Matplotlib:

Matplotlib is a plotting library for the Python Programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications. Some of the major Pros of MatplotLib are:

- Generally easy to get started for simple plots
- Support for custom label and texts
- High-Quality output in many formats
- Very Customizable in general

```
# Importing Libraries
import numpy as np           # To get data for our plots
import pandas as pd
import matplotlib.pyplot as plt             #pyplot is python plot
```

Double-click (or enter) to edit

```
# linspace is a function in numpy that return evenly spaced 100 value b/w 0 and 10
x = np.linspace(0,10,100)
y = np.sin(x)
z = np.cos(x)
```

```
print(x)
```

```
[ 0.          0.1010101   0.2020202   0.3030303   0.4040404   0.50505051
  0.60606061  0.70707071  0.80808081  0.90909091  1.01010101  1.11111111
  1.21212121  1.31313131  1.41414141  1.51515152  1.61616162  1.71717172
  1.81818182  1.91919192  2.02020202  2.12121212  2.22222222  2.32323232
  2.42424242  2.52525253  2.62626263  2.72727273  2.82828283  2.92929293
  3.03030303  3.13131313  3.23232323  3.33333333  3.43434343  3.53535354
  3.63636364  3.73737374  3.83838384  3.93939394  4.04040404  4.14141414
  4.24242424  4.34343434  4.44444444  4.54545455  4.64646465  4.74747475
  4.84848485  4.94949495  5.05050505  5.15151515  5.25252525  5.35353535
  5.45454545  5.55555556  5.65656566  5.75757576  5.85858586  5.95959596
  6.06060606  6.16161616  6.26262626  6.36363636  6.46464646  6.56565657
  6.66666667  6.76767677  6.86868687  6.96969697  7.07070707  7.17171717
  7.27272727  7.37373737  7.47474747  7.57575758  7.67676768  7.77777778
  7.87878788  7.97979798  8.08080808  8.18181818  8.28282828  8.38383838
  8.48484848  8.58585859  8.68686869  8.78787879  8.88888889  8.98989899
  9.09090909  9.19191919  9.29292929  9.39393939  9.49494949  9.5959596
  9.6969697   9.7979798   9.8989899  10.        ]
```

```
print(y)
```

```
[ 0.          0.10083842  0.20064886  0.2984138   0.39313661  0.48385164
  0.56963411  0.64960951  0.72296256  0.78894546  0.84688556  0.8961922
  0.93636273  0.96698762  0.98775469  0.99845223  0.99897117  0.98930624
  0.96955595  0.93992165  0.90070545  0.85230712  0.79522006  0.73002623
  0.65739025  0.57805259  0.49282204  0.40256749  0.30820902  0.21070855
  0.11106004  0.01027934 -0.09060615 -0.19056796 -0.28858706 -0.38366419
 -0.47483011 -0.56115544 -0.64176014 -0.7158225  -0.7825875  -0.84137452
 -0.89158426 -0.93270486 -0.96431712 -0.98609877 -0.99782778 -0.99938456
 -0.99075324 -0.97202182 -0.94338126 -0.90512352 -0.85763861 -0.80141062
 -0.73701276 -0.66510151 -0.58640998 -0.50174037 -0.41195583 -0.31797166
 -0.22074597 -0.12126992 -0.0205576   0.0803643   0.18046693  0.27872982
  0.37415123  0.46575841  0.55261747  0.63384295  0.7086068   0.77614685
  0.83577457  0.8868821   0.92894843  0.96154471  0.98433866  0.99709789
  0.99969234  0.99209556  0.97438499  0.94674118  0.90944594  0.86287948
  0.8075165   0.74392141  0.6727425   0.59470541  0.51060568  0.42130064
  0.32770071  0.23076008  0.13146699  0.03083368 -0.07011396 -0.17034683
 -0.26884313 -0.36459873 -0.45663749 -0.54402111]
```

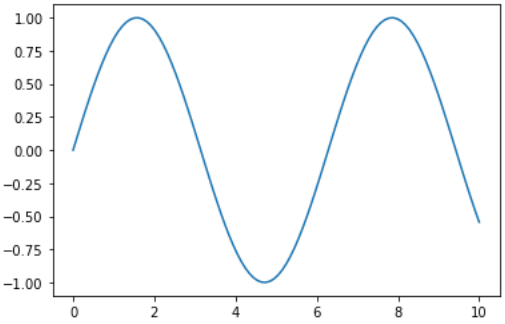
```
print(z)
```

```
[ 1.          0.99490282  0.97966323  0.95443659  0.91948007  0.87515004
  0.8218984   0.76026803  0.69088721  0.61446323  0.53177518  0.44366602
  0.35103397  0.25482335  0.15601496  0.0556161  -0.04534973 -0.14585325
 -0.24486989 -0.34139023 -0.43443032 -0.52304166 -0.60632092 -0.68341913
 -0.75355031 -0.81599952 -0.87013012 -0.91539031 -0.95131866 -0.97754893
 -0.9938137  -0.99994717 -0.9958868  -0.981674   -0.95745366 -0.92347268
 -0.88007748 -0.82771044 -0.76690542 -0.69828229 -0.6225406  -0.54045251
 -0.45285485 -0.36064061 -0.26474988 -0.16616018 -0.06587659  0.03507857
  0.13567613  0.23489055  0.33171042  0.4251487   0.51425287  0.59811455
  0.67587883  0.74675295  0.8100144   0.86501827  0.91120382  0.94810022
  0.97533134  0.99261957  0.99978867  0.99676556  0.98358105  0.96036956
  0.9273677   0.88491192  0.83343502  0.77346177  0.70560358  0.63055219
  0.54907273  0.46199582  0.37020915  0.27464844  0.17628785  0.07613012
 -0.0248037  -0.12548467 -0.2248864  -0.32199555 -0.41582217 -0.50540974
 -0.58984498 -0.66826712 -0.7398767  -0.8039437  -0.859815   -0.90692104
```

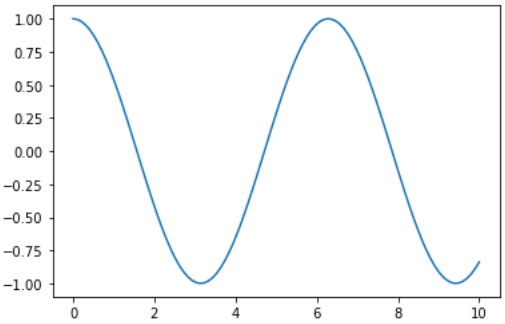
```
-0.94478159 -0.97301068 -0.99132055 -0.99952453 -0.99753899 -0.98538417
-0.96318398 -0.93116473 -0.88965286 -0.83907153]
```

Plotting the Data

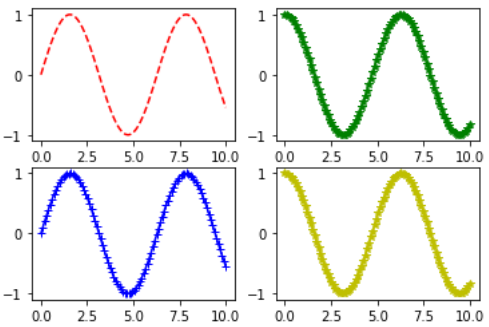
```
# Sine Wave
plt.plot(x,y)
plt.show()
```



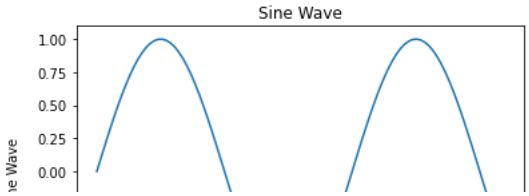
```
# Cosine Wave
plt.plot(x,z)
plt.show()
```



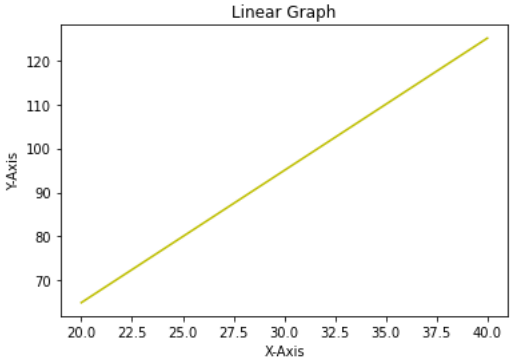
```
# Multiple plots:
# (2,2,1) having two rows, two columns and position of that plot is 1
plt.subplot(2,2,1)
plt.plot(x,y,'r--')
plt.subplot(2,2,2)
plt.plot(x,z,'g-*')
plt.subplot(2,2,3)
plt.plot(x,y,'b+-')
plt.subplot(2,2,4)
plt.plot(x,z,'y-*')
plt.show()
```



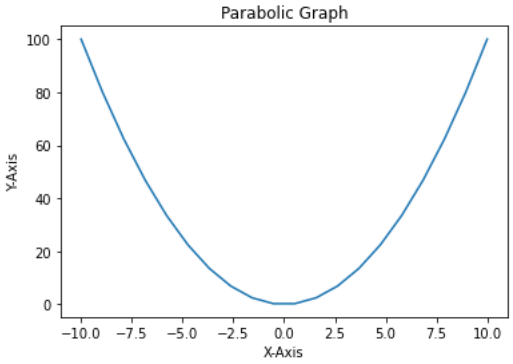
```
# Adding title to x-axis and y-axis labels
plt.plot(x,y)
plt.xlabel('Angle')
plt.ylabel('Sine Wave')
plt.title('Sine Wave')
plt.show()
```



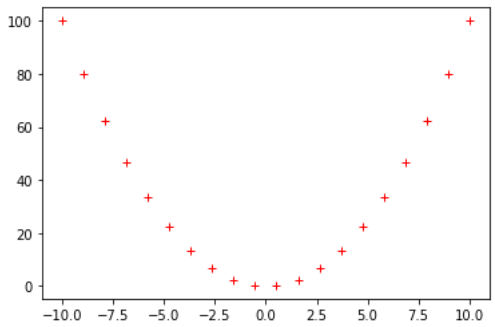

```
# Linear Graph
x = np.linspace(20,40,15)
y = 3*x + 5
plt.plot(x,y,'y-')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')
plt.title('Linear Graph')
plt.show()
```




```
# Parabola
x = np.linspace(-10,10,20)
y = x**2                # Equn of Parabola y= x^2
plt.plot(x,y)
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')
plt.title('Parabolic Graph')
plt.show()
```

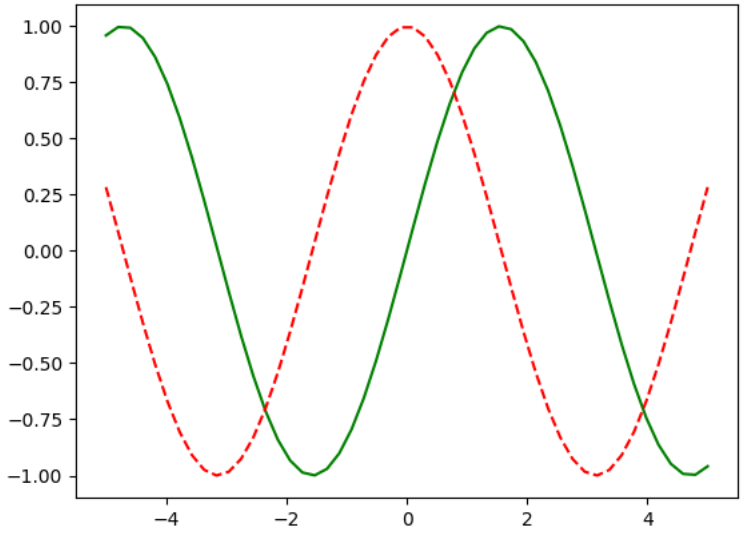



```
plt.plot(x,y,'r+')
plt.show()
```



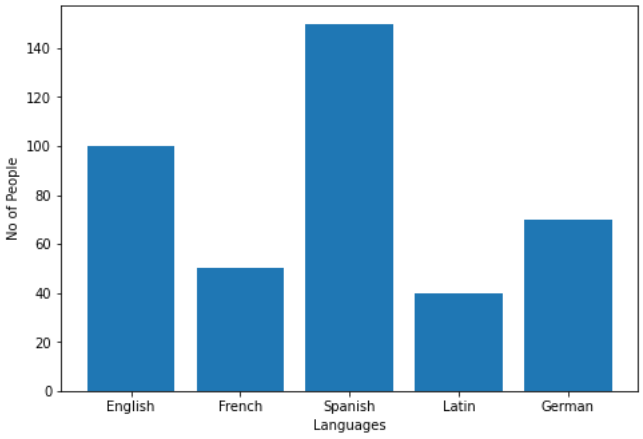

```
plt.plot(x,y,'g.')
plt.show()
```


```
# Multiple graphs in same plot
x = np.linspace(-5,5,50)
plt.plot(x,np.sin(x),'g-')
plt.plot(x,np.cos(x),'r--')
plt.savefig('SinCos.png')              # To download the figure
plt.show()
```



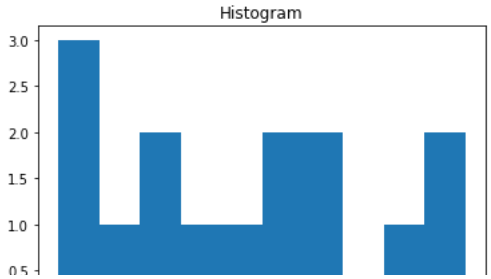
Bar Plot

```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])        # Area in which plot is defined coodinate(0,0) and lenght and height is 1
languages = ['English','French','Spanish','Latin','German']
people = [100,50,150,40,70]
ax.bar(languages,people)
plt.xlabel('Languages')
plt.ylabel('No of People')
plt.show()
# plt.savefig('BarGraph.jpeg')
```



Histograms

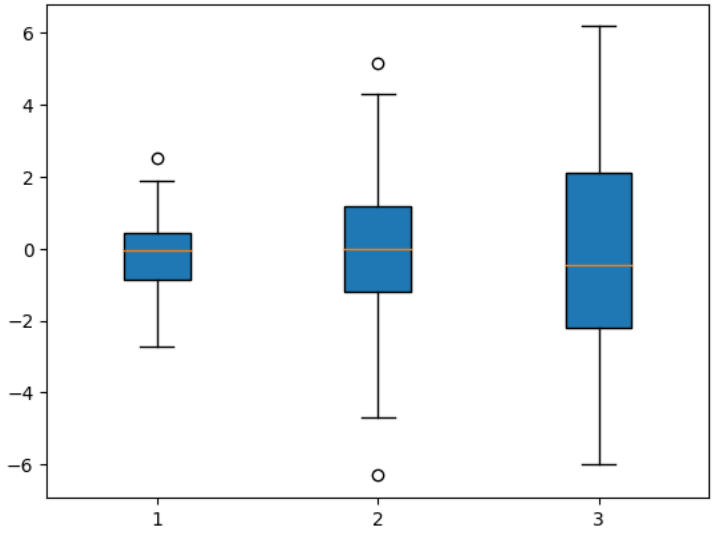
```
a = np.array([22,87,5,43,56,73,55,54,11,20,51,5,79,31,27])
plt.hist(a)
plt.title('Histogram')
plt.show()
```



Box Plot

It is used to plot percentiles

```
data = [np.random.normal(0, std, 100) for std in range(1,4)]
# Rectungular box plot
plt.boxplot(data, vert=True,patch_artist=True);       # vert=True means vertical plot and vert=False means horizontal plot
plt.show()                                            # Patch_artist=True gives colour
```



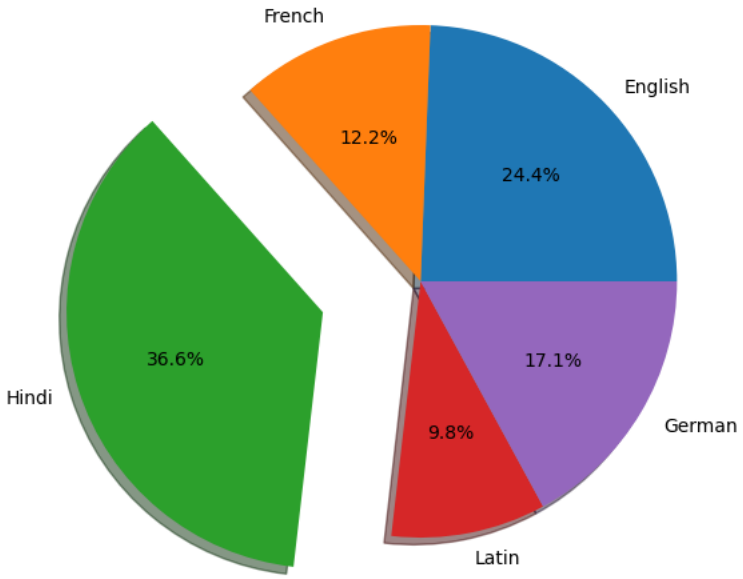

```
data
```

```
            -0.88347704, -0.61447485,  1.15587202,  0.82930649,  1.40028419,
            -1.27535789,  0.97922106,  0.28945142, -0.51194373,  0.86874839,
            -1.95685541,  0.10069888, -0.29096555, -0.36904877, -0.88393579,
             2.39904886, -0.78510951, -0.47787793, -1.95888003, -1.7759986 ,
             1.58533512,  1.14882139,  1.21162306,  0.3335587 ,  0.4117975 ,
             0.19520013, -1.16241773, -1.23249515, -2.10979292,  1.10094122,
             0.50690071,  1.33058838, -1.10704638,  0.79782552, -0.53121379,
            -0.15191895,  0.88155859,  2.15455966, -0.08758234, -0.2482907 ,
             0.59673984,  1.00300266,  0.08624747,  1.78002894, -0.68537896,
            -1.33548366,  0.16936179,  1.65044944,  1.02707573, -1.54424907,
            -0.03276242, -1.88286237, -0.13593019,  0.17873268, -0.01062903,
             2.46086343, -0.12486706,  0.4977548 , -1.17062723, -0.210924  ,
            -0.78301599, -0.89711823, -0.40399765,  2.57617274, -0.94746784,
             0.19053429,  1.31770685,  0.4229892 ,  1.17134795, -0.81662853,
             1.15739373, -0.28003753, -0.49726629, -2.29399281, -1.61705731,
             1.23670205, -0.9708677 , -0.84782819, -1.65169023,  0.2171765 ,
             0.31516612, -0.76341942, -0.33903113, -0.72787296,  0.53454941,
            -0.48166884, -0.63007943, -1.36482064, -1.38610755, -0.06927755]),
     array([-0.45987879,  1.16138769,  1.16659603,  0.05795138, -2.3096261 ,
            -1.34998988,  2.86185551,  1.62435089,  0.17749887,  0.58587558,
             3.96765965,  1.78037302, -1.65188491, -4.41583148, -0.49751337,
            -1.27439442, -0.88424967,  0.69734769, -1.26406066,  1.55977329,
             1.18179106, -0.66061455,  1.13197931,  1.47417753,  0.8907704 ,
            -0.86412484,  1.54766032, -2.75207823, -0.06465966, -0.5317329 ,
             1.55910884, -0.13220706,  3.07040721,  1.90438549,  1.40998925,
             0.39161785, -1.89903551, -0.73187277,  2.14617766, -2.43988456,
            -0.70250272, -0.41633132,  2.07389372, -0.7267551 ,  3.8905756 ,
             0.74060174, -2.56454261, -2.28356559,  0.39483769, -1.50941124,
             1.53675361, -4.51227532,  1.15494224,  0.93439202,  0.91330911,
            -0.84258732,  0.85336402, -3.6043402 ,  2.18537906, -1.27920924,
            -2.403633  , -0.05974665,  4.10315256,  1.04783579, -1.48704476,
             3.02977498, -1.96233926,  1.27114445, -0.99939213, -0.57103361,
             2.89494169, -2.30118822,  0.30344816,  3.74618381,  1.76457259,
            -2.00537809, -0.46246284,  2.1541636 , -1.55910748, -4.07974856,
            -1.16293275, -1.77955047,  0.32057528, -2.638999  ,  0.5365132 ,
            -1.28870432, -1.5641472 , -1.73214571,  0.11988115, -1.04564024,
            -2.45157821, -5.18934774,  1.39061461, -2.74450032,  0.0176401 ,
```

```
 4.41145662,  0.11460121,  2.87983063, -4.95050719, -3.05454284],
-1.96028194,  3.94811282,  9.52334596,  3.58545322,  5.64640126],
-1.61758152,  0.32235112, -3.02699298,  0.95703047, -0.38431415],
-3.04888526,  2.3275444 ,  2.14637294, -0.32901733,  3.62391443],
 6.63315619,  5.13316116, -3.31228372, -4.30706588, -0.83883396],
 1.13409376, -3.56879326, -1.26195907, -0.90532991,  0.2935327 ],
 1.26193677, -2.14192134, -3.91282168,  3.18306364,  3.0419118 ],
 2.0674823 , -2.47636051, -4.61583958,  0.23973827, -5.75801643],
-0.71576031, -3.85459416, -1.66592131, -4.9140376 , -2.26581349],
-0.57398341, -1.73345654, -5.46028428, -5.09272056, -0.45857382],
 0.38939822,  6.15066942,  0.39715826, -1.0360457 , -2.307384  ],
 0.4694015 ,  1.35787295,  5.47984469,  2.10089352,  0.59139234],
-0.88097976,  0.55056785,  2.9936798 , -5.23769818,  0.27047238],
-0.62281921,  0.50691396, -0.32367306,  1.6555451 ,  0.15501913],
 0.28576779, -3.42592862,  0.81221276,  3.2058876 , -1.59087734],
 0.65202584,  1.66028514, -3.30156897,  1.39069657, -0.5968571 ],
 1.47247157,  2.78236792, -0.08749004,  1.42375973,  0.63892549],
 0.94084013, -5.78332762,  3.87959597, -1.08858047, -1.30754247],
 3.89347906, -0.49700096,  0.59456745,  2.48827902, -0.47350425])]
```

Pie Chart

```
fig1 = plt.figure()
ax = fig1.add_axes([0,0,1,1])
languages =['English','French','Hindi','Latin','German']
people = [100,50,150,40,70]
ex = (0,0,0.4,0,0)              # Explode 1st slide
ax.pie(people, labels=languages, explode=ex, autopct='%1.1f%%',shadow=True)
plt.show()
```



Scatter Plot

```
# Used in Clustring Applications
x = np.linspace(0,10,30)
y = np.sin(x)
z = np.cos(x)
fig2= plt.figure()
ax = fig2.add_axes([0,0,1,1])
ax.scatter(x,y,color='g')
ax.scatter(x,z,color='b')
plt.show()
```

3D Scatter Plot

```
fig3 = plt.figure()
ax = plt.axes(projection ='3d')
z = 20 * np.random.random(100)
x = np.sin(z)
y = np.cos(z)
ax.scatter(x,y,z,cmap='Blues')
plt.show()
```