Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

Matplotlib consists of several plots like line, bar, scatter, histogram etc.

**Installation :**

python -mpip install -U matplotlib

pip install matplotlib

**Importing matplotlib :**

from matplotlib import pyplot as plt

or

import matplotlib.pyplot as plt

Line Graph

import matplotlib.pyplot as plt

# x axis values

x = [1,2,3]

# corresponding y axis values

y = [2,4,1]

# plotting the points

plt.plot(x, y)

# naming the x axis

plt.xlabel('x - axis')

# naming the y axis

plt.ylabel('y - axis')

# giving a title to my graph

plt.title('My first graph!')

# function to show the plot

plt.show()

**Two Line Graph**

import matplotlib.pyplot as plt

# line 1 points

x1 = [1,2,3]

y1 = [2,4,1]

# plotting the line 1 points

plt.plot(x1, y1, label = "line 1")

#plt.plot(x1, y1)

# line 2 points

x2 = [1,2,3]

y2 = [4,1,3]

# plotting the line 2 points

plt.plot(x2, y2, label = "line 2")

#plt.plot(x2, y2)

# naming the x axis

plt.xlabel('x - axis')

# naming the y axis

plt.ylabel('y - axis')

# giving a title to my graph

plt.title('Two lines on same graph!')

# show a legend on the plot

plt.legend()

# function to show the plot

plt.show()

**Customized\_graph**

import matplotlib.pyplot as plt

# x axis values

x = [1,2,3,4,5,6]

# corresponding y axis values

y = [2,4,1,5,2,6]

# plotting the points

plt.plot(x, y, color='green', linestyle='dashed', linewidth = 1.5,

marker='o', markerfacecolor='blue', markersize=8)

# setting x and y axis range

plt.ylim(1,8)

plt.xlim(1,8)

# naming the x axis

plt.xlabel('x - axis')

# naming the y axis

plt.ylabel('y - axis')

# giving a title to my graph

plt.title('Customized Grah!')

# function to show the plot

plt.show()

**Bar Graph**

import matplotlib.pyplot as plt

# x-coordinates of left sides of bars

left = [1, 2, 3, 4, 5]

# heights of bars

height = [10, 24, 36, 40, 5]

# labels for bars

tick\_label = ['one', 'two', 'three', 'four', 'five']

# plotting a bar chart

plt.bar(left, height, tick\_label = tick\_label,

width = 0.8, color = ['red', 'green'])

#plt.bar(left, height,

#width = 0.8, color = ['red', 'green'])

# naming the x-axis

plt.xlabel('x - axis')

# naming the y-axis

plt.ylabel('y - axis')

# plot title

plt.title('Bar chart!')

# function to show the plot

plt.show()

* Here, we use **plt.bar()** function to plot a bar chart.
* x-coordinates of left side of bars are passed along with heights of bars.
* you can also give some name to x-axis coordinates by defining **tick\_labels**

**Histogram**

import matplotlib.pyplot as plt

# frequencies

ages = [2,5,70,40,30,45,50,45,43,40,44,

        60,7,13,57,18,90,77,32,21,20,40]

# setting the ranges and no. of intervals

range = (0, 100)

bins = 10

# plotting a histogram

plt.hist(ages, bins, range, color = 'green',

        histtype = 'bar', rwidth = 0.8)

# x-axis label

plt.xlabel('age')

# frequency label

plt.ylabel('No. of people')

# plot title

plt.title('My histogram')

# function to show the plot

plt.show()

* Here, we use **plt.hist()** function to plot a histogram.
* frequencies are passed as the **ages** list.
* Range could be set by defining a tuple containing min and max value.
* Next step is to “**bin**” the range of values—that is, divide the entire range of values into a series of intervals—and then count how many values fall into each interval. Here we have defined **bins** = 10. So, there are a total of 100/10 = 10 intervals.

Scatter Plot

import matplotlib.pyplot as plt

#from IPython import get\_ipython

# x-axis values

x = [1,2,3,4,5,6,7,8,9,10]

# y-axis values

y = [2,4,5,7,6,8,9,11,12,12]

# plotting points as a scatter plot

plt.scatter(x, y, label= "stars", color= "red",

marker= "+", s=30)

# x-axis label

plt.xlabel('x - axis')

# frequency label

plt.ylabel('y - axis')

# plot title

plt.title('My scatter plot!')

# showing legend

plt.legend()

# function to show the plot

plt.show()

#get\_ipython().run\_line\_magic('matplotlib', 'qt')

* Here, we use **plt.scatter()** function to plot a scatter plot.
* Like a line, we define x and corresponding y – axis values here as well.
* **marker** argument is used to set the character to use as marker. Its size can be defined using **s** parameter.

Pie\_Chart:

import matplotlib.pyplot as plt

# defining labels

activities = ['eat', 'sleep', 'work', 'play']

# portion covered by each label

slices = [3, 7, 8, 6]

# color for each label

colors = ['r', 'y', 'g', 'b']

# plotting the pie chart

plt.pie(slices, labels = activities, colors=colors,

        startangle=90, shadow = True, explode = (0, 0, 0.1, 0),

        radius = 1.2, autopct = '%1.1f%%')

# plotting legend

plt.legend()

# showing the plot

plt.show()