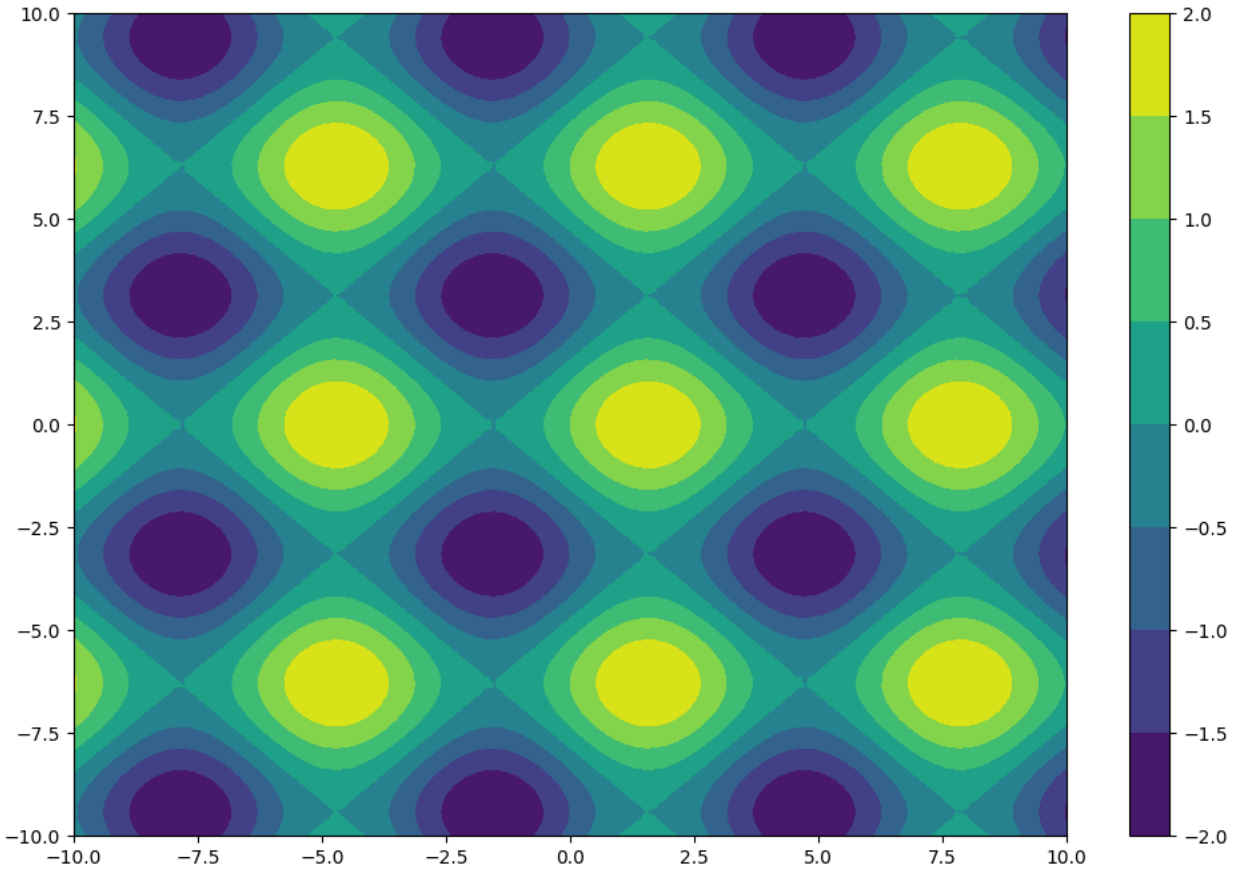
```
fig = plt.figure(figsize=(12, 8))

ax = plt.subplot()

p = ax.contour(xx, yy, z, cmap='viridis')

fig.colorbar(p)
```

```
<matplotlib.colorbar.Colorbar at 0x1ab0681a870>
```




```
fig = plt.figure(figsize=(12, 8))

ax = plt.subplot()

p = ax.contourf(xx, yy, z, cmap='viridis')

fig.colorbar(p)
```

```
<matplotlib.colorbar.Colorbar at 0x1ab070f1d60>
```



Heatmap

```
delivery = pd.read_csv('Datasets/IPL_Ball_by_Ball_2008_2022.csv')
delivery.head()
```

```
        ID  innings  overs  ballnumber       batter          bowler  \
0  1312200        1      0           1  YBK Jaiswal  Mohammed Shami
1  1312200        1      0           2  YBK Jaiswal  Mohammed Shami
2  1312200        1      0           3   JC Buttler  Mohammed Shami
3  1312200        1      0           4  YBK Jaiswal  Mohammed Shami
4  1312200        1      0           5  YBK Jaiswal  Mohammed Shami

   non-striker extra_type  batsman_run  extras_run  total_run
non_boundary  \
0   JC Buttler        NaN            0           0          0
0
1   JC Buttler    legbyes            0           1          1
0
2  YBK Jaiswal        NaN            1           0          1
0
3   JC Buttler        NaN            0           0          0
0
4   JC Buttler        NaN            0           0          0
```

```
0

   isWicketDelivery player_out kind fielders_involved
BattingTeam
0                 0        NaN  NaN               NaN  Rajasthan
Royals
1                 0        NaN  NaN               NaN  Rajasthan
Royals
2                 0        NaN  NaN               NaN  Rajasthan
Royals
3                 0        NaN  NaN               NaN  Rajasthan
Royals
4                 0        NaN  NaN               NaN  Rajasthan
Royals
```

```
temp = delivery[(delivery['ballnumber'].isin([1,2,3,4,5,6]) ) &
(delivery['batsman_run']==6)]

temp.head()
```

```
          ID  innings  overs  ballnumber       batter
bowler  \
16   1312200        1      2           5  YBK Jaiswal     Mohammed
Shami
22   1312200        1      3           5  YBK Jaiswal         Yash
Dayal
103  1312200        1     17           2     TA Boult      R Sai
Kishore
107  1312200        1     17           6     OC McCoy      R Sai
Kishore
142  1312200        2      3           5      MS Wade  M Prasidh
Krishna

      non-striker extra_type  batsman_run  extras_run  total_run  \
16     JC Buttler        NaN            6           0          6
22     JC Buttler        NaN            6           0          6
103       R Parag        NaN            6           0          6
107       R Parag        NaN            6           0          6
142  Shubman Gill        NaN            6           0          6

     non_boundary  isWicketDelivery player_out kind fielders_involved
\
16              0                 0        NaN  NaN               NaN

22              0                 0        NaN  NaN               NaN

103             0                 0        NaN  NaN               NaN

107             0                 0        NaN  NaN               NaN
```

```
142             0                   0        NaN  NaN                 NaN
```

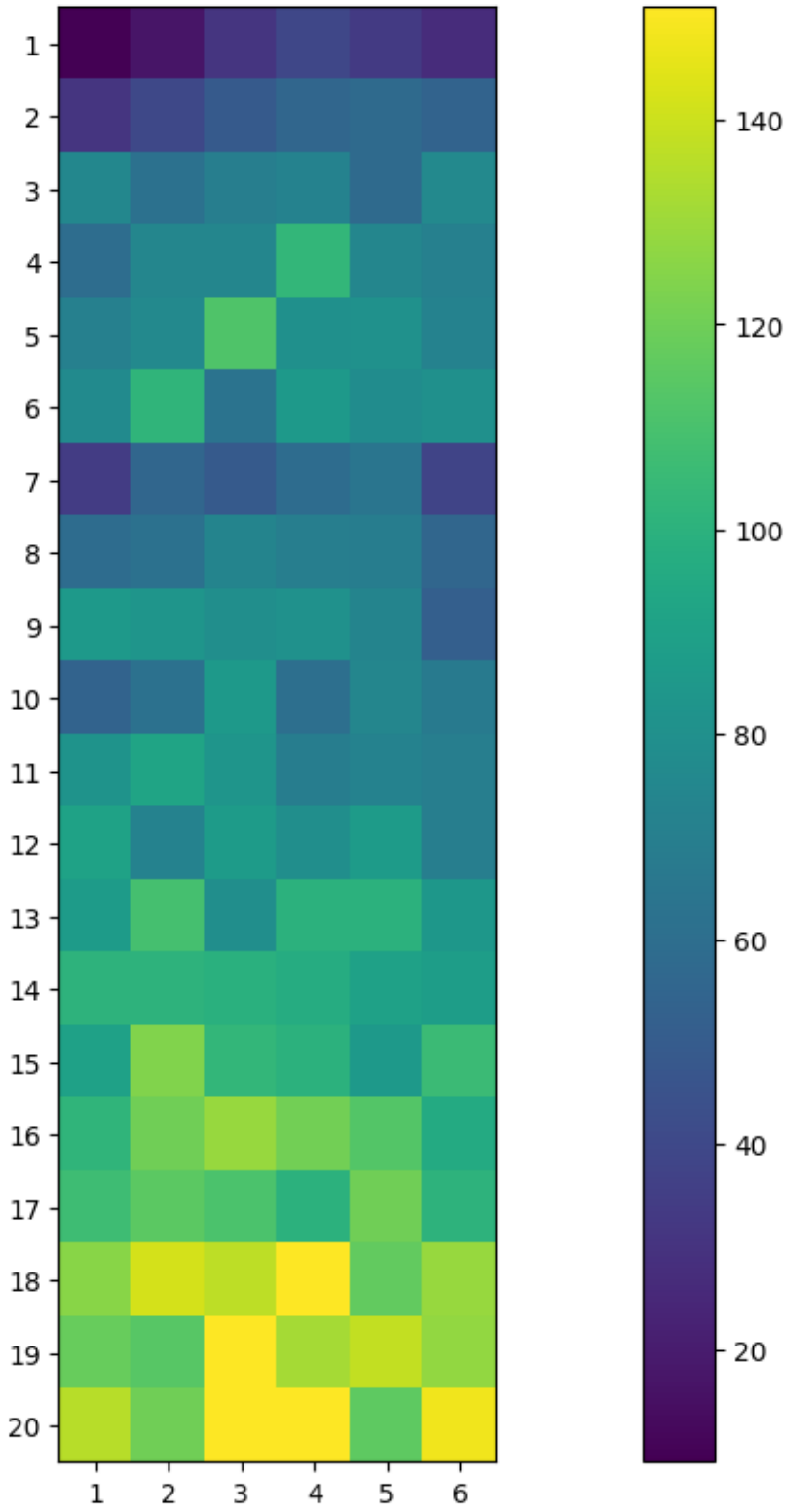
```
          BattingTeam
16   Rajasthan Royals
22   Rajasthan Royals
103  Rajasthan Royals
107  Rajasthan Royals
142    Gujarat Titans
```

```
grid = temp.pivot_table(index='overs', columns='ballnumber',
values='batsman_run', aggfunc='count')

grid.head(5)
```

ballnumber	1	2	3	4	5	6
overs						
0	9	17	31	39	33	27
1	31	40	49	56	58	54
2	75	62	70	72	58	76
3	60	74	74	103	74	71
4	71	76	112	80	81	72

```
plt.figure(figsize=(20,10))
plt.xticks(np.arange(0,6), list(range(1,7)))
plt.yticks(delivery['overs'].unique(), list(range(1,21)))
plt.imshow(grid)
plt.colorbar()
```

```
<matplotlib.colorbar.Colorbar at 0x1ab07356300>
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
1
2
3
4
5
6
140
120
100
80
60
40
20

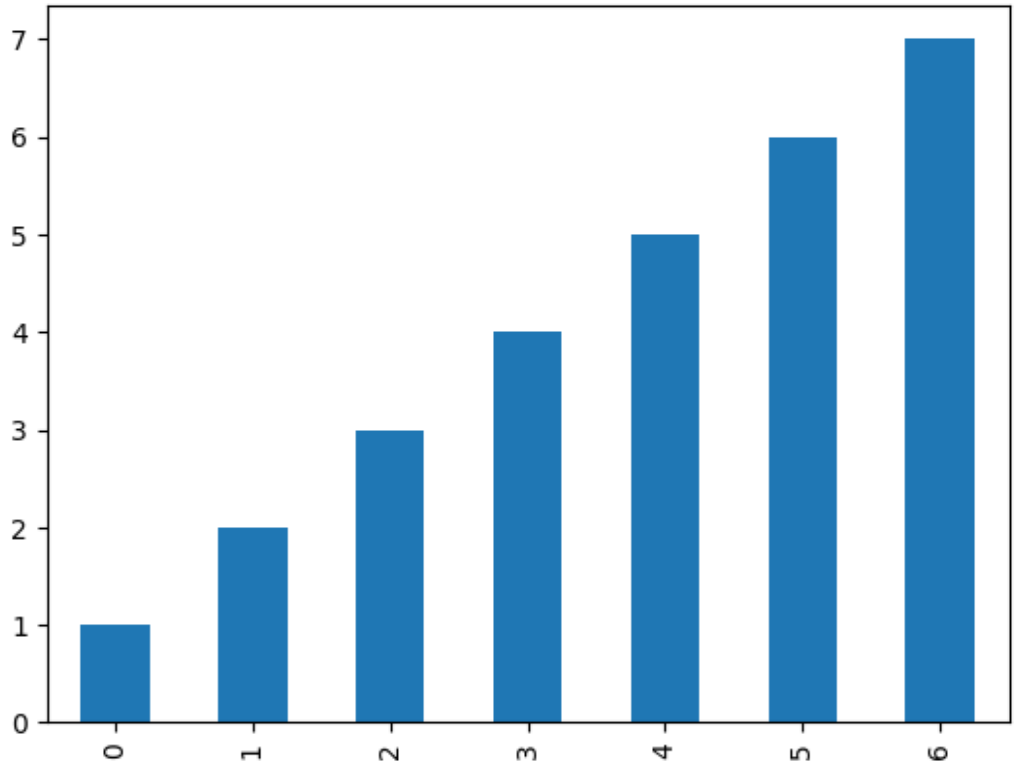
Pandas Plot()

kind =	Plot type
line	Line plot (default)
bar	Vertical bar plot
barh	Horizontal bar plot
hist	Histogram
box	Boxplot
area	Area plot
pie	Pie plot
scatter	Scatter plot

Series

```
s = pd.Series([1,2,3,4,5,6,7])
s.plot(kind='bar')
```

```
<Axes: >
```



DataFrame

```
tips = sns.load_dataset('tips')
tips.head(5)
```

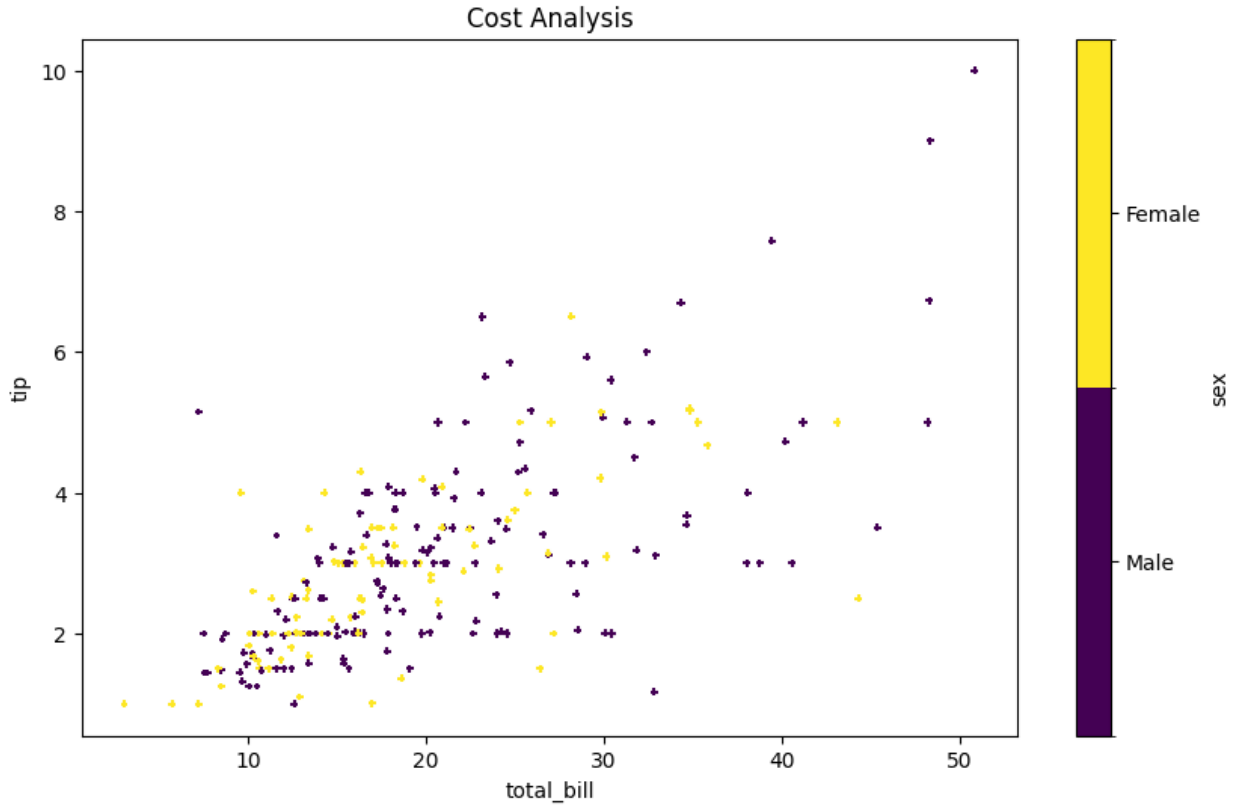
```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
```

```
tips['size'] += 10
```

Scatter Plot > labels > markers > figsize > color > cmap

```
tips.plot(kind='scatter', x='total_bill', y='tip', title='Cost
Analysis', marker='+', figsize=(10,6), s='size', c='sex',
cmap='viridis')
```

```
<Axes: title={'center': 'Cost Analysis'}, xlabel='total_bill',
ylabel='tip'>
```




```
stocks =
pd.read_csv('https://raw.githubusercontent.com/m-mehdi/pandas_tutorial
s/main/weekly_stocks.csv')
stocks.head()
```

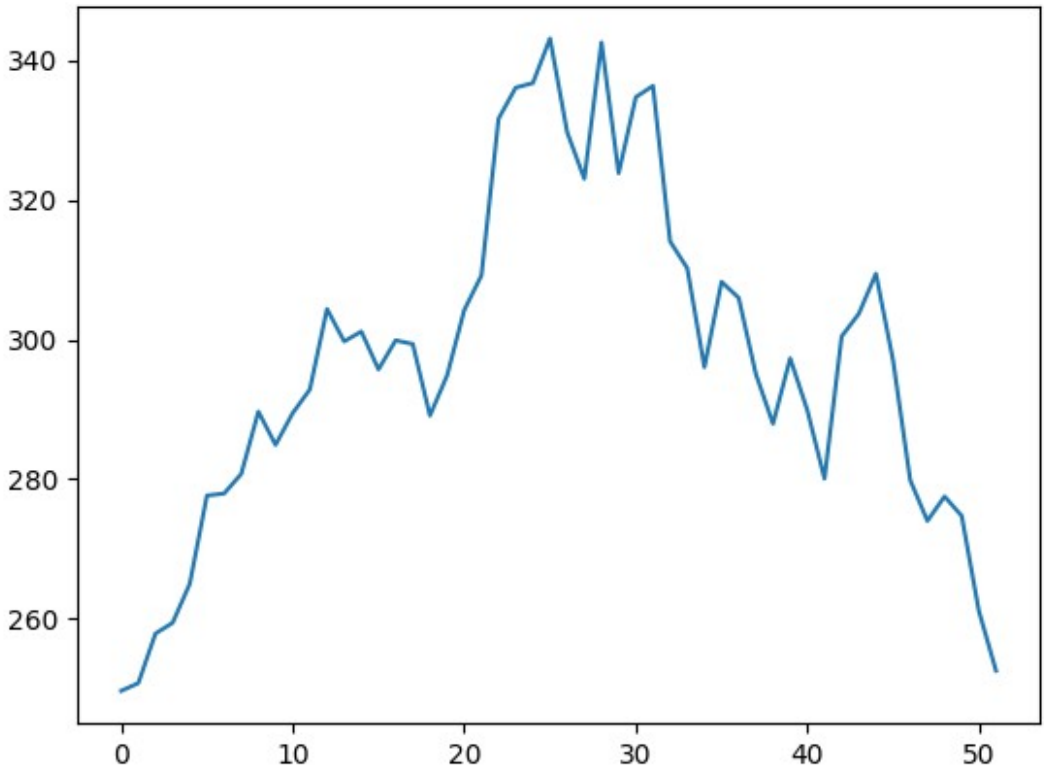
```
         Date        MSFT          FB        AAPL
0  2021-05-24  249.679993  328.730011  124.610001
```

```
1  2021-05-31  250.789993  330.350006  125.889999
2  2021-06-07  257.890015  331.260010  127.349998
3  2021-06-14  259.429993  329.660004  130.460007
4  2021-06-21  265.019989  341.369995  133.110001
```

Single line plot

```
stocks['MSFT'].plot()
```

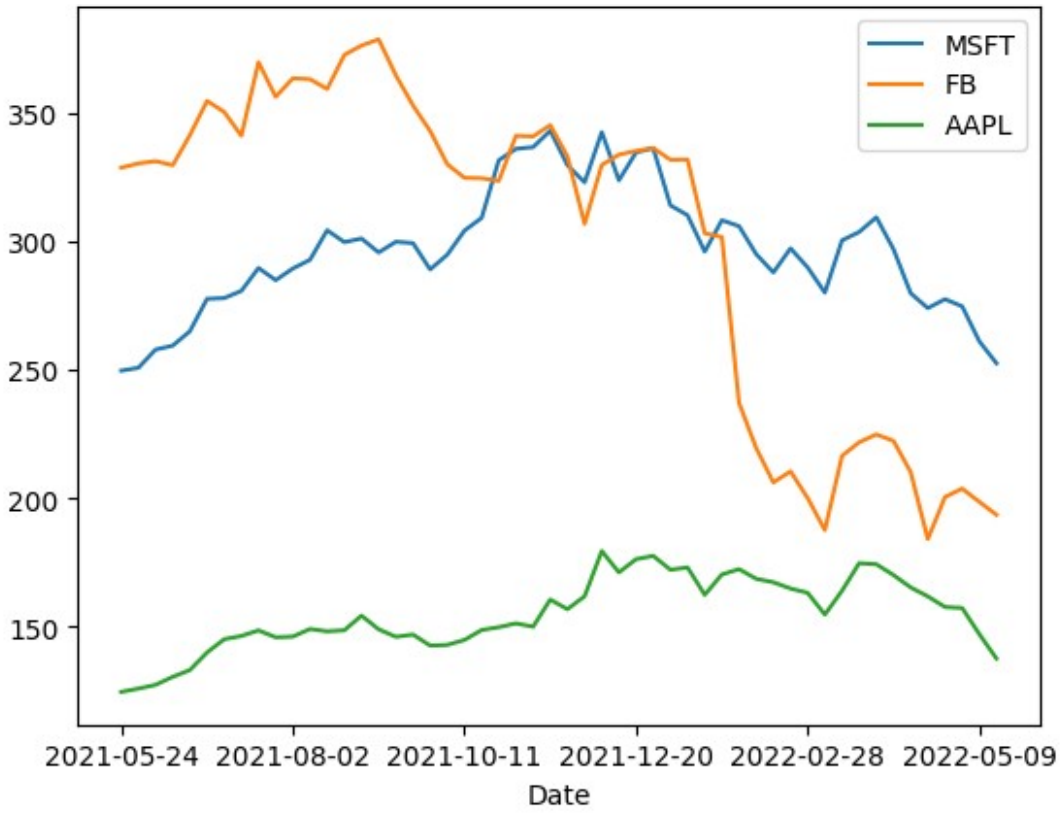
```
<Axes: >
```



All cols line plot with x axis as date

```
stocks.plot(kind='line', x='Date')
```

```
<Axes: xlabel='Date'>
```



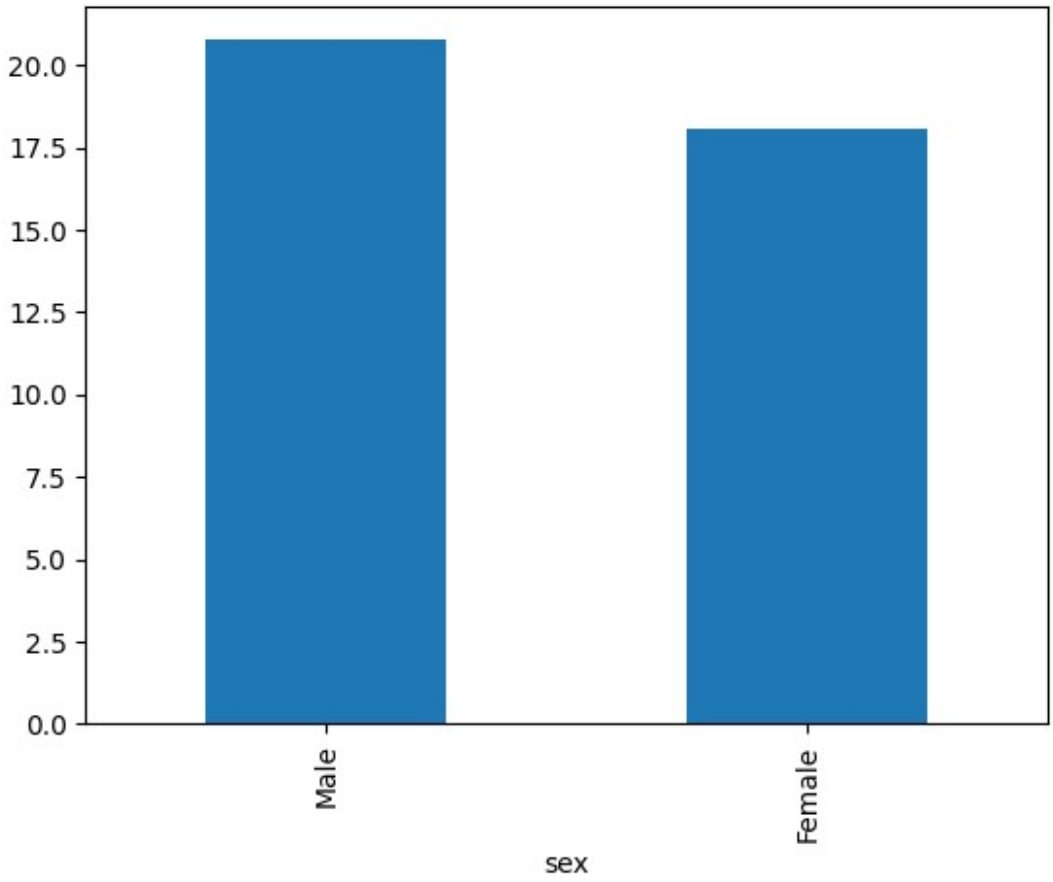

```
stocks[['Date', 'AAPL', 'FB']].plot(kind='line', x='Date')
```

```
<Axes: xlabel='Date'>
```

Ploting on groupby DF

```
tips.groupby('sex', observed=False)
['total_bill'].mean().plot(kind='bar')
```

```
<Axes: xlabel='sex'>
```



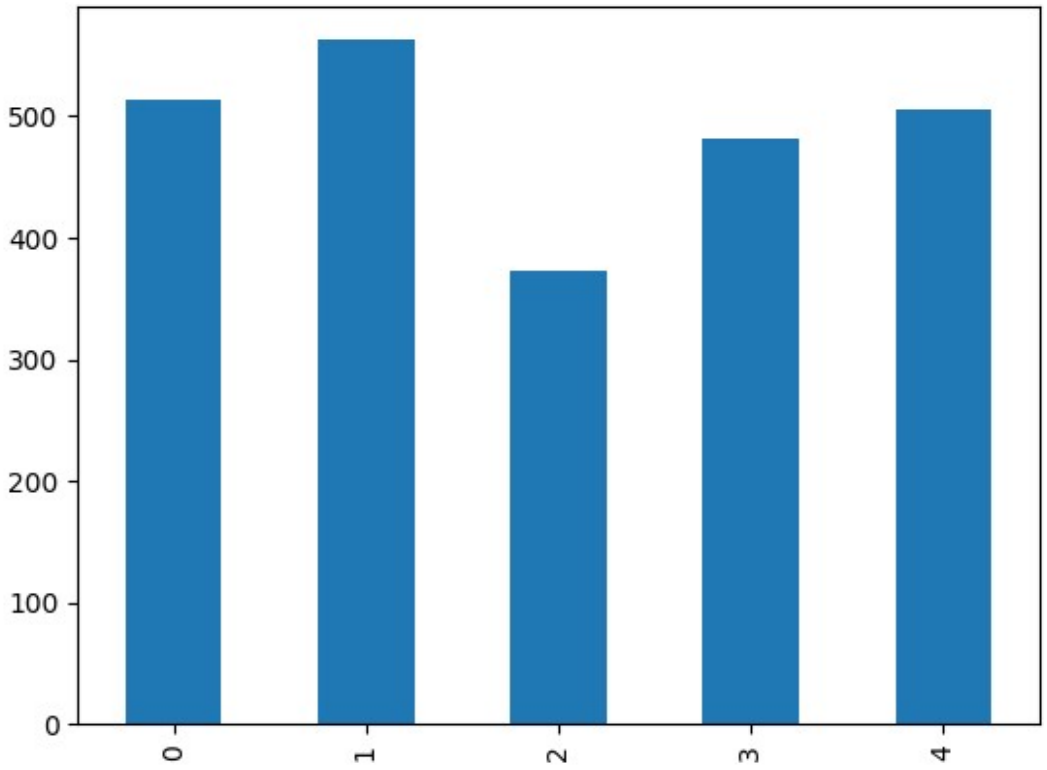

```
temp = pd.read_csv('Datasets/batsman_season_record.csv')
temp.head()
```

	batsman	2015	2016	2017
0	AB de Villiers	513	687	216
1	DA Warner	562	848	641
2	MS Dhoni	372	284	290
3	RG Sharma	482	489	333
4	V Kohli	505	973	308

Ploting on single column

```
temp['2015'].plot(kind='bar')
```

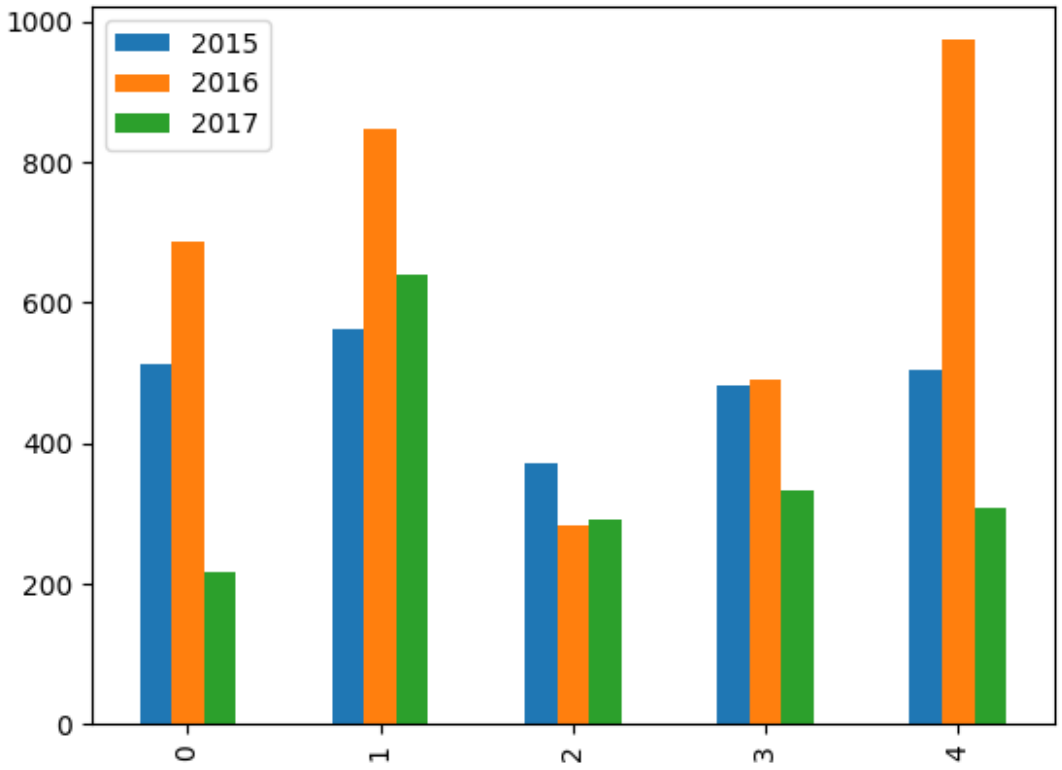
```
<Axes: >
```



Side by side multiple bar graphs

```
temp.plot(kind='bar')
```

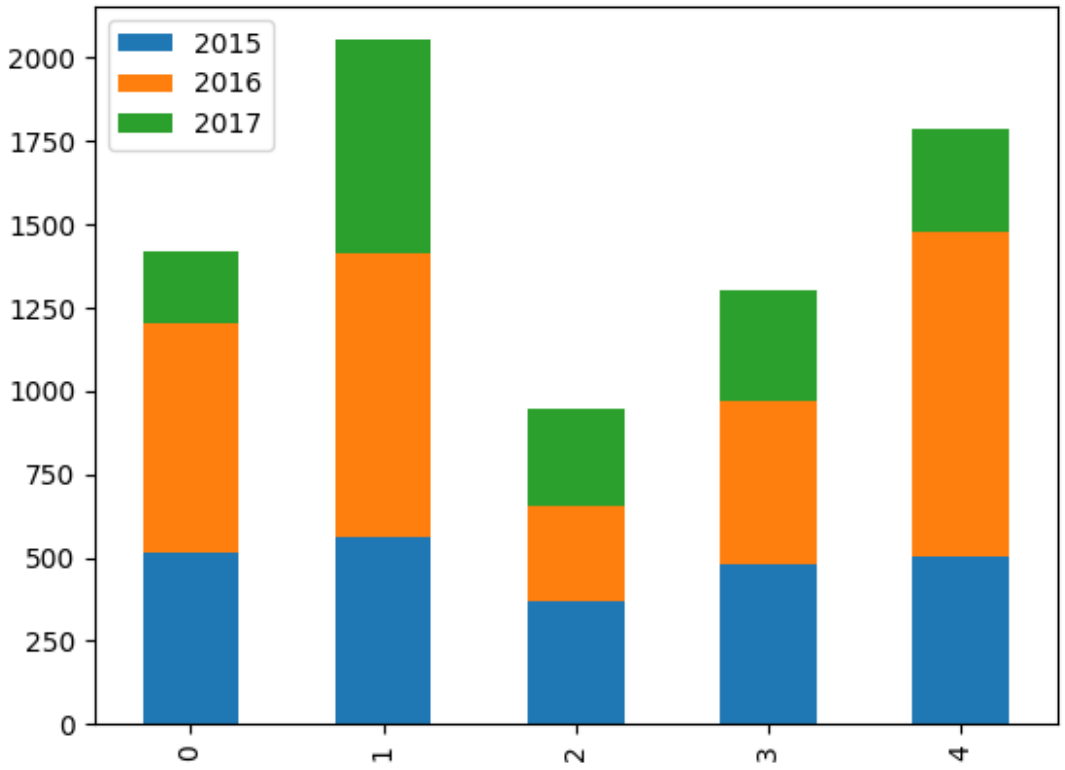
```
<Axes: >
```



Stacked bar chart

```
temp.plot(kind='bar', stacked=True)
```

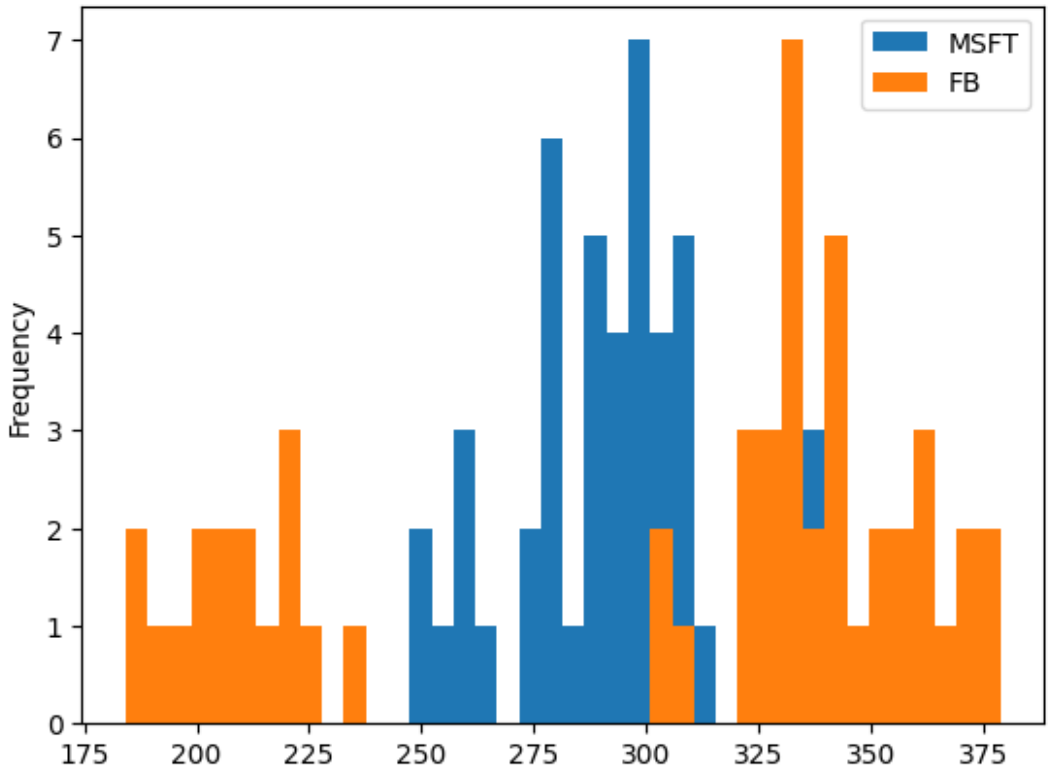
```
<Axes: >
```



Histograms

```
stocks[['MSFT', 'FB']].plot(kind='hist', bins=40)
```

```
<Axes: ylabel='Frequency'>
```



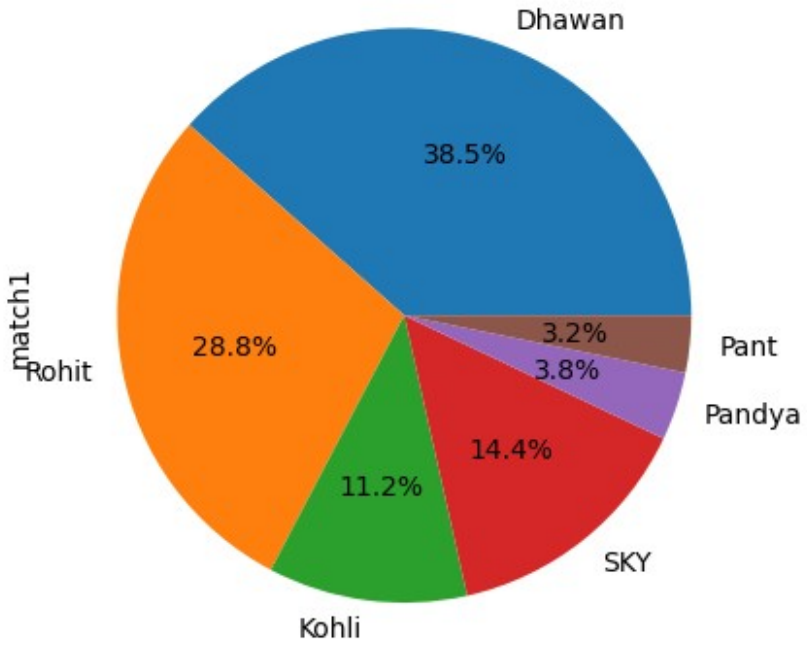

```
df = pd.DataFrame(
    {
        'batsman':['Dhawan','Rohit','Kohli','SKY','Pandya','Pant'],
        'match1':[120,90,35,45,12,10],
        'match2':[0,1,123,130,34,45],
        'match3':[50,24,145,45,10,90]
    }
)

df.head()
```

	batsman	match1	match2	match3
0	Dhawan	120	0	50
1	Rohit	90	1	24
2	Kohli	35	123	145
3	SKY	45	130	45
4	Pandya	12	34	10

Single pie chart

```
df['match1'].plot(kind='pie', labels=df['batsman'].values,
autopct="%0.1f%%")
```

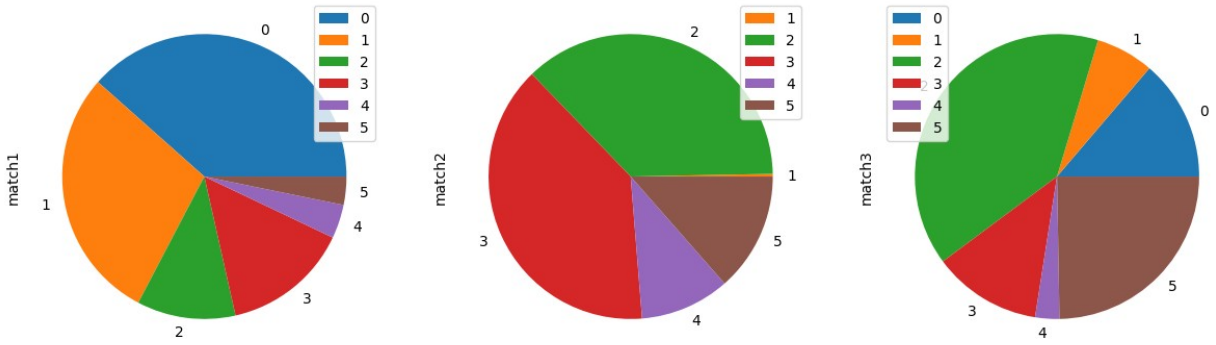
```
<Axes: ylabel='match1'>
```



Multiple pie charts

```
df[['match1', 'match2', 'match3']].plot(kind='pie', subplots=True,
figsize=(15,8))
```

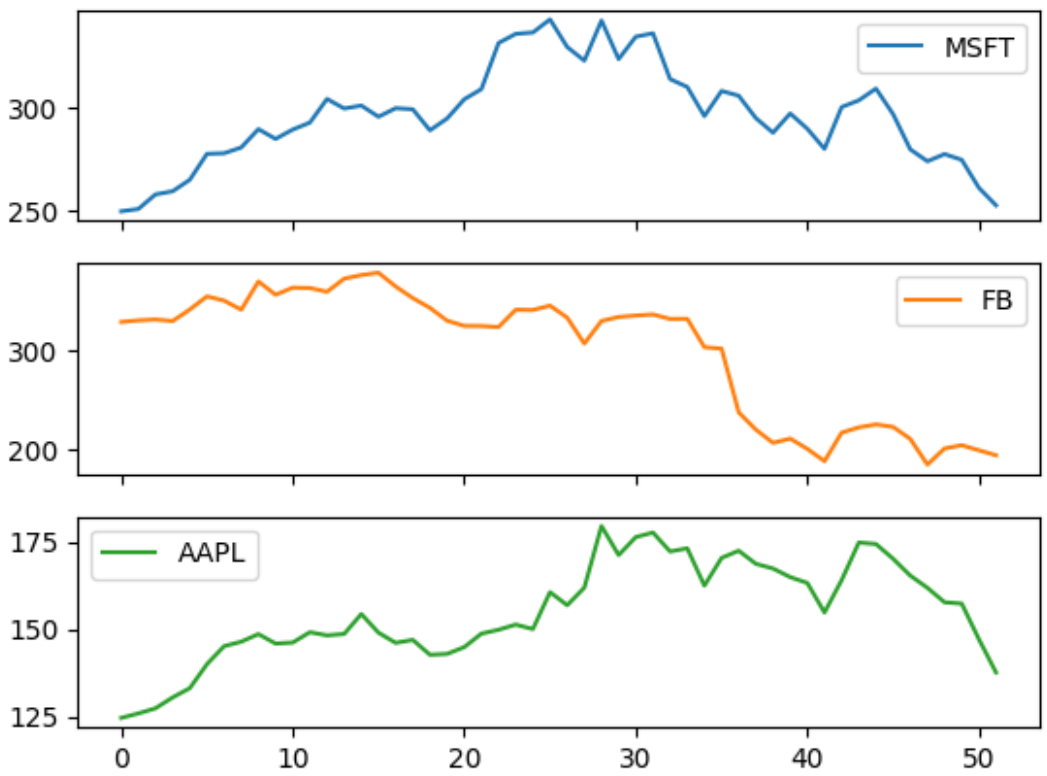
```
array([<Axes: ylabel='match1'>, <Axes: ylabel='match2'>,
       <Axes: ylabel='match3'>], dtype=object)
```



Multiple separate graphs together using inbuilt subplots parameter

```
stocks.plot(kind='line', subplots=True)
```

```
array([<Axes: >, <Axes: >, <Axes: >], dtype=object)
```




```
tips.pivot_table(index=['day', 'time'], columns=['sex', 'smoker'],
values='total_bill', aggfunc='mean', observed=False).plot(kind='pie',
subplots=True, figsize=(20,10))
```

```
array([<Axes: ylabel='(Male, Yes)'>, <Axes: ylabel='(Male, No)'>,
       <Axes: ylabel='(Female, Yes)'>, <Axes: ylabel='(Female, No)'>],
      dtype=object)
```