Python for Beginners

Archer Infotech , PUNE

Python - Matplotlib

What is Matplotlib ?

- **Matplotlib** is a low level graph plotting library in python that serves as a visualization utility.

- Matplotlib was created by **John D. Hunter.**

- Matplotlib is open source and we can use it freely.

- Matplotlib is mostly written in python, a few segments are written in C, Objective-C and Javascript for Platform compatibility.

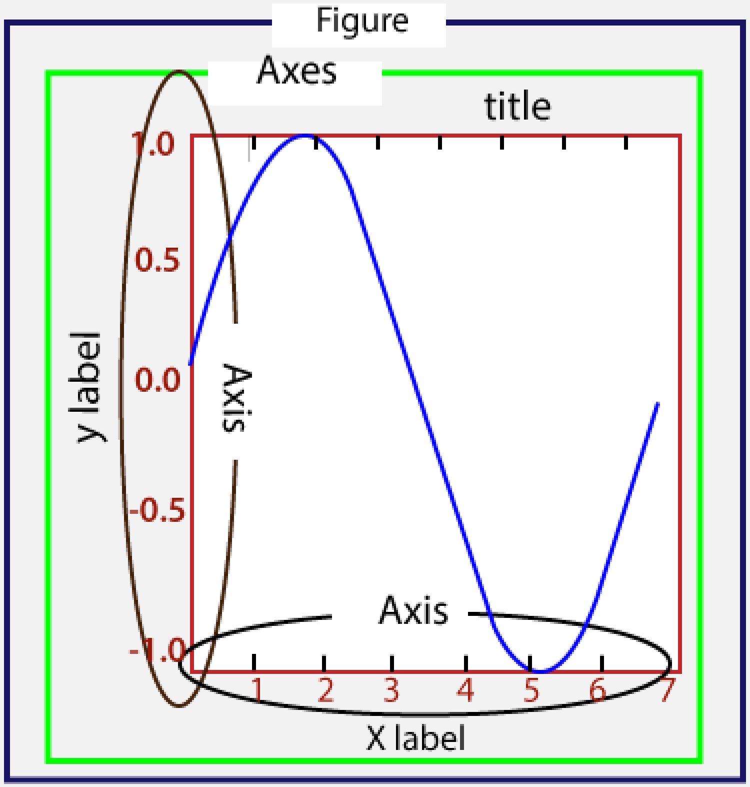
- **matplotlib.pyplot** is a plotting library used for 2D graphics in python programming language.

Types of Plots

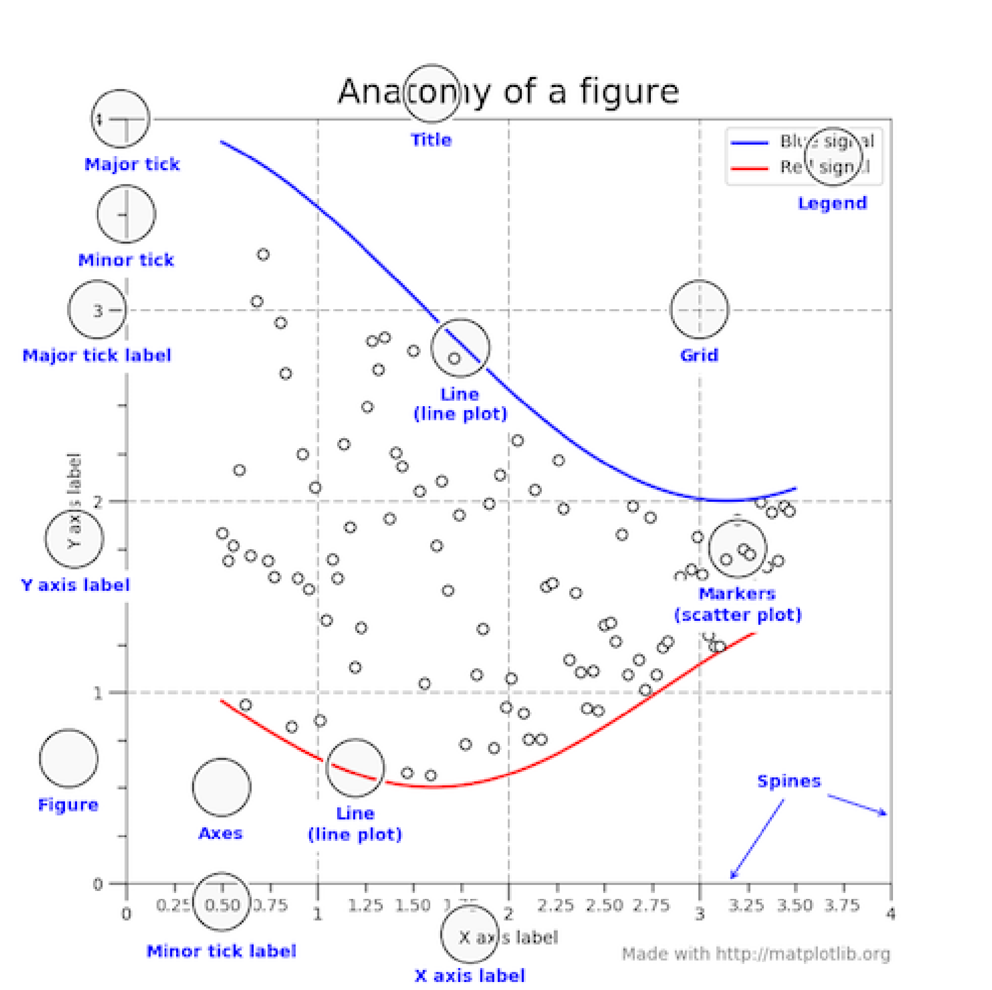
Bar Graph
Histogram
Scatter Plot
Area Plot
25%
Pie Plot

Matplotlib Architecture

Figure: It is a whole figure which may hold one or more axes (plots). We can think of a Figure as a canvas that holds plots.
Axes: A Figure can contain several Axes. It consists of two or three (in the case of 3D) Axis objects. Each Axes is comprised of a title, an x-label, and a y-label.
Axis: Axises are the number of line like objects and responsible for generating the graph limits.
Artist: An artist is the all which we see on the graph like Text objects, Line2D objects, and collection objects. Most Artists are tied to Axes.



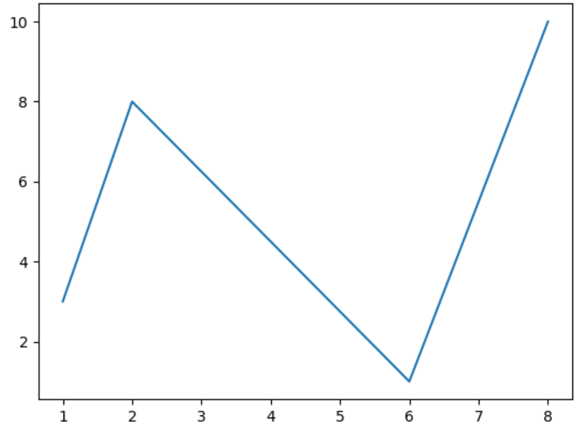
Figure



Anatomy of a figure
Title
Major tick
Minor tick
Major tick label
Y axis label
Figure
Axes
Line
(line plot)
Line
(line plot)
Grid
Legend
Blue signal
Red signal
Markers
(scatter plot)
Spines
Minor tick label
X axis label
Y axis label
X axis label
0
1
2
3
0.25
0.50
0.75
1.25
1.50
1.75
2.25
2.50
2.75
3.25
3.50
3.75
4
Made with http://matplotlib.org

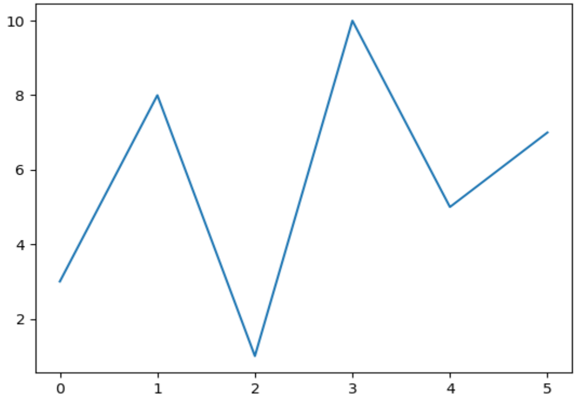
Matplotlib Plotting

```
xpoints = np.array([1, 2, 6, 8])
ypoints = np.array([3, 8, 1, 10])

plt.plot(xpoints, ypoints)
plt.show()
```



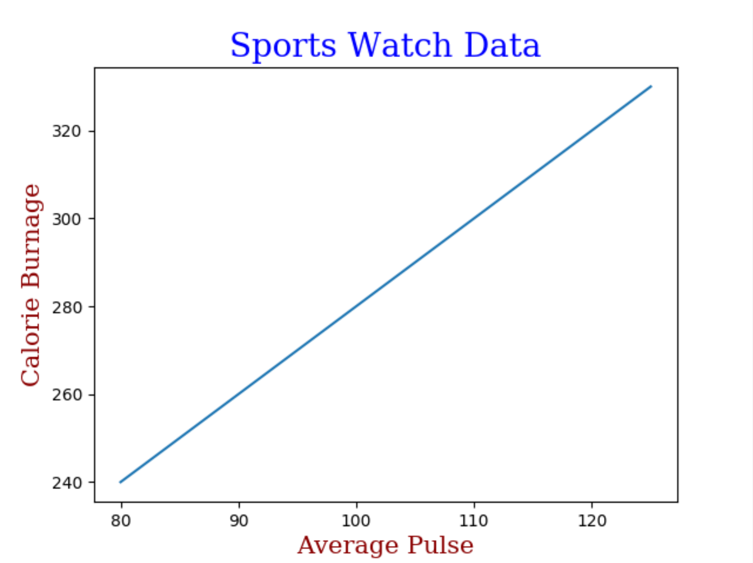
Default X-Points

```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3, 8, 1, 10, 5, 7])

plt.plot(ypoints)
plt.show()
```



Labels

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

plt.title("Sports Watch Data", fontdict = font1)
plt.xlabel("Average Pulse", fontdict = font2)
plt.ylabel("Calorie Burnage", fontdict = font2)
```



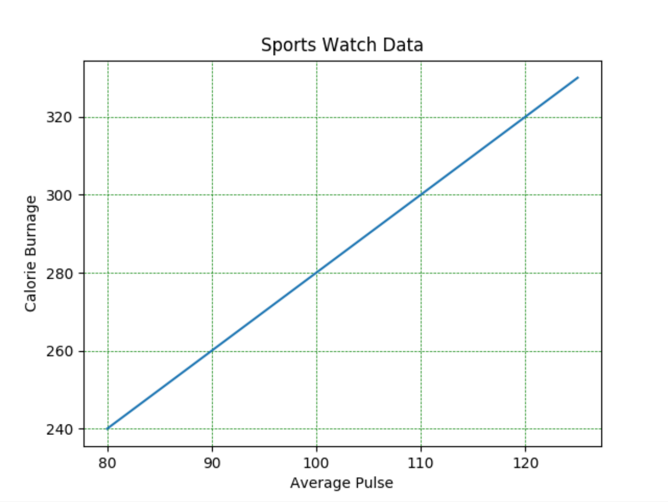
Grids

With Pyplot, you can use the grid() function to add grid lines to the plot

```
plt.grid()
```

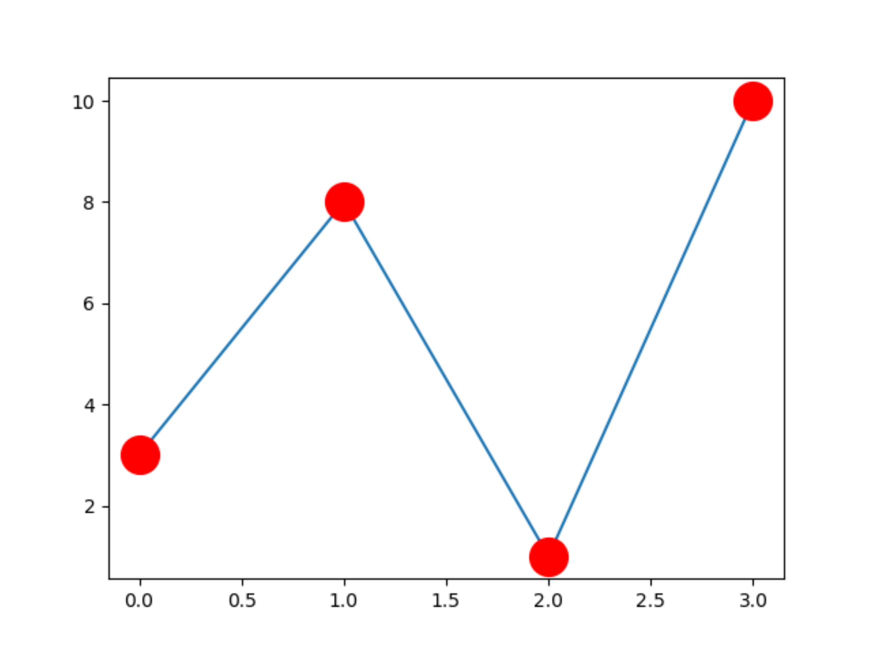
Or

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



Matplotlib Markers

```
plt.plot(ypoints, marker = 'o', ms = 20, mec = 'r', mfc = 'r')
```

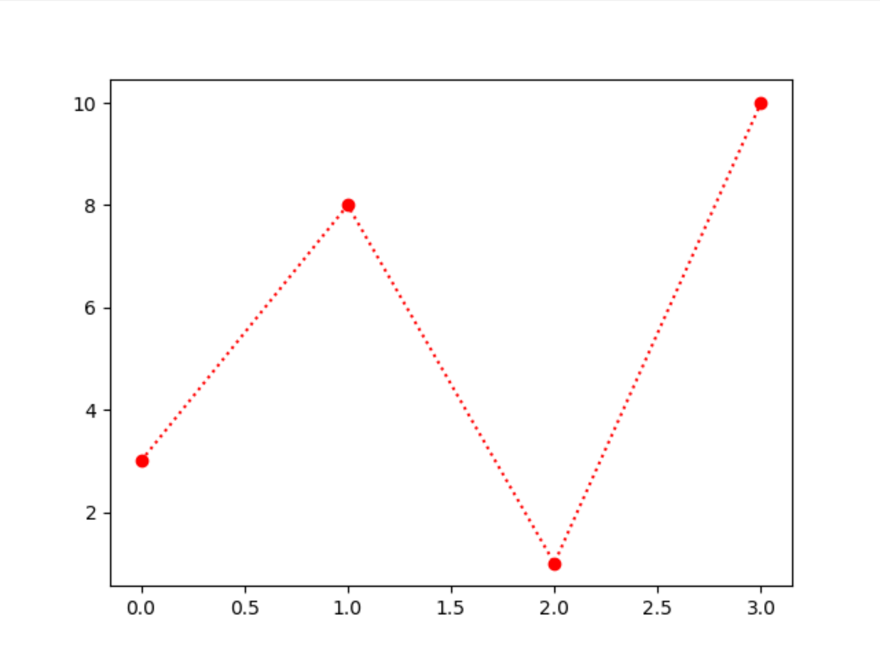


Matplotlib Markers Using format String

This parameter is also called fmt, and is written with this syntax:

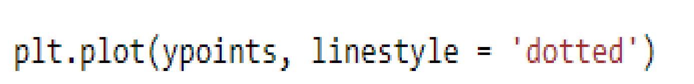
marker|line|color

```
plt.plot(ypoints, 'o:r')
```

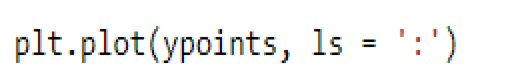


Matplotlib Lines

Line Style :

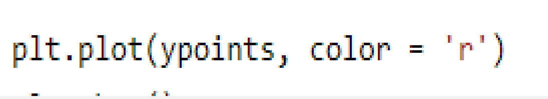


```
plt.plot(ypoints, linestyle = 'dotted')
```

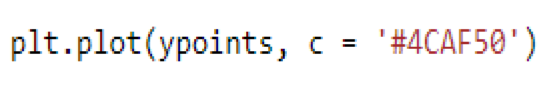


```
plt.plot(ypoints, ls = ':')
```

Line Color :

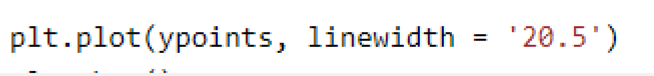


```
plt.plot(ypoints, color = 'r')
```



```
plt.plot(ypoints, c = '#4CAF50')
```

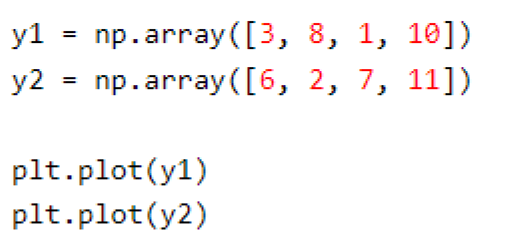
Line Width :



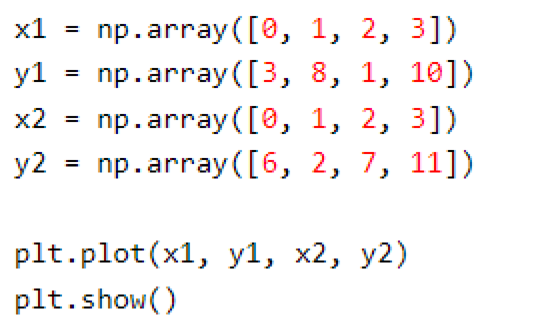
```
plt.plot(ypoints, linewidth = '20.5')
```

Multiple Lines

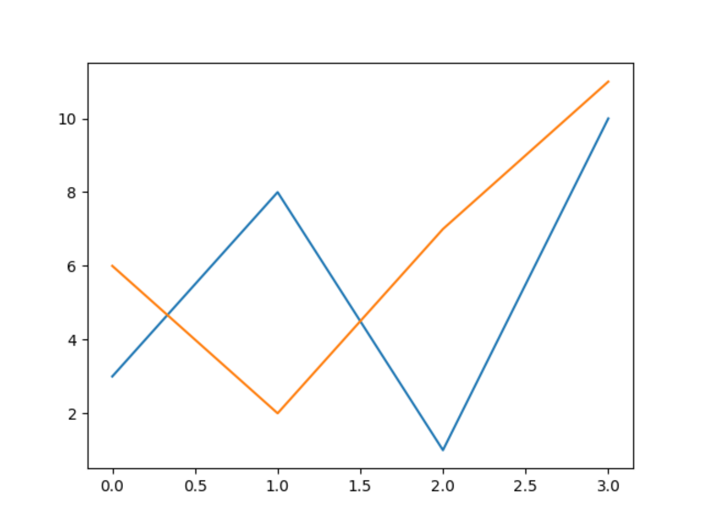
You can plot as many lines as you like by simply adding more plt.plot() functions:



```
y1 = np.array([3, 8, 1, 10])
y2 = np.array([6, 2, 7, 11])

plt.plot(y1)
plt.plot(y2)
```



```
x1 = np.array([0, 1, 2, 3])
y1 = np.array([3, 8, 1, 10])
x2 = np.array([0, 1, 2, 3])
y2 = np.array([6, 2, 7, 11])

plt.plot(x1, y1, x2, y2)
plt.show()
```

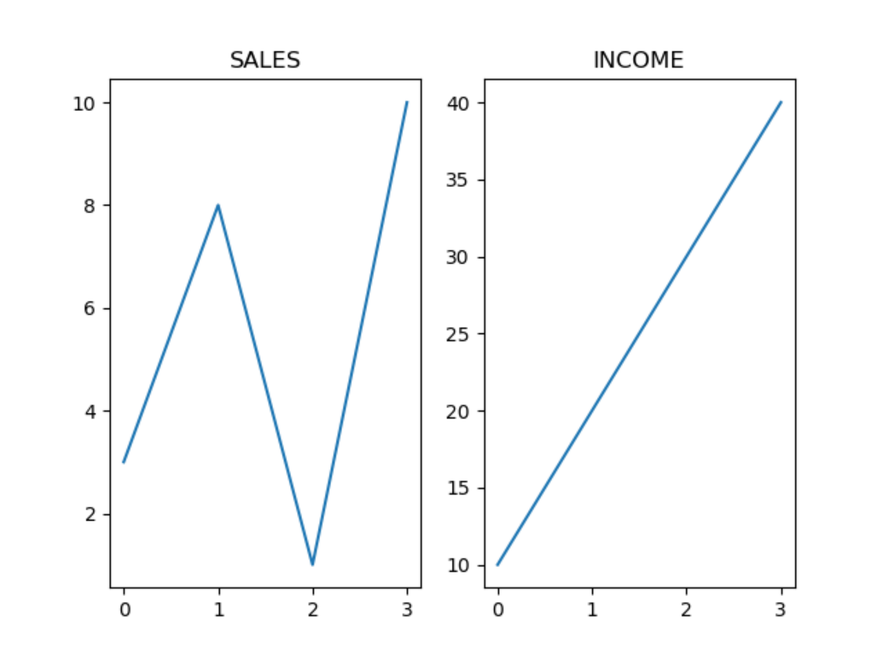


Subplots

With the subplots() function you can draw multiple plots in one figure

The subplots() function takes three arguments that describes the layout of the figure. The layout is organized in rows and columns, which are represented by the *first* and *second* argument. The third argument represents the index of the current plot.

```
#plot 1:
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])

plt.subplot(1, 2, 1)
plt.plot(x,y)
plt.title("SALES")

#plot 2:
x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

plt.subplot(1, 2, 2)
plt.plot(x,y)
plt.title("INCOME")
```

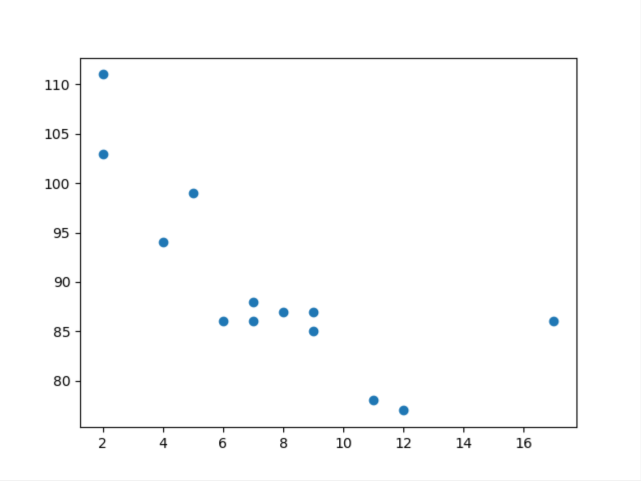


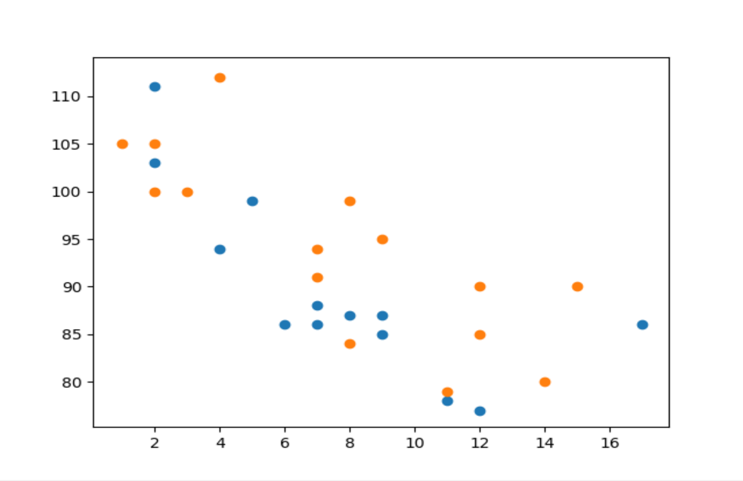
Matplotlib Scatter

With Pyplot, you can use the scatter() function to draw a scatter plot.

e.g.

```
plt.scatter(x, y)
```



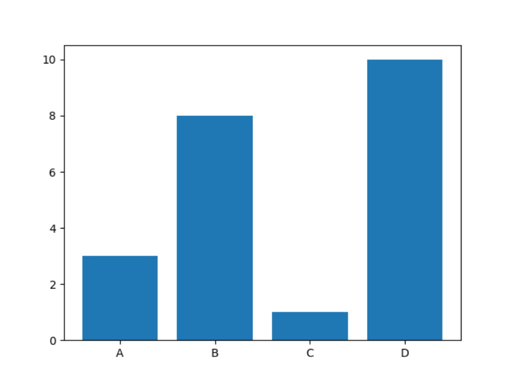


Matplotlib Bars

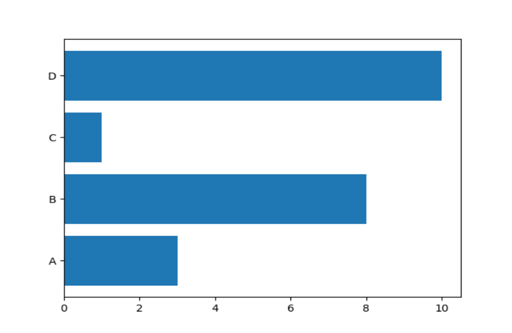
- **Creating Bars**

With Pyplot, you can use the bar() function to draw bar graphs:

```
x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])
plt.bar(x,y)
```

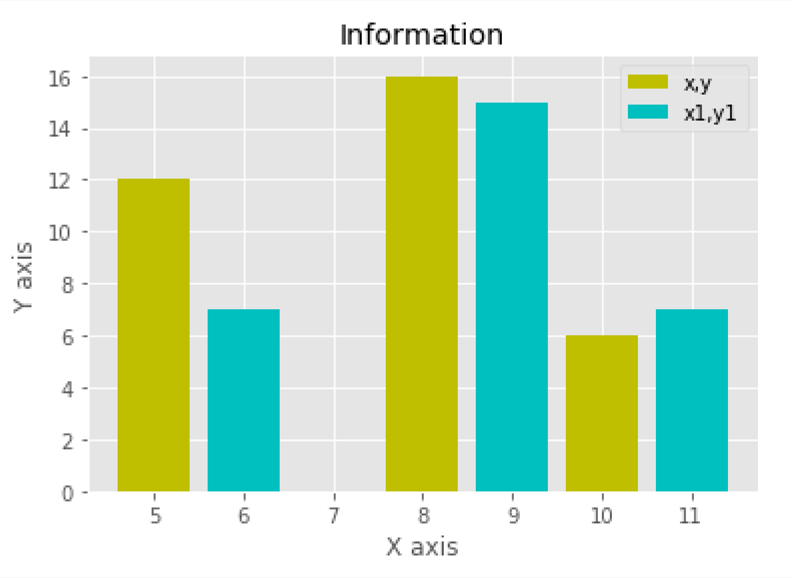


```
plt.barh(x, y ,color ..  ,width  ..)
```



Matplotlib Bars Using Style

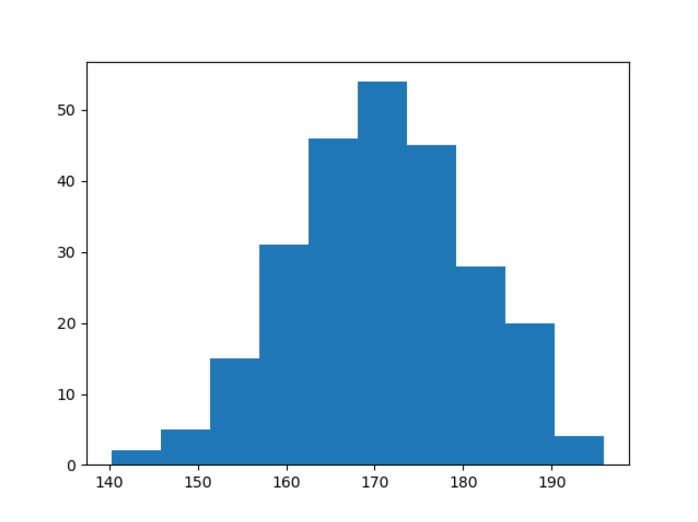
```
from matplotlib import style

style.use('ggplot')
x = [5,8,10]
y = [12,16,6]
x2 = [6,9,11]
y2 = [7,15,7]

plt.bar(x, y, color = 'y', align='center')
plt.bar(x2, y2, color='c', align='center')
```



Matplotlib Histograms

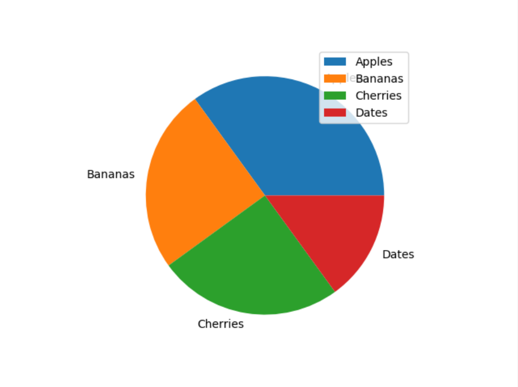
- **A histogram is a graph showing *frequency* distributions.**
- **It is a graph showing the number of observations within each given interval.**

```
x = np.random.normal(170, 10, 250)

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

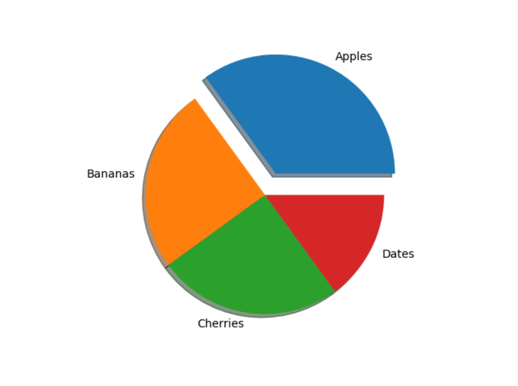


Matplotlib Pie Charts

-
  ```
  y = np.array([35, 25, 25, 15])
  mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
  plt.pie(y, labels = mylabels)
  plt.legend()
  ```

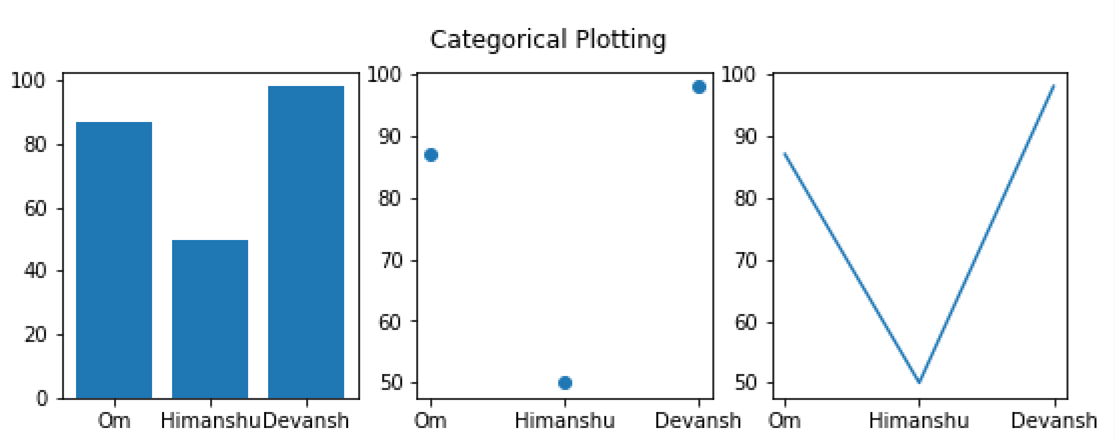


```
y = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
myexplode = [0.2, 0, 0, 0]

plt.pie(y, labels = mylabels, explode = myexplode, shadow = True)
```



Plotting with categorical variables

```
names = ['Abhishek', 'Himanshu', 'Devansh']

marks= [87,50,98]

plt.figure(figsize=(9,3))
plt.subplot(131)
plt.bar(names, marks)
plt.subplot(132)
plt.scatter(names, marks)
plt.subplot(133)
plt.plot(names, marks)
plt.suptitle('Categorical Plotting')
```



THANK YOU !!!

Amol Patil - 9822291613