

[Data Types](#) [Control Flow](#) [Functions](#) [List](#) [String](#) [Set](#) [Tuple](#) [Dictionary](#) [Oops](#) [Exception Handl](#)

# Data Visualization using Matplotlib



nikhilaggarwal3

[Read](#) [Discuss](#) [Courses](#) [Practice](#) [Video](#)

**Data Visualization** is the process of presenting data in the form of graphs or charts. It helps to understand large and complex amounts of data very easily. It allows the decision-makers to make decisions very efficiently and also allows them in identifying new trends and patterns very easily. It is also used in high-level data analysis for Machine Learning and Exploratory Data Analysis (EDA). Data visualization can be done with various tools like Tableau, Power BI, Python.

In this article, we will discuss how to visualize data with the help of the Matplotlib library of Python.

## Matplotlib

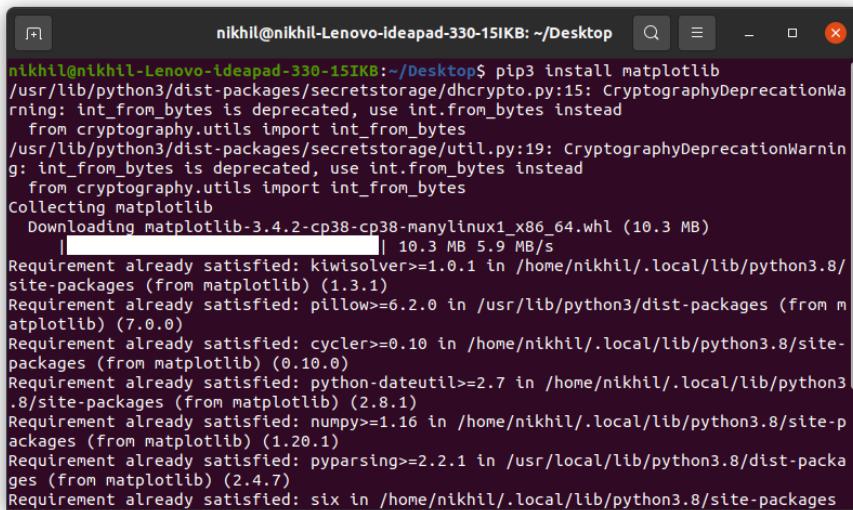
Matplotlib is a low-level library of Python which is used for data visualization. It is easy to use and emulates MATLAB like graphs and visualization. This library is built on the top of NumPy arrays and consist of several plots like line chart, bar chart, histogram, etc. It provides a lot of flexibility but at the cost of writing more code.

## Installation

We will use the pip command to install this module. If you do not have pip installed then refer to the article, [Download and install pip Latest Version.](#)

To install Matplotlib type the below command in the terminal.

```
pip install matplotlib
```



```
nikhil@nikhil-Lenovo-ideapad-330-15IKB: ~/Desktop$ pip3 install matplotlib
/usr/lib/python3/dist-packages/secretstorage/dhcrypto.py:15: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_bytes instead
    from cryptography.utils import int_from_bytes
/usr/lib/python3/dist-packages/secretstorage/util.py:19: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.from_bytes instead
    from cryptography.utils import int_from_bytes
Collecting matplotlib
  Downloading matplotlib-3.4.2-cp38-cp38-manylinux1_x86_64.whl (10.3 MB)
    |██████████| 10.3 MB 5.9 MB/s
Requirement already satisfied: kiwisolver>=1.0.1 in /home/nikhil/.local/lib/python3.8/site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/lib/python3/dist-packages (from matplotlib) (7.0.0)
Requirement already satisfied: cycler>=0.10 in /home/nikhil/.local/lib/python3.8/site-packages (from matplotlib) (0.10.0)
Requirement already satisfied: python-dateutil>=2.7 in /home/nikhil/.local/lib/python3.8/site-packages (from matplotlib) (2.8.1)
Requirement already satisfied: numpy>=1.16 in /home/nikhil/.local/lib/python3.8/site-packages (from matplotlib) (1.20.1)
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.8/dist-packages (from matplotlib) (2.4.7)
Requirement already satisfied: six in /home/nikhil/.local/lib/python3.8/site-packages
```

Refer to the below articles to get more information setting up an environment with Matplotlib.

- [Environment Setup for Matplotlib](#)
- [Using Matplotlib with Jupyter Notebook](#)

## Pyplot

[\*\*Pyplot\*\*](#) is a Matplotlib module that provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python and the advantage of being free and open-source. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. The various plots we can utilize using Pyplot are Line Plot, Histogram, Scatter, 3D Plot, Image, Contour, and Polar.

simple plot.

### Example:

## Python3

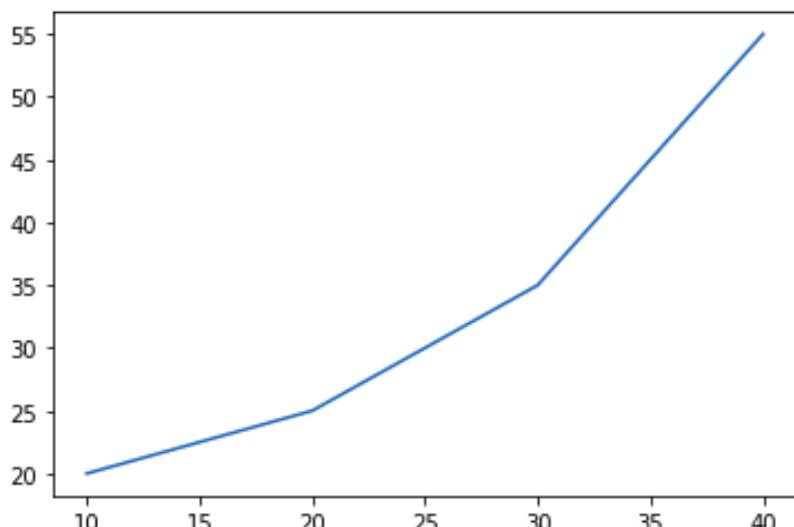
```
import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

plt.show()
```

### Output:



Now let see how to add some basic elements like title, legends, labels to the graph.

**Note:** For more information about Pyplot, refer [Pyplot in Matplotlib](#)

## Adding Title

The [`title\(\)`](#) method in matplotlib module is used to specify the title of the visualization depicted and displays the title using various attributes.



We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
matplotlib.pyplot.title(label, fontdict=None, loc='center', pad=None,  
**kwargs)
```

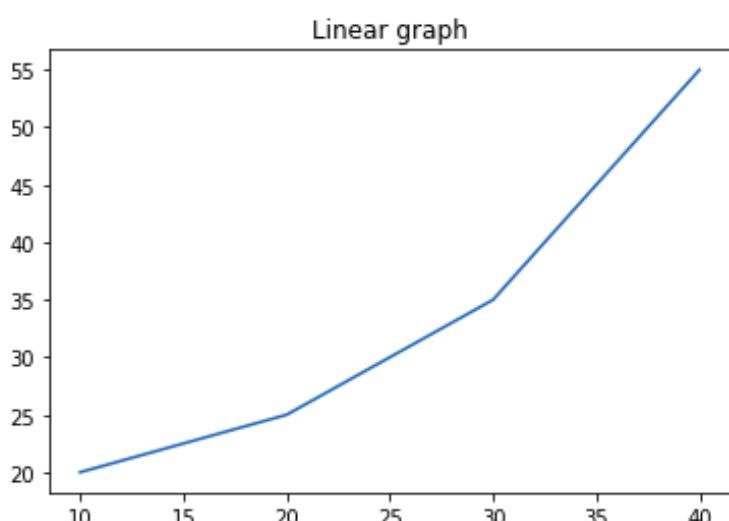
## Example:

---

### Python3

```
import matplotlib.pyplot as plt  
  
# initializing the data  
x = [10, 20, 30, 40]  
y = [20, 25, 35, 55]  
  
# plotting the data  
plt.plot(x, y)  
  
# Adding title to the plot  
plt.title("Linear graph")  
  
plt.show()
```

## Output:



We can also change the appearance of the title by using the parameters of this function.

## Python3

```
import matplotlib.pyplot as plt

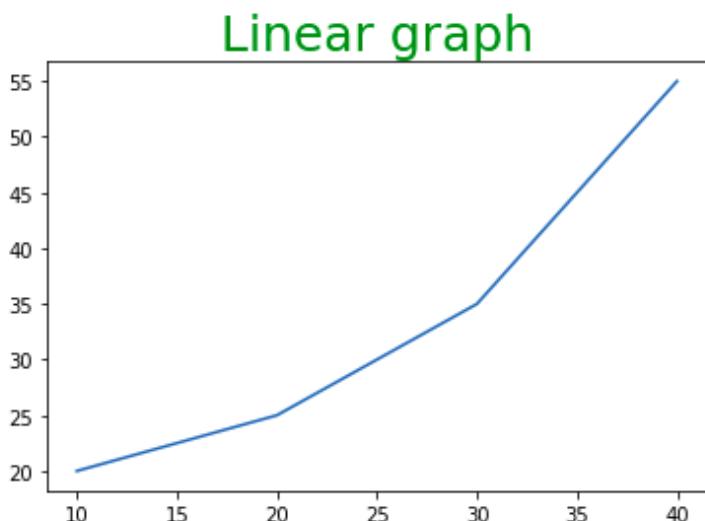
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="green")

plt.show()
```

### Output:



**Note:** For more information about adding the title and its customization, refer [Matplotlib.pyplot.title\(\) in Python](#)

## Adding X Label and Y Label

In layman's terms, the X label and the Y label are the titles given to X-axis and Y-axis respectively. These can be added to the graph by using the [xlabel\(\)](#) and [ylabel\(\)](#) methods.

### Syntax:

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
**kwargs)

matplotlib.pyplot.ylabel(ylabel, fontdict=None, labelpad=None,
**kwargs)
```

## Example:

---

### Python3

```
import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

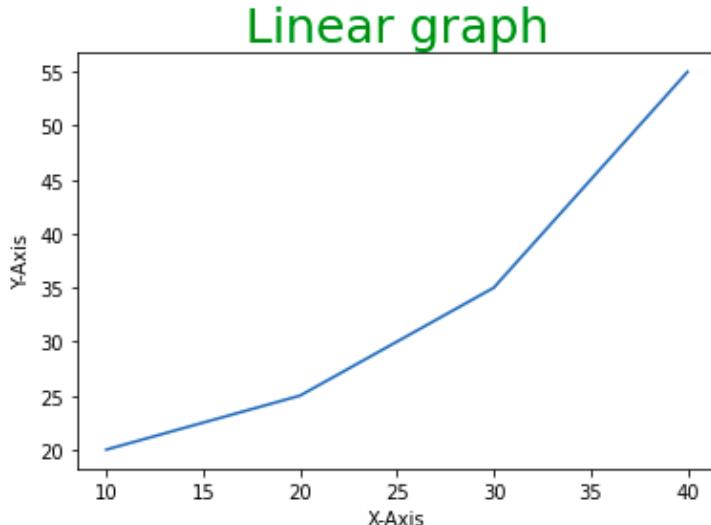
# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="green")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

# Adding label on the x-axis
plt.xlabel('X-Axis')

plt.show()
```

## Output:



## Setting Limits and Tick labels

You might have seen that Matplotlib automatically sets the values and the markers(points) of the X and Y axis, however, it is possible to set the limit and markers manually. [xlim\(\)](#) and [ylim\(\)](#) functions are used to set the limits of the X-axis and Y-axis respectively. Similarly, [xticks\(\)](#) and [yticks\(\)](#) functions are used to set tick labels.

**Example:** In this example, we will be changing the limit of Y-axis and will be setting the labels for X-axis.

---

### Python3

```
import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="green")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

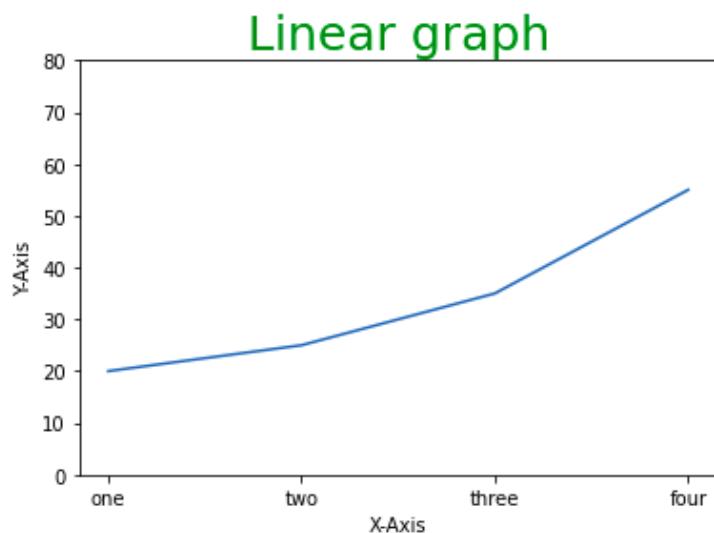
# Adding label on the x-axis
```

```
plt.ylim(0, 80)

# setting the labels of x-axis
plt.xticks(x, labels=["one", "two", "three", "four"])

plt.show()
```

## Output:



## Adding Legends

A legend is an area describing the elements of the graph. In simple terms, it reflects the data displayed in the graph's Y-axis. It generally appears as the box containing a small sample of each color on the graph and a small description of what this data means.

The attribute `bbox_to_anchor=(x, y)` of `legend()` function is used to specify the coordinates of the legend, and the attribute `ncol` represents the number of columns that the legend has. Its default value is 1.

## Syntax:

```
matplotlib.pyplot.legend(["name1", "name2"], bbox_to_anchor=(x, y),
ncol=1)
```

## Example:-

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

## Python3

```
import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="green")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

# Adding label on the x-axis
plt.xlabel('X-Axis')

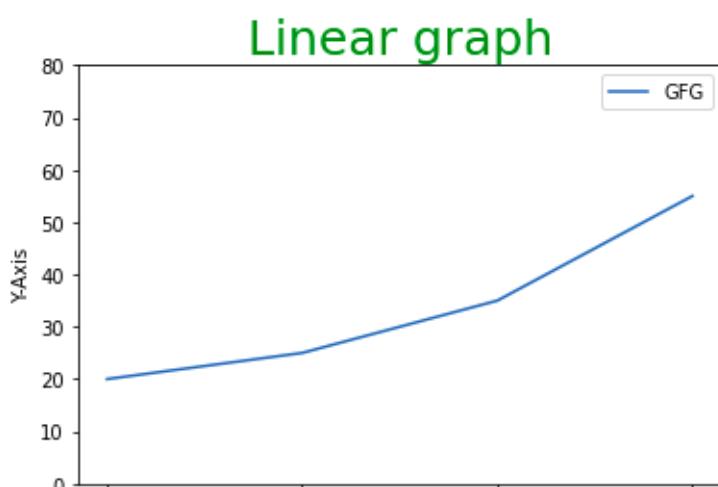
# Setting the limit of y-axis
plt.ylim(0, 80)

# setting the labels of x-axis
plt.xticks(x, labels=["one", "two", "three", "four"])

# Adding legends
plt.legend(["GFG"])

plt.show()
```

### Output:



We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

Before moving any further with Matplotlib let's discuss some important classes that will be used further in the tutorial. These classes are –

- **Figure**
- **Axes**

**Note:** Matplotlib take care of the creation of inbuilt defaults like Figure and Axes.

## Figure class

Consider the figure class as the overall window or page on which everything is drawn. It is a top-level container that contains one or more axes. A figure can be created using the `figure()` method.

### Syntax:

```
class matplotlib.figure.Figure(figsize=None, dpi=None,  
facecolor=None, edgecolor=None, linewidth=0.0, frameon=None,  
subplotpars=None, tight_layout=None, constrained_layout=None)
```

### Example:

## Python3

```
# Python program to show pyplot module  
import matplotlib.pyplot as plt  
from matplotlib.figure import Figure  
  
# initializing the data  
x = [10, 20, 30, 40]  
y = [20, 25, 35, 55]  
  
# Creating a new figure with width = 7 inches  
# and height = 5 inches with face color as  
# green, edgecolor as red and the line width  
# of the edge as 7  
fig = plt.figure(figsize =(7, 5), facecolor='g',  
                 edgecolor='b', linewidth=7)  
  
# Creating a new axes for the figure
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
# Adding the data to be plotted
ax.plot(x, y)

# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="yellow")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

# Adding label on the x-axis
plt.xlabel('X-Axis')

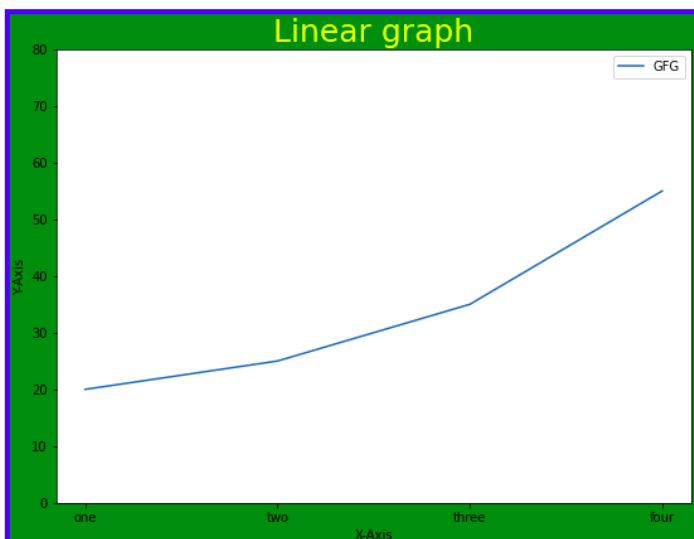
# Setting the limit of y-axis
plt.ylim(0, 80)

# setting the labels of x-axis
plt.xticks(x, labels=["one", "two", "three", "four"])

# Adding legends
plt.legend(["GFG"])

plt.show()
```

## Output:



## >>> More Functions in Figure Class

## Axes Class

Axes class is the most basic and flexible unit for creating sub-plots. A given figure may contain many axes, but a given axes can only be present in one figure. The `axes()` function creates the axes object.

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

*axes([left, bottom, width, height])*

Just like pyplot class, axes class also provides methods for adding titles, legends, limits, labels, etc. Let's see a few of them –

- **Adding Title** – [ax.set\\_title\(\)](#)
- **Adding X Label and Y label** – [ax.set\\_xlabel\(\)](#), [ax.set\\_ylabel\(\)](#)
- **Setting Limits** – [ax.set\\_xlim\(\)](#), [ax.set\\_ylim\(\)](#)
- **Tick labels** – [ax.set\\_xticklabels\(\)](#), [ax.set\\_yticklabels\(\)](#)
- **Adding Legends** – [ax.legend\(\)](#)

**Example:**

---

## Python3

```
# Python program to show pyplot module
import matplotlib.pyplot as plt
from matplotlib.figure import Figure

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

fig = plt.figure(figsize = (5, 4))

# Adding the axes to the figure
ax = fig.add_axes([1, 1, 1, 1])

# plotting 1st dataset to the figure
ax1 = ax.plot(x, y)

# plotting 2nd dataset to the figure
ax2 = ax.plot(y, x)

# Setting Title
ax.set_title("Linear Graph")

# Setting Label
ax.set_xlabel("X-Axis")
ax.set_ylabel("Y-Axis")

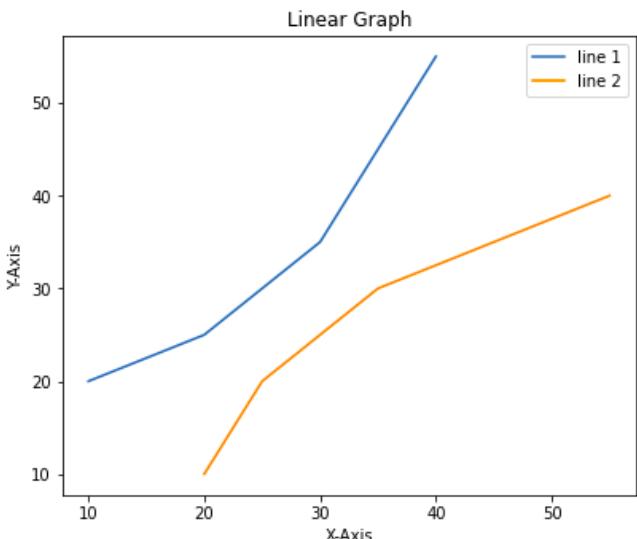
# Adding Legend
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
plt.show()
```

## Output:



## Multiple Plots

We have learned about the basic components of a graph that can be added so that it can convey more information. One method can be by calling the plot function again and again with a different set of values as shown in the above example. Now let's see how to plot multiple graphs using some functions and also how to plot subplots.

### Method 1: Using the `add_axes()` method

The `add_axes()` method is used to add axes to the figure. This is a method of figure class

#### Syntax:

```
add_axes(self, *args, **kwargs)
```

#### Example:

### Python3

```
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# Creating a new figure with width = 5 inches
# and height = 4 inches
fig = plt.figure(figsize =(5, 4))

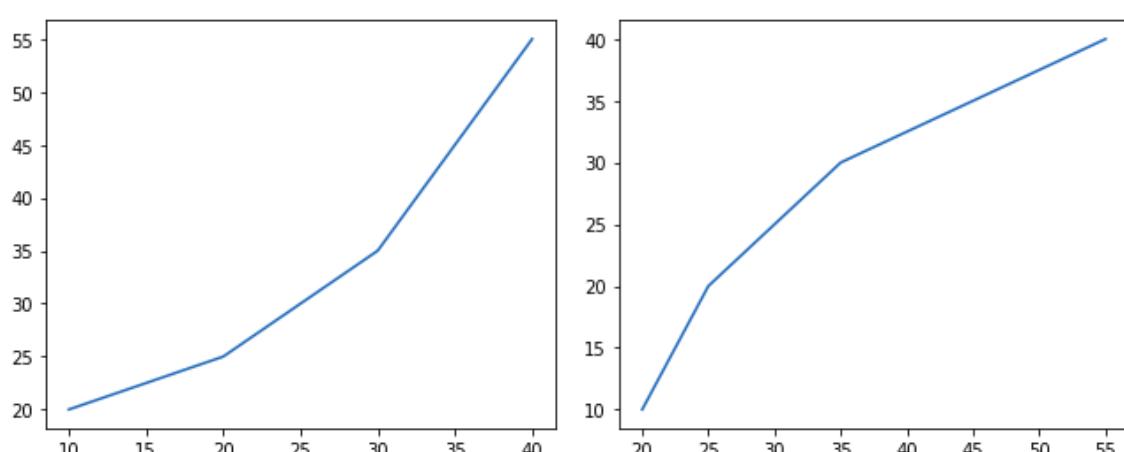
# Creating first axes for the figure
ax1 = fig.add_axes([0.1, 0.1, 0.8, 0.8])

# Creating second axes for the figure
ax2 = fig.add_axes([1, 0.1, 0.8, 0.8])

# Adding the data to be plotted
ax1.plot(x, y)
ax2.plot(y, x)

plt.show()
```

## Output:



## Method 2: Using `subplot()` method.

This method adds another plot at the specified grid position in the current figure.

### Syntax:

```
subplot(nrows, ncols, index, **kwargs)
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

`subplot(ax)`

### Example:

## Python3

```
import matplotlib.pyplot as plt

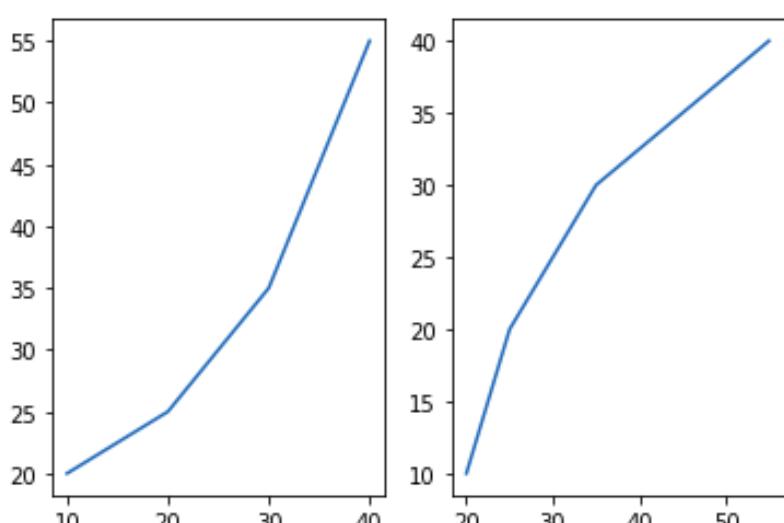
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# Creating figure object
plt.figure()

# adding first subplot
plt.subplot(121)
plt.plot(x, y)

# adding second subplot
plt.subplot(122)
plt.plot(y, x)
```

### Output:



### Method 3: Using `subplots()` method

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

time.

### Syntax:

```
matplotlib.pyplot.subplots(nrows=1, ncols=1, sharex=False,  
sharey=False, squeeze=True, subplot_kw=None, gridspec_kw=None,  
**fig_kw)
```

### Example:

---

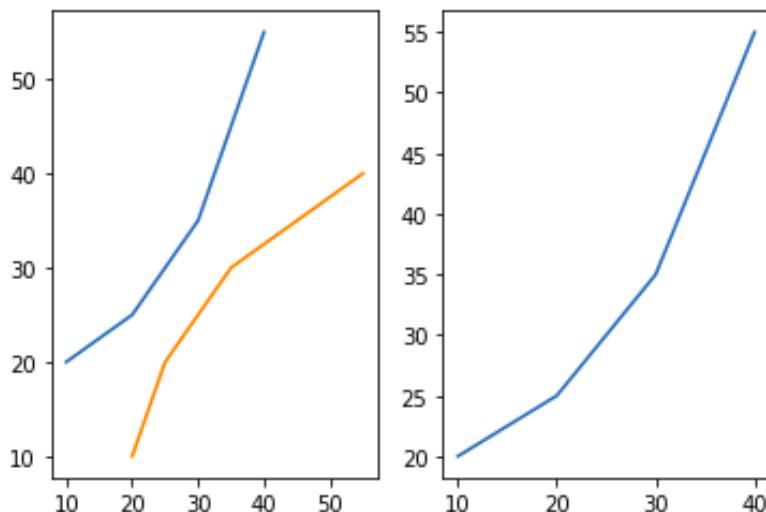
### Python3

```
import matplotlib.pyplot as plt  
  
# initializing the data  
x = [10, 20, 30, 40]  
y = [20, 25, 35, 55]  
  
# Creating the figure and subplots  
# according the argument passed  
fig, axes = plt.subplots(1, 2)  
  
# plotting the data in the  
# 1st subplot  
axes[0].plot(x, y)  
  
# plotting the data in the 1st  
# subplot only  
axes[0].plot(y, x)  
  
# plotting the data in the 2nd  
# subplot only  
axes[1].plot(x, y)
```

### Output:

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**



#### Method 4: Using [subplot2grid\(\)](#) method

This function creates axes object at a specified location inside a grid and also helps in spanning the axes object across multiple rows or columns. In simpler words, this function is used to create multiple charts within the same figure.

#### Syntax:

*Plt.subplot2grid(shape, location, rowspan, colspan)*

---

#### Example:

#### Python3

```
import matplotlib.pyplot as plt

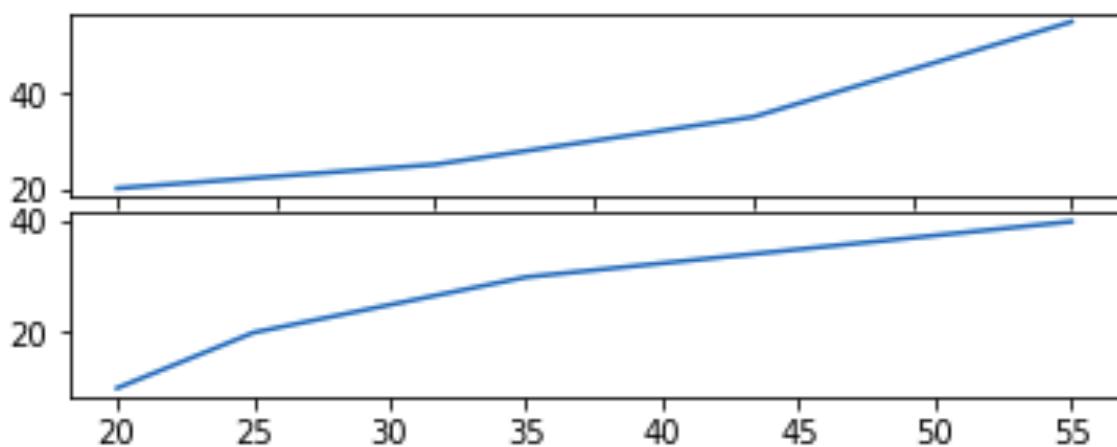
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# adding the subplots
axes1 = plt.subplot2grid(
(7, 1), (0, 0), rowspan = 2, colspan = 1)

axes2 = plt.subplot2grid(
(7, 1), (2, 0), rowspan = 2, colspan = 1)
```

```
axes2.plot(y, x)
```

### Output:



## Different types of Matplotlib Plots

Matplotlib supports a variety of plots including line charts, bar charts, histograms, scatter plots, etc. We will discuss the most commonly used charts in this article with the help of some good examples and will also see how to customize each plot.

**Note:** Some elements like axis, color are common to each plot whereas some elements are plot specific.

### Line Chart

**Line chart** is one of the basic plots and can be created using the [\*\*plot\(\)\*\*](#) function. It is used to represent a relationship between two data X and Y on a different axis.

### Syntax:

```
matplotlib.pyplot.plot(*args, scalex=True, scaley=True, data=None,  
                      **kwargs)
```

## Python3

```
import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y)

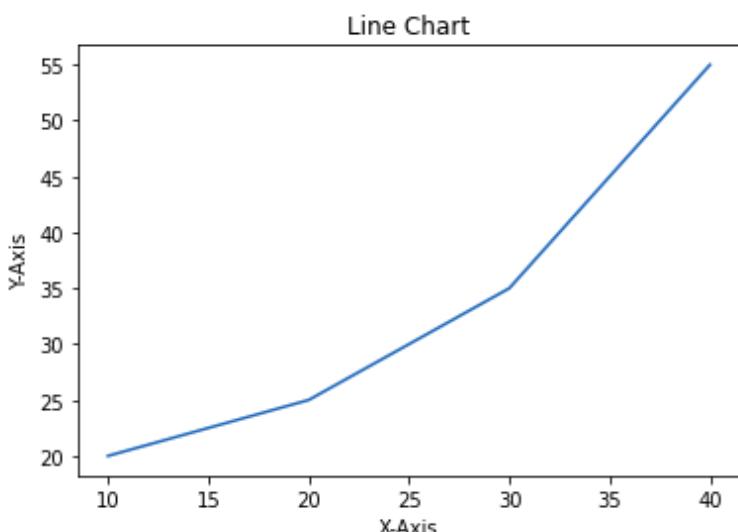
# Adding title to the plot
plt.title("Line Chart")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

# Adding label on the x-axis
plt.xlabel('X-Axis')

plt.show()
```

### Output:



Let's see how to customize the above-created line chart. We will be using the following properties –

- **color:** Changing the color of the line
- **linewidth:** Customizing the width of the line
- **marker:** For changing the style of actual plotted point
- **markersize:** For changing the size of the markers
- **linestyle:** For defining the style of the plotted line

## Different Linestyle available

Character	Definition
-	Solid line
--	Dashed line
-.	dash-dot line
:	Dotted line
.	Point marker
o	Circle marker
,	Pixel marker
v	triangle_down marker
^	triangle_up marker
<	triangle_left marker
>	triangle_right marker

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

Character	Definition
2	tri_up marker
3	tri_left marker
4	tri_right marker
s	square marker
p	pentagon marker
*	star marker
h	hexagon1 marker
H	hexagon2 marker
+	Plus marker
x	X marker
D	Diamond marker
d	thin_diamond marker
	vline marker
-	hline marker

### Example:

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

## Python3

```
import matplotlib.pyplot as plt

# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]

# plotting the data
plt.plot(x, y, color='green', linewidth=3, marker='o',
          markersize=15, linestyle='--')

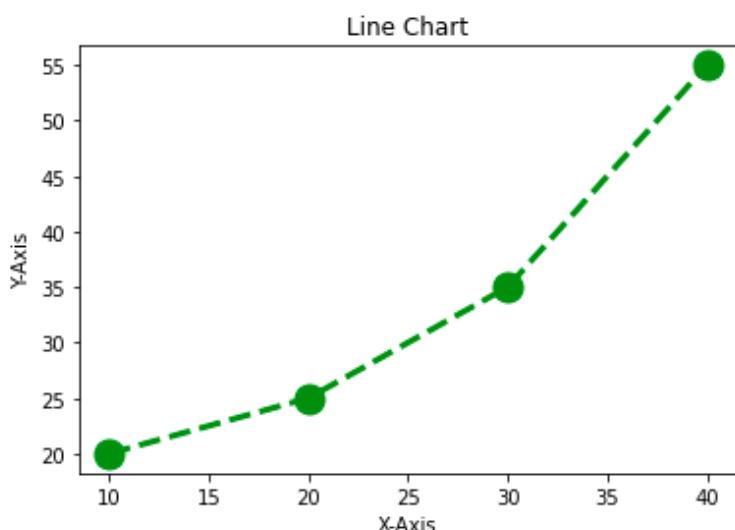
# Adding title to the plot
plt.title("Line Chart")

# Adding label on the y-axis
plt.ylabel('Y-Axis')

# Adding label on the x-axis
plt.xlabel('X-Axis')

plt.show()
```

### Output:



**Note:** For more information, refer [Line plot styles in Matplotlib](#)

## Bar Chart

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

rectangular bars with lengths and heights that is proportional to the values which they represent. The bar plots can be plotted horizontally or vertically. A bar chart describes the comparisons between the discrete categories. It can be created using the bar() method.

In the below example, we will use the tips dataset. Tips database is the record of the tip given by the customers in a restaurant for two and a half months in the early 1990s. It contains 6 columns as total\_bill, tip, sex, smoker, day, time, size.

### Example:

---

## Python3

```
import matplotlib.pyplot as plt
import pandas as pd

# Reading the tips.csv file
data = pd.read_csv('tips.csv')

# initializing the data
x = data['day']
y = data['total_bill']

# plotting the data
plt.bar(x, y)

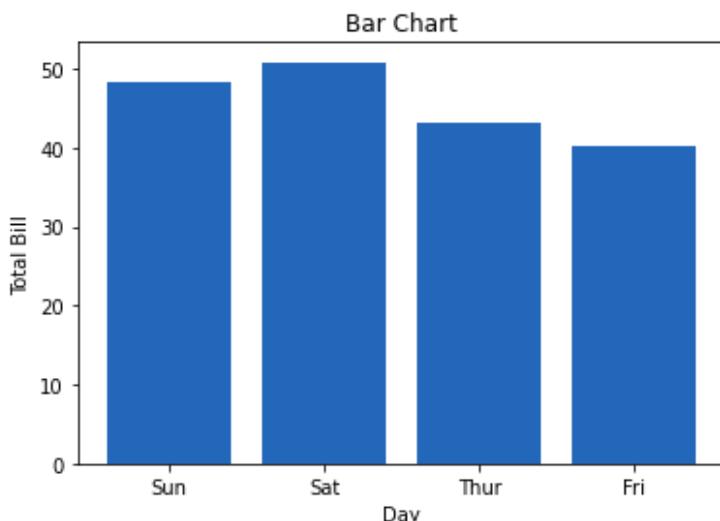
# Adding title to the plot
plt.title("Tips Dataset")

# Adding label on the y-axis
plt.ylabel('Total Bill')

# Adding label on the x-axis
plt.xlabel('Day')

plt.show()
```

### Output:



Customization that is available for the Bar Chart –

- **color:** For the bar faces
- **edgecolor:** Color of edges of the bar
- **linewidth:** Width of the bar edges
- **width:** Width of the bar

**Example:**

## Python3

```
import matplotlib.pyplot as plt
import pandas as pd

# Reading the tips.csv file
data = pd.read_csv('tips.csv')

# initializing the data
x = data['day']
y = data['total_bill']

# plotting the data
plt.bar(x, y, color='green', edgecolor='blue',
        linewidth=2)

# Adding title to the plot
plt.title("Tips Dataset")

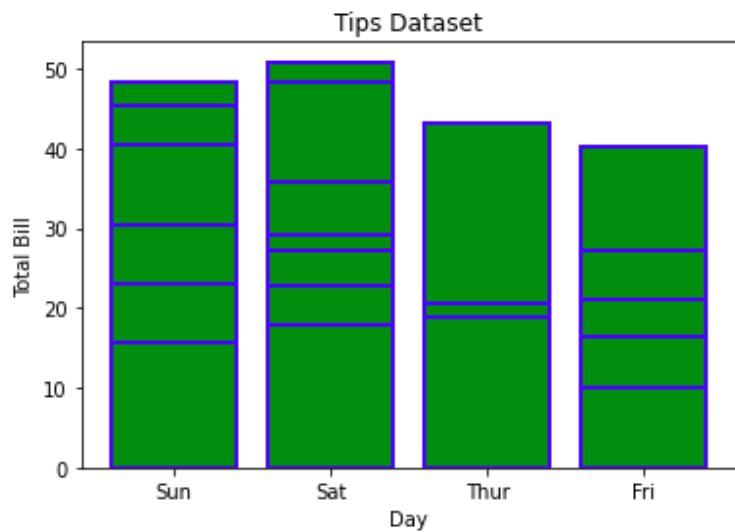
# Adding label on the y-axis
plt.ylabel('Total Bill')
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
plt.show()
```

## Output:



**Note:** The lines in between the bars refer to the different values in the Y-axis of the particular value of the X-axis.

## Histogram

A histogram is basically used to represent data provided in a form of some groups. It is a type of bar plot where the X-axis represents the bin ranges while the Y-axis gives information about frequency. The hist() function is used to compute and create histogram of x.

### Syntax:

```
matplotlib.pyplot.hist(x, bins=None, range=None, density=False,  
weights=None, cumulative=False, bottom=None, histtype='bar',  
align='mid', orientation='vertical', rwidth=None, log=False,  
color=None, label=None, stacked=False, *, data=None, **kwargs)
```

### Example:

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
import matplotlib.pyplot as plt
import pandas as pd

# Reading the tips.csv file
data = pd.read_csv('tips.csv')

# initializing the data
x = data['total_bill']

# plotting the data
plt.hist(x)

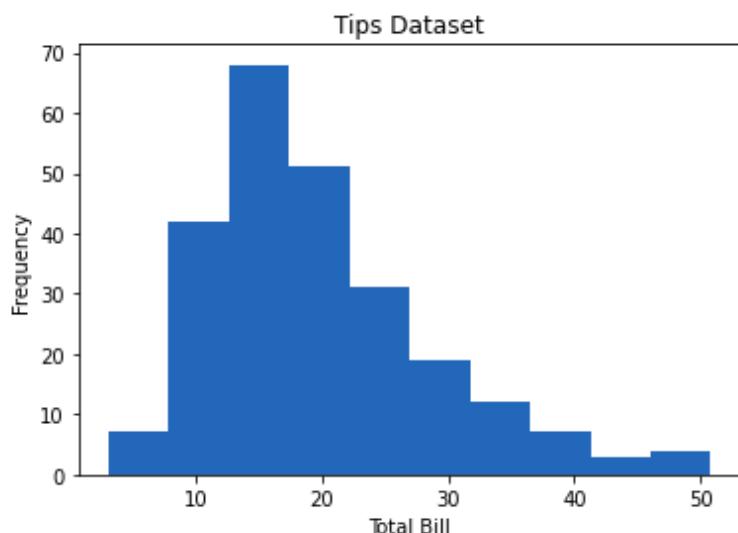
# Adding title to the plot
plt.title("Tips Dataset")

# Adding label on the y-axis
plt.ylabel('Frequency')

# Adding label on the x-axis
plt.xlabel('Total Bill')

plt.show()
```

## Output:



Customization that is available for the Histogram –

- **bins:** Number of equal-width bins
- **color:** For changing the face color
- **edgecolor:** Color of the edges

[More Details](#)

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

**Example:****Python3**

```
import matplotlib.pyplot as plt
import pandas as pd

# Reading the tips.csv file
data = pd.read_csv('tips.csv')

# initializing the data
x = data['total_bill']

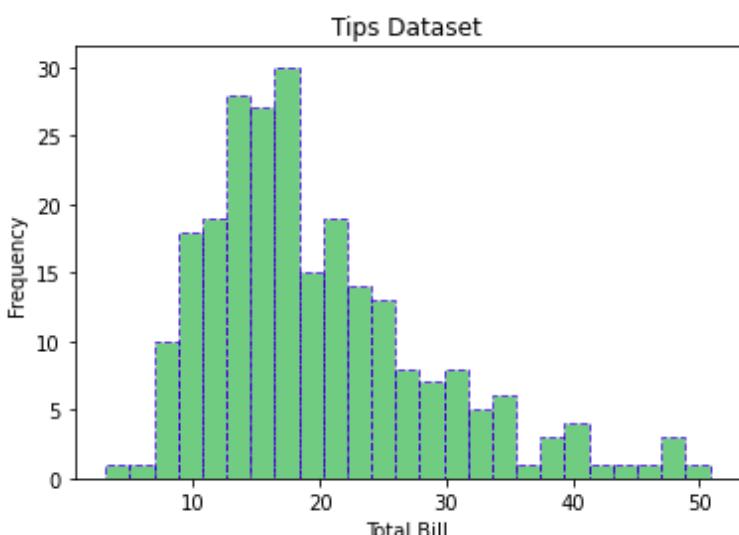
# plotting the data
plt.hist(x, bins=25, color='green', edgecolor='blue',
         linestyle='--', alpha=0.5)

# Adding title to the plot
plt.title("Tips Dataset")

# Adding label on the y-axis
plt.ylabel('Frequency')

# Adding label on the x-axis
plt.xlabel('Total Bill')

plt.show()
```

**Output:**

**Scatter plots** are used to observe relationships between variables. The [scatter\(\)](#) method in the matplotlib library is used to draw a scatter plot.

### Syntax:

```
matplotlib.pyplot.scatter(x_axis_data, y_axis_data, s=None, c=None,  
marker=None, cmap=None, vmin=None, vmax=None, alpha=None,  
linewidths=None, edgecolors=None)
```

### Example:

---

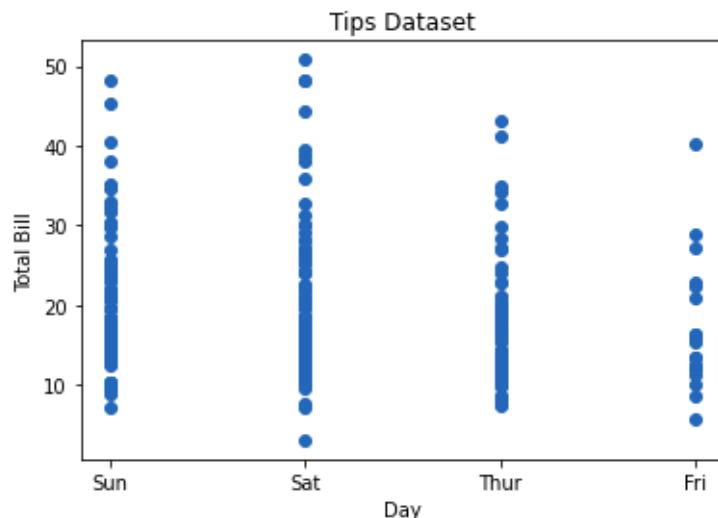
## Python3

```
import matplotlib.pyplot as plt  
import pandas as pd  
  
# Reading the tips.csv file  
data = pd.read_csv('tips.csv')  
  
# initializing the data  
x = data['day']  
y = data['total_bill']  
  
# plotting the data  
plt.scatter(x, y)  
  
# Adding title to the plot  
plt.title("Tips Dataset")  
  
# Adding label on the y-axis  
plt.ylabel('Total Bill')  
  
# Adding label on the x-axis  
plt.xlabel('Day')  
  
plt.show()
```

### Output:

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**



Customizations that are available for the scatter plot are –

- **s:** marker size (can be scalar or array of size equal to size of x or y)
- **c:** color of sequence of colors for markers
- **marker:** marker style
- **linewidths:** width of marker border
- **edgecolor:** marker border color
- **alpha:** blending value, between 0 (transparent) and 1 (opaque)

## Python3

```
import matplotlib.pyplot as plt
import pandas as pd

# Reading the tips.csv file
data = pd.read_csv('tips.csv')

# initializing the data
x = data['day']
y = data['total_bill']

# plotting the data
plt.scatter(x, y, c=data['size'], s=data['total_bill'],
            marker='D', alpha=0.5)

# Adding title to the plot
plt.title("Tips Dataset")

# Adding label on the y-axis
```

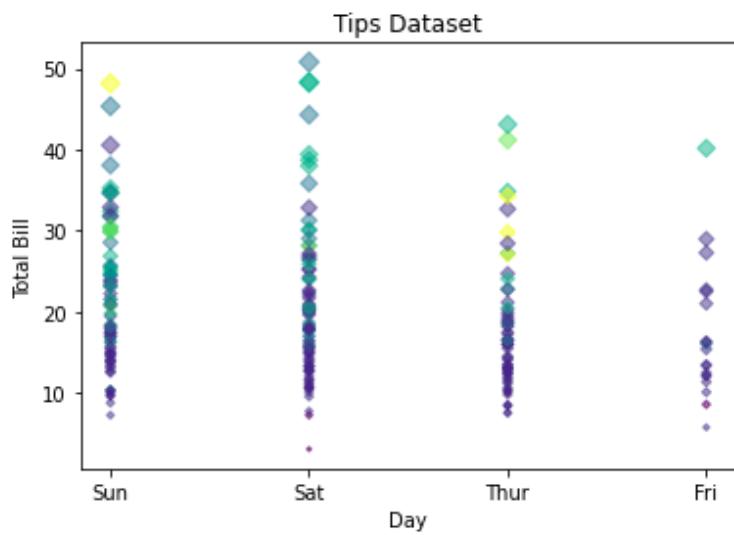
We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
# Adding label on the x-axis
plt.xlabel('Day')

plt.show()
```

## Output:



## Pie Chart

**Pie chart** is a circular chart used to display only one series of data. The area of slices of the pie represents the percentage of the parts of the data. The slices of pie are called wedges. It can be created using the `pie()` method.

### Syntax:

```
matplotlib.pyplot.pie(data, explode=None, labels=None,
                      colors=None, autopct=None, shadow=False)
```

---

### Example:

## Python3

```
import matplotlib.pyplot as plt
import pandas as pd
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

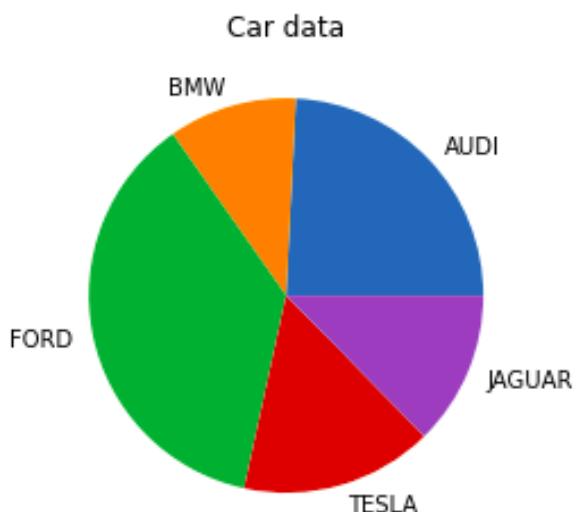
```
# initializing the data
cars = ['AUDI', 'BMW', 'FORD',
        'TESLA', 'JAGUAR',]
data = [23, 10, 35, 15, 12]

# plotting the data
plt.pie(data, labels=cars)

# Adding title to the plot
plt.title("Car data")

plt.show()
```

## Output:



Customizations that are available for the Pie chart are –

- **explode:** Moving the wedges of the plot
- **autopct:** Label the wedge with their numerical value.
- **color:** Attribute is used to provide color to the wedges.
- **shadow:** Used to create shadow of wedge.

## Example:

---

### Python3

```
import matplotlib.pyplot as plt
import pandas as pd
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

```
data = pd.read_csv('tips.csv')

# initializing the data
cars = ['AUDI', 'BMW', 'FORD',
        'TESLA', 'JAGUAR',]
data = [23, 13, 35, 15, 12]

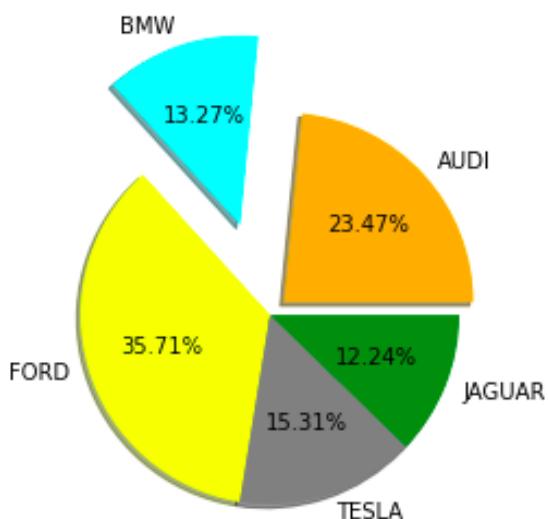
explode = [0.1, 0.5, 0, 0, 0]

colors = ( "orange", "cyan", "yellow",
           "grey", "green",)

# plotting the data
plt.pie(data, labels=cars, explode=explode, autopct='%.2f%%',
         colors=colors, shadow=True)

plt.show()
```

## Output:



## Saving a Plot

For [saving a plot](#) in a file on storage disk, `savefig()` method is used. A file can be saved in many formats like .png, .jpg, .pdf, etc.

## Syntax:

```
pyplot.savefig(fname, dpi=None, facecolor='w', edgecolor='w',
               orientation='portrait', papertype=None, format=None,
               transparent=False, bbox_inches=None, pad_inches=0.1,
               frameon=None, metadata=None)
```

## Example:

### Python3

```
import matplotlib.pyplot as plt

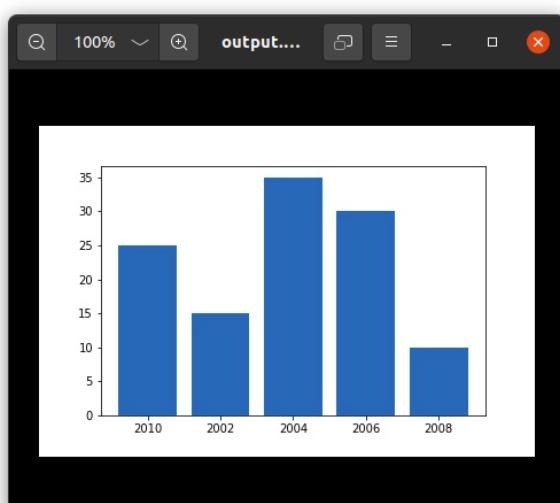
# Creating data
year = ['2010', '2002', '2004', '2006', '2008']
production = [25, 15, 35, 30, 10]

# Plotting barchart
plt.bar(year, production)

# Saving the figure.
plt.savefig("output.jpg")

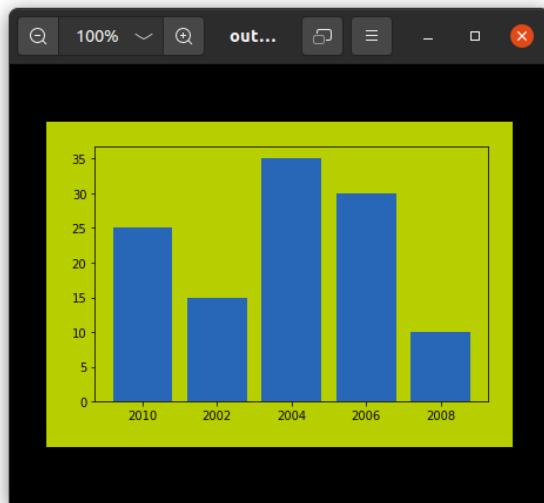
# Saving figure by changing parameter values
plt.savefig("output1", facecolor='y', bbox_inches="tight",
            pad_inches=0.3, transparent=True)
```

## Output:



We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**



Last Updated : 30 Sep, 2022

10

## Similar Reads

1. [COVID-19 Data Visualization using matplotlib in Python](#)

---
2. [Insertion Sort Visualization using Matplotlib in Python](#)

---
3. [Visualization of Quick sort using Matplotlib](#)

---
4. [Visualization of Merge sort using Matplotlib](#)

---
5. [Python - Data visualization using Bokeh](#)

---
6. [Data Visualization Using Chartjs and Django](#)

---
7. [Interactive visualization of data using Bokeh](#)

---
8. [Animated Data Visualization using Plotly Express](#)

---
9. [Data Visualization using Turicreate in Python](#)

---
10. [Visualization and Prediction of Crop Production data using Python](#)

## Related Tutorials

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy & Privacy Policy](#).

**Got It !**

---

2. Flask Tutorial

---



---

3. Natural Language Processing (NLP) Tutorial

---



---

4. Data Science for Beginners

---



---

5. Data Science With Python Tutorial

Previous [Company](#)

[Explore](#) Next

## Data Visualization with Python

Careers

In Media

Contact Us

## isocalendar() Function Of Datetime.date Class In Python

PDF Related

Python Backend LIVE

Android App Development

DevOps LIVE

DSA in JavaScript

## Article Contributed By:



Privacy Policy

[nikhilaggarwal3](#)

Copyright Policy

[nikhilaggarwal3](#)

Third-Party Copyright Notices

## Vote for difficulty with us

Current difficulty : [Hard](#)

### Languages

Easy

Normal

Medium

Hard

Expert

### Data Structures

Array

String

Linked List

Stack

Queue

Tree

Graph

Improved By :

C++ [sumitgumber28](#), [varshagumber28](#), [sagartomar9927](#)

GoLang

Article Tags :

[Python-matplotlib](#), [Python](#)

SQL

Practice Tags :

R Language

Java

Python

Android Tutorial

### Web Development

HTML

CSS

JavaScript

Bootstrap

ReactJS

AngularJS

Improve Article

[Algorithms](#)

Report Issue

Sorting

Searching

Greedy

Dynamic Programming

Pattern Searching

Recursion

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

## Computer Science

- GATE CS Notes
- Operating Systems
- Computer Network
- Database Management System
- Software Engineering
- Digital Logic Design
- Engineering Maths

## Python

- Python Programming Examples
- Django Tutorial
- Python Projects
- Python Tkinter
- OpenCV Python Tutorial
- Python Interview Question

## Data Science & ML

- Data Science With Python
- Data Science For Beginner
- Machine Learning Tutorial
- Maths For Machine Learning
- Pandas Tutorial
- NumPy Tutorial
- NLP Tutorial
- Deep Learning Tutorial

## DevOps

- Git
- AWS
- Docker
- Kubernetes
- Azure
- GCP

## Competitive Programming

- Top DSA for CP
- Top 50 Tree Problems
- Top 50 Graph Problems
- Top 50 Array Problems
- Top 50 String Problems
- Top 50 DP Problems
- Top 15 Websites for CP

## System Design

- What is System Design
- Monolithic and Distributed SD
- Scalability in SD
- Databases in SD
- High Level Design or HLD
- Low Level Design or LLD
- Top SD Interview Questions

## Interview Corner

- Company Preparation
- Preparation for SDE
- Company Interview Corner
- Experienced Interview
- Internship Interview
- Competitive Programming
- Aptitude

## GfG School

- CBSE Notes for Class 8
- CBSE Notes for Class 9
- CBSE Notes for Class 10
- CBSE Notes for Class 11
- CBSE Notes for Class 12

## Commerce

- Accountancy
- Business Studies
- Microeconomics
- Macroeconomics
- Statistics for Economics
- Indian Economic Development

## UPSC

- Polity Notes
- Geography Notes
- History Notes
- Science and Technology Notes
- Economics Notes
- Important Topics in Ethics
- UPSC Previous Year Papers

## SSC/ BANKING

@geeksforgeeks , Some rights reserved

**SSC CGL Syllabus** **Please Login To Continue** ×

[SBI Clerk Syllabus](#) [IBPS PO Syllabus](#) [Sign Up](#)

SBI Clerk

IBPS PO

Remember me

[IBPS Clerk Syllabus](#) [Aptitude Questions](#) [Sign In](#)

**SSC CGL Practice Papers**

[Sign Up](#)

or



### Why Create an Account?

By creating this account you agree to our [Privacy Policy](#) & [Cookie Policy](#)

## Write & Earn

- [Write an Article](#)
- [Improve an Article](#)
- [Pick Topics to Write](#)
- [Write Interview Experience](#)
- [Internships](#)
- [Forgot Password](#)
- [Video Internship](#)

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**

[Back to Login](#)[Reset Password](#)

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#).

**Got It !**