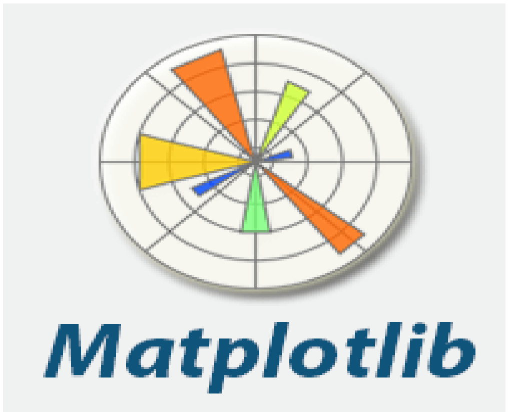
303: Database handling using Python

Unit-5: Data Visualization using dataframe:

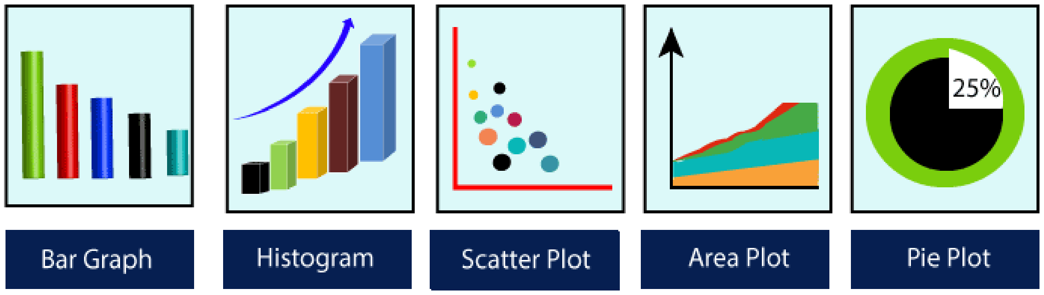


Human minds are more adaptive for the visual representation of data rather than textual data. We can easily understand things when they are visualized. It is better to represent the data through the graph where we can analyze the data more efficiently and make the specific decision according to data analysis. Before learning the matplotlib, we need to understand data visualization and why data visualization is important.

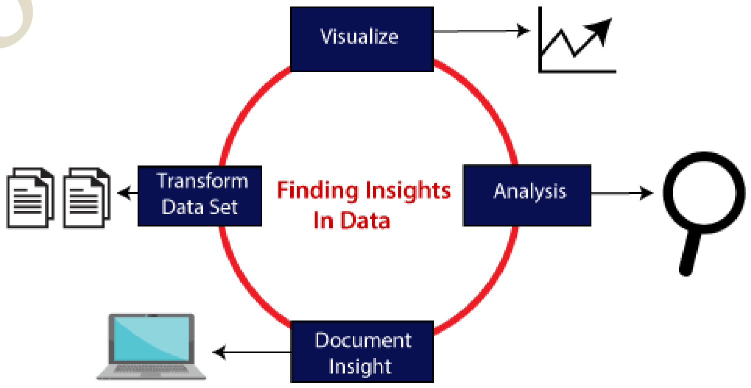
Data Visualization

Graphics provides an excellent approach for exploring the data, which is essential for presenting results. Data visualization is a new term. It expresses the idea that involves more than just representing data in the graphical form.
This can be very helpful when discovering and getting to know a dataset and can help with classifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts. The static does indeed focus on quantitative description and estimations of data. It provides an important set of tools for gaining a qualitative understanding.

There are five key plots that are used for data visualization.



There are five phases which are essential to make the decision for the organization:



- **Visualize:** We analyze the raw data, which means it makes complex data more accessible, understandable, and more usable. Tabular data representation is used where the user will look up a specific measurement, while the chart of several types is used to show patterns or relationships in the data for one or more variables.
- **Analysis:** Data analysis is defined as cleaning, inspecting, transforming, and modeling data to derive useful information. Whenever we make a decision for the business or in daily life, is by past experience. **What will happen to choose a particular decision**, it is nothing but analyzing our past. That may be affected in the future, so the proper analysis is necessary for better decisions for any business or organization.
- **Document Insight:** Document insight is the process where the useful data or information is organized in the document in the standard format.
- **Transform Data Set:** Standard data is used to make the decision more effectively.

❖ **Why need data visualization?**

Data visualization can perform below tasks:

1. It identifies areas that need improvement and attention.
2. It clarifies the factors.
3. It helps to understand which product to place where.
4. Predict sales volumes.

- Matplotlib is a Python library which is defined as a multi-platform data visualization library built on Numpy array. It can be used in python scripts, shell, web application, and other graphical user interface toolkit.
- The **John D. Hunter** originally conceived the matplotlib in 2002. It has an active development community and is distributed under a BSD-style license. Its first version was released in 2003, and the latest version 3.1.1 is released on 1 July 2019.
- Matplotlib 2.0.x supports Python versions 2.7 to 3.6 till 23 June 2007. Python3 support started with Matplotlib 1.2. Matplotlib 1.4 is the last version that supports Python 2.6.

The General Concept of Matplotlib:

A Matplotlib figure can be categorized into various parts as below:

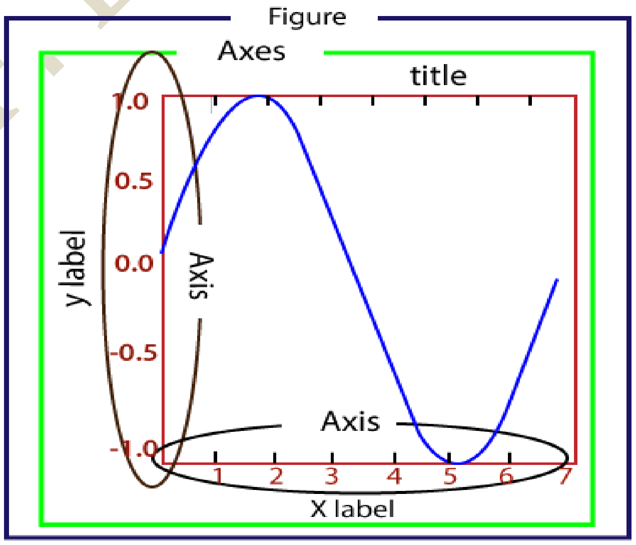


Figure: It is a whole figure which may hold one or more axes (plots). We can think of a Figure as a canvas that holds plots.

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.

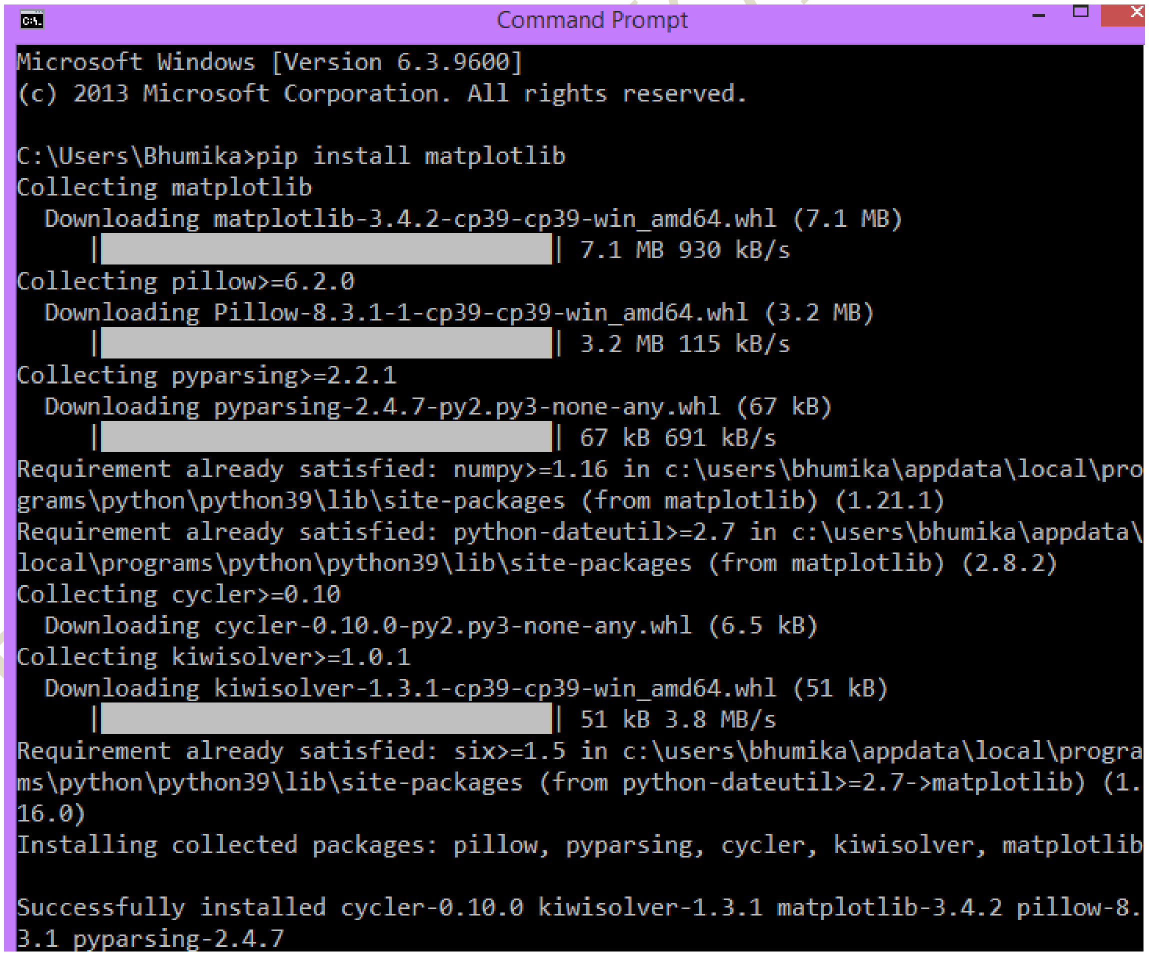
Artist: An artist is the all which we see on the graph like Text objects, Line2D objects, and collection objects. Most Artists are tied to Axes.

5.1 importing matplotlib.pyplot and plotting: (only two dimensional Plots)

Installing Matplotlib

Before start working with the Matplotlib or its plotting functions first, it needs to be installed. The installation of matplotlib is dependent on the distribution that is installed on your computer. These installation methods are following:
Go to Command Prompt and type following command:

```
pip install matplotlib
```



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Bhumika>pip install matplotlib
Collecting matplotlib
  Downloading matplotlib-3.4.2-cp39-cp39-win_amd64.whl (7.1 MB)
     |████████████████████████████████| 7.1 MB 930 kB/s
Collecting pillow>=6.2.0
  Downloading Pillow-8.3.1-1-cp39-cp39-win_amd64.whl (3.2 MB)
     |████████████████████████████████| 3.2 MB 115 kB/s
Collecting pyparsing>=2.2.1
  Downloading pyparsing-2.4.7-py2.py3-none-any.whl (67 kB)
     |████████████████████████████████| 67 kB 691 kB/s
Requirement already satisfied: numpy>=1.16 in c:\users\bhumika\appdata\local\pro
grams\python\python39\lib\site-packages (from matplotlib) (1.21.1)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\bhumika\appdata\
local\programs\python\python39\lib\site-packages (from matplotlib) (2.8.2)
Collecting cycler>=0.10
  Downloading cycler-0.10.0-py2.py3-none-any.whl (6.5 kB)
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.3.1-cp39-cp39-win_amd64.whl (51 kB)
     |████████████████████████████████| 51 kB 3.8 MB/s
Requirement already satisfied: six>=1.5 in c:\users\bhumika\appdata\local\progra
ms\python\python39\lib\site-packages (from python-dateutil>=2.7->matplotlib) (1.
16.0)
Installing collected packages: pillow, pyparsing, cycler, kiwisolver, matplotlib

Successfully installed cycler-0.10.0 kiwisolver-1.3.1 matplotlib-3.4.2 pillow-8.
3.1 pyparsing-2.4.7
```

❖ Verify the Installation

To verify that matplotlib is installed properly or not, type the following command includes calling .__version __ in the terminal.

```
>>> import matplotlib
>>> print(matplotlib.__version__)
3.4.2
>>>
```

5.1.1 range() , subplot() , legend(), columns(), len() functions.

Matplotlib.Pyplot

Most of the Matplotlib utilities lies under the pyplot submodule, and are usually imported under the plt alias:

import matplotlib.pyplot as plt

Now the Pyplot package can be referred to as **plt**.

Matplotlib Plotting

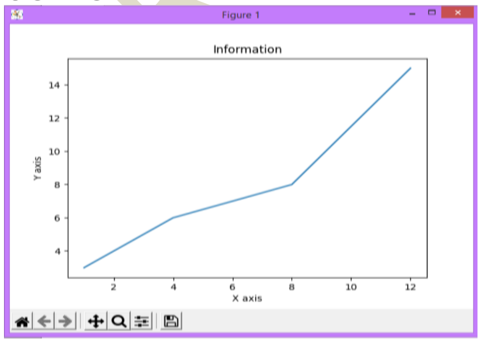
Plotting x and y points

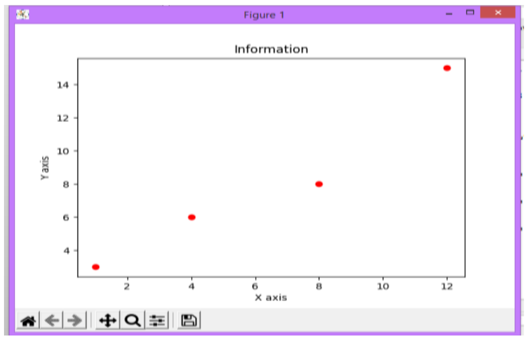
- The plot() function is used to draw points (markers) in a diagram.
- By default, the plot() function draws a line from point to point.
- The function takes parameters for specifying points in the diagram.
- Parameter 1 is an array containing the points on the x-axis.
- Parameter 2 is an array containing the points on the y-axis.

Example: Draw a line in a diagram from given position.

```
#plot.py
import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([1, 8, 12])
ypoints = np.array([3, 10, 15])

plt.plot(xpoints, ypoints)          #Using for the blue marker with default
#plt.plot(xpoints, ypoints,'ro')    #Red circle
#plt.plot(xpoints, ypoints,'-g')    #Green solid line
#plt.plot(xpoints, ypoints,'--')

plt.ylabel('Y axis')
plt.xlabel('X axis')
plt.title('Information')
plt.show()
```

OUTPUT:





The x-axis is the horizontal axis. The y-axis is the vertical axis.

matplotlib.pyplot.legend()

A legend is an area describing the elements of the graph. In the matplotlib library, there's a function called legend() which is used to Place a legend on the axes.

Syntax:

```
matplotlib.pyplot.legend(*args, **kwargs)
```

Place a legend on the Axes.

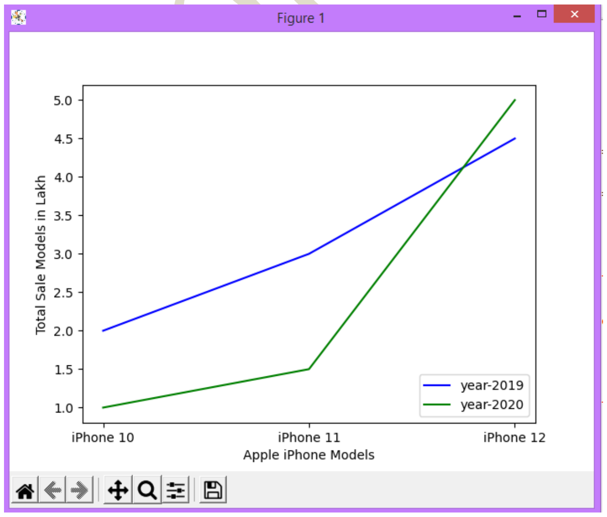
Call signatures:

```
legend()
legend(labels)
legend(handles, labels)
```

Example : Line Graph with legend()

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

# Y-axis Sale values in Lakh
y2019 = [2, 3, 4.5]
# Y-axis Sale values in Lakh
y2020 = [1, 1.5, 5]

x=['iPhone 10','iPhone 11','iPhone 12']

# Function to plot line graph
plt.plot(x,y2019,c='blue')
plt.plot(x,y2020,c='green')

# Function add a legend
plt.legend(["year-2019", "year-2020"], loc ="lower right")

plt.xlabel('Apple iPhone Models')
plt.ylabel('Total Sale Models in Lakh')

plt.show()
```

OUTPUT:



What is subplot() ?
The Matplotlib subplot() function is defined as to plot two or more plots in one figure. We can use this method to separate two graphs which plotted in the same axis Matplotlib supports all kinds of subplots, including 2x1 vertical, 2x1 horizontal, or a 2x2 grid.

It accepts the three arguments: they are nrows, ncols, and index. It denote the number of rows, number of columns and the index.

Syntax:
The subplot() function can be called in the following way:

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

Parameters:

***args:**
Three separate integers or three-digit integer describes the position of the subplot. If the three integers are nrows, ncols, and index in order, the subplot will take the index position on a grid with nrows row and ncol column.

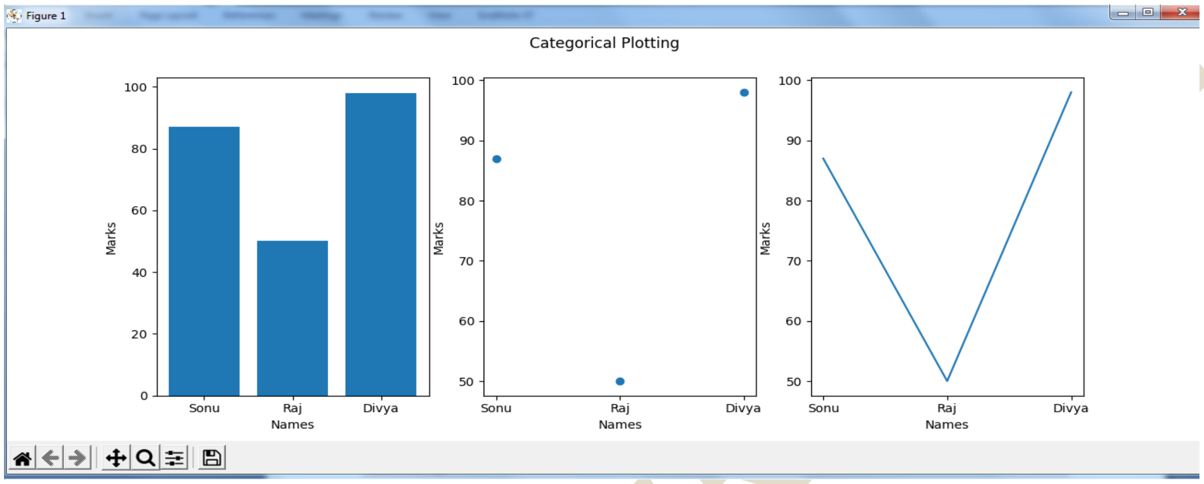
The argument pos are a three-digit integer, where the first digit is denoted the number of rows, the second digit denoted the number of columns, and the third represents the index of the subplot. For example, subplot (1, 3, 2) is the same as the subplot (132).
Note: Passed integer must be less than 10.

****kwargs**
The subplot() function also accepts the keyword arguments for the returned axes base class.

Matplotlib allows us to pass categorical variables directly to many plotting functions: consider the following example.
Example of subplot(): (subplot1.py)

```
from matplotlib import pyplot as plt

names = ['Sonu', 'Raj', 'Divya']
marks= [87,50,98]

#names = ['Sonu', 'Raj', 'Divya','mohan']
#marks= [87,50,98,80]

plt.figure(figsize=(13,3))
plt.subplot(131)
plt.bar(names, marks)
plt.xlabel('Names')
plt.ylabel('Marks')

plt.subplot(132)
plt.scatter(names, marks)
plt.xlabel('Names')
plt.ylabel('Marks')

plt.subplot(133)
plt.plot(names, marks)
plt.xlabel('Names')
plt.ylabel('Marks')

plt.suptitle('Categorical Plotting')
plt.show()
```

OUTPUT:



The DataFrame class of Python pandas library is a two dimensional data container. The DataFrame class has several categories of methods defined on it to help analyze its data. The methods include statistical methods, mathematical methods and other subset selection methods. Any data analysis without diagrams is not so powerful. Hence pandas DataFrame class is equipped with the capability of plotting diagrams as well.

The pandas library uses matplotlib as the plotting backend. In future versions of pandas this may become a configurable pandas option. To use the plotting features of DataFrame and other pandas classes matplotlib is a soft dependency. Pandas do not require matplotlib otherwise.

5.2 scatter plot: concept of Scatter plot, set title, xlabel and ylabel)

- ➢ A scatter plot is a diagram drawn between two distributions of variables X and Y on a two dimensional plane.
- ➢ Scatter plot is used as an initial screening tool while analyzing two variables for any relationship (linear, non-linear, inverse relationships) that may exist between them.
- ➢ A scatter plot is used only as an initial tool in the process of finding any relationship between two variables. Even if a relationship is found between two variables using scatter plot, it may not be true that one variable influences another variable. To establish relationship between two variables tools like correlation can be used.

Plotting a scatter plot using Pandas DataFrame:

- ➢ The pandas DataFrame class in Python has a member plot.

- Invoking the scatter() method on the plot member draws a scatter plot between two given columns of a pandas DataFrame.
- A pandas DataFrame can have several columns. Any two columns can be chosen as X and Y parameters for the scatter() method.

Syntax:
DataFrame.plot.scatter(x, y, s=None, c=None, **kwargs)

Parameters
x : int or str
The column name or column position is to be used as horizontal coordinates for each point.

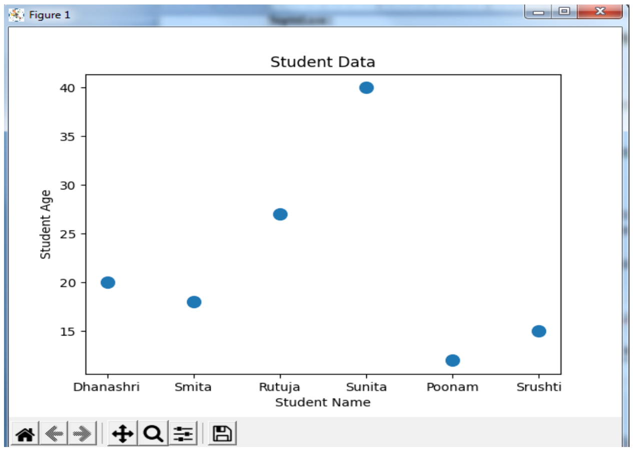
y : int or str
The column name or column position is to be used as vertical coordinates for each point.

s : str, scalar or array-like, optional
It is size of each point. Possible values are:
- A string with the name of the column to be used for marker's size.
- A single scalar so all points have the same size.

c : str, int or array-like, optional
It is color of each dot. Possible values are: A single color string referred to by name, RGB or RGBA code, for instance 'red' or '#a98d19'.

****kwargs**
Keyword arguments to pass on to DataFrame.plot().
Example 1: Program to draw scatter plot using Dataframe.plot(scatter1.py)

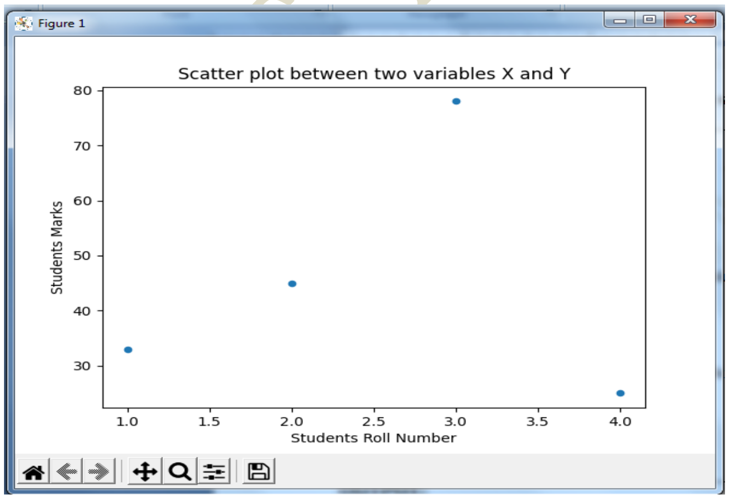
```
import pandas as pd                    #Import libraries
import matplotlib.pyplot as plot

data={'Name':['Dhanashri', 'Smita', 'Rutuja','Sunita', 'Poonam', 'Srushti'],
        'Age':[20, 18, 27, 40, 12, 15]}
# Load data into DataFrame
df = pd.DataFrame(data = data)

# Draw a scatter plot
df.plot.scatter(x = 'Name', y = 'Age', s = 100)
#df.plot.scatter(x = 'Name', y = 'Age', s = 150, c = 'red')

plot.xlabel('Student Name')
plot.ylabel('Student Age')
plot.title('Student Data')
plot.show()
```
OUTPUT:



Example 2: Python program to draw a scatter plot for two columns of a pandas DataFrame (SCATTER_COL.PY)

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

# List of tuples
data = [(1, 33), (2, 45), (3, 78), (4, 25)]

# Load data into pandas DataFrame
dataFrame = pd.DataFrame(data=data, columns=['RollNo','Marks'])
# print(dataFrame)

# Draw a scatter plot
dataFrame.plot.scatter(x='RollNo', y='Marks', title="Scatter plot between two variables X
and Y")
plot.xlabel('Students Roll Number')
plot.ylabel('Students Marks')
plot.show()
```

OUTPUT:



5.3 Line chart: concept of line plot: plot(), set_title(), legend()

Line Chart:

- A line chart plots a set of (x, y) values in a two-dimensional plane and connects those data points through straight lines.
- A line chart is one of the most commonly used charts to understand the relationship, trend of one variable with another.

Drawing a Line chart using pandas DataFrame in Python:

- The DataFrame class has a plot member through which several graphs for visualization can be plotted. A line chart or line graph is one among them.
- Calling the line() method on the plot instance draws a line chart. If the column name for X-axis is not specified, the method takes the index of the column as the X-axis, which is of the pattern 0, 1, 2, 3 and so on.
- After drawing the X-axis from the index of the DataFrame or using the specified column, the subsequent numeric columns are plotted as lines against the X-axis.

Syntax:

DataFrame.plot.line(x, y)

The table below explains the main parameters of the method:

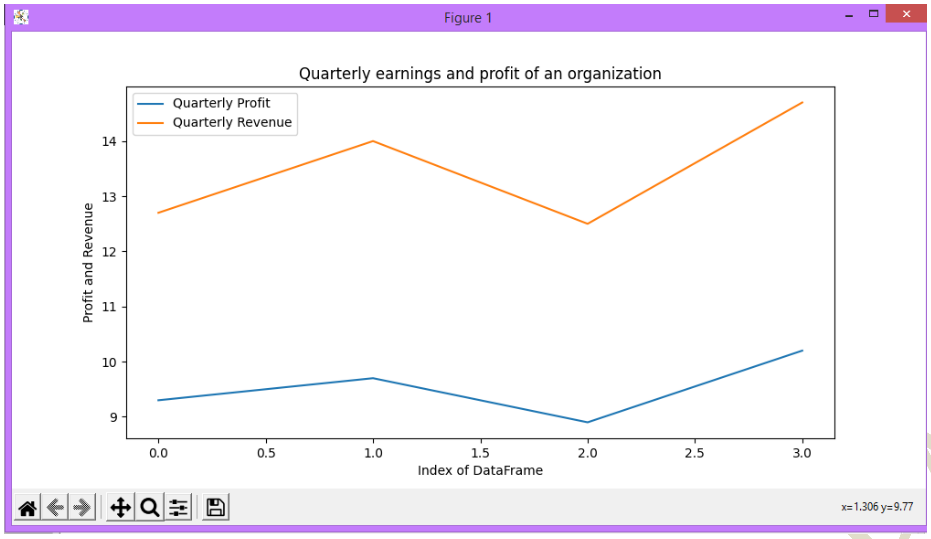
Parameter	**Value**	**Default Value**	**Use**
x	Int or string	DataFrame indices	Set the values to be represented in the x-axis.
y	Int or string	Remaining columns in DataFrame	Set the values to be represented in the y-axis.

Additional parameters include color (specifies the color of the line), title (specifies the title of the plot), and kind (specifies which type of plot to use). The default variable for the "kind" parameter of this method is 'line'. Therefore, you don't have to set it in order to create a line plot.

Example1: Python program to plot a line chart (X-axis is not specified)(LINE_1.py)

```
#Example Python program to plot a line chart for a pandas DataFrame
import pandas as pd
import matplotlib.pyplot as plot

#Earnings data for 4 quarters as a Python Dictionary
earningsData = {"Quarterly Profit": [9.3, 9.7, 8.9, 10.2],
          "Quarterly Revenue": [12.7, 14.0, 12.5, 14.7]}

df = pd.DataFrame(data=earningsData)
#print(df)
#plot.figure(figsize=(10,3))

# Draw a line chart
df.plot.line(figsize=(10,5),title="Quarterly earnings and profit of an organization")
plot.xlabel('Index of DataFrame')
plot.ylabel('Profit and Revenue')
plot.show()
```

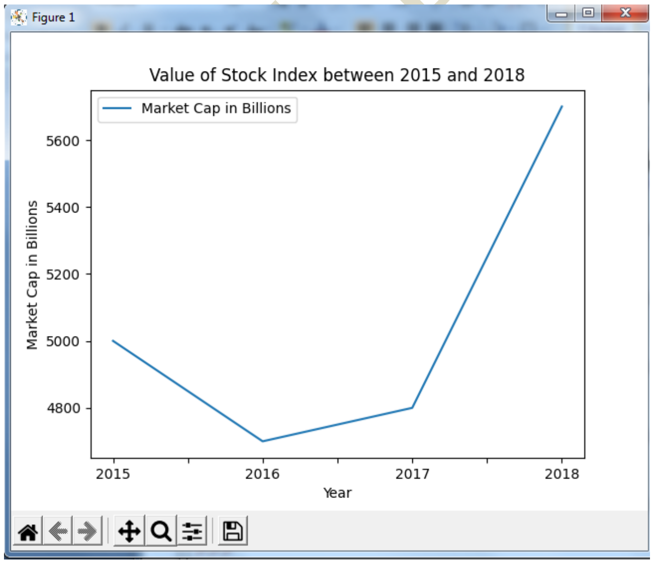
OUTPUT:



Example2: Python program to plot a line chart for a pandas DataFrame(LINE_2.py)

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

stockIndex = {"Year":  ["2015", "2016", "2017", "2018"],
             "Index": [35000, 33000, 34000, 36000],
             "Market Cap in Billions":[5000, 4700, 4800, 5700] }
 df = pd.DataFrame(data=stockIndex)
#print(df)
df.plot.line(x="Year", y="Market Cap in Billions")
plot.title('Value of Stock Index between 2015 and 2018')
plot.xlabel('Year')
plot.ylabel('Market Cap in Billions')
plot.show()
```

OUTPUT:



5.4 histogram chart: Concepts of histogram hist(),set title, xlabel and ylabel

histogram chart :

- A histogram is a representation of the distribution of data. This function groups the values of all given Series in the DataFrame into bins and draws all bins in one matplotlib.axes.Axes. This is useful when the DataFrame's Series are in a similar scale.
- A histogram is a frequency plot. Each interval data is marked with different heights based on the frequency of the data in that interval. Higher the frequency greater the height marked for that frequency.
- Calling the hist() method on the plot member of a DataFrame instance draws the histogram for each column present in the DataFrame.
- Number of bins or intervals of the histogram is specified for all the columns of the DataFrame combined.
- Each histogram will be drawn in an overlapping (if any) fashion using the alpha-blending value specified.

Syntax:

```
DataFrame.plot.hist(by=None, bins=10, **kwargs)
```

Parameters

by: str or sequence, optional
Column in the DataFrame to group by.

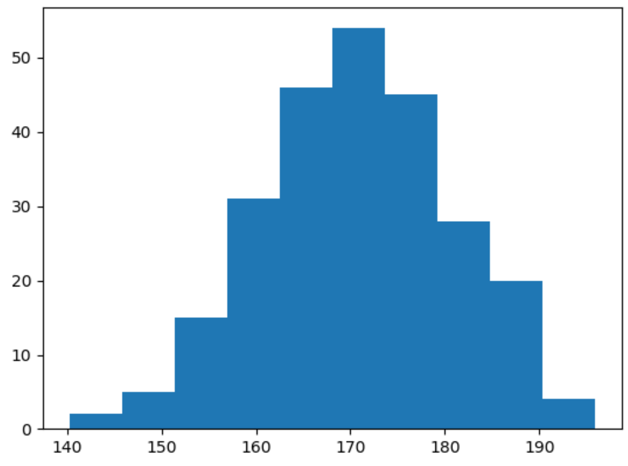
bins: int, default is 10
Number of histogram bins to be used.

****kwargs**
Additional keyword arguments are documented in DataFrame.plot().

Returns: class:matplotlib.AxesSubplot
Return a histogram plot.

Using rwidth parameter : The relative width of the bars as a fraction of the bin width

Example: Say you ask for the height of 250 people, you might end up with a histogram like this

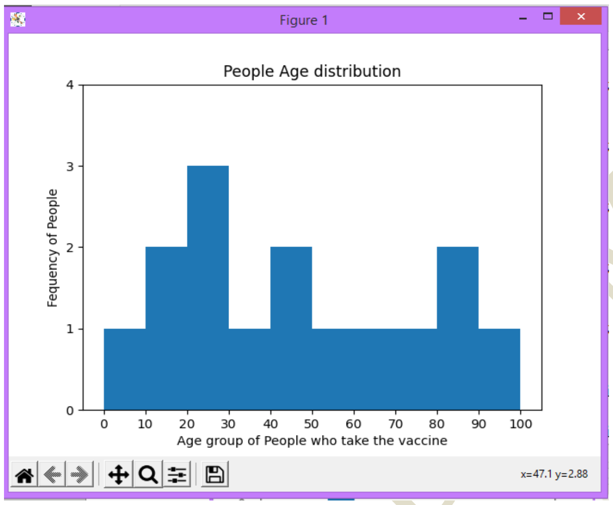


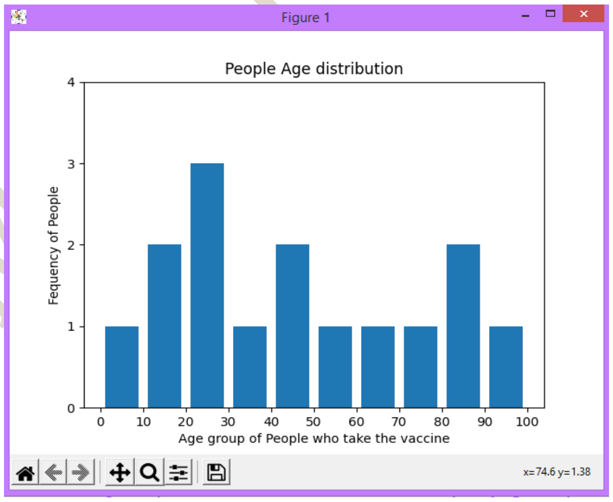
You can read from the histogram that there are approximately:
2 people from 140 to 145cm
5 people from 145 to 150cm
15 people from 151 to 156cm
31 people from 157 to 162cm
46 people from 163 to 168cm
53 people from 168 to 173cm
45 people from 173 to 178cm
28 people from 179 to 184cm
21 people from 185 to 190cm
4 people from 190 to 195cm

Example1: Python program that plots histograms for weekly data(hist1.py)

```
import matplotlib.pyplot as plot

ages=[12,32,10,46,67,87,95,23,73,6,25,56,49,81,29]
age_group=[0,10,20,30,40,50,60,70,80,90,100]

plot.hist(ages,age_group)
#plot.hist(ages,age_group,rwidth=0.8)

plot.xticks(age_group)
plot.yticks(range(0,5,1))
plot.title('People Age distribution')
plot.xlabel('Age group of People who take the vaccine')
plot.ylabel("Fequency of People")
plot.show()
```

Output:

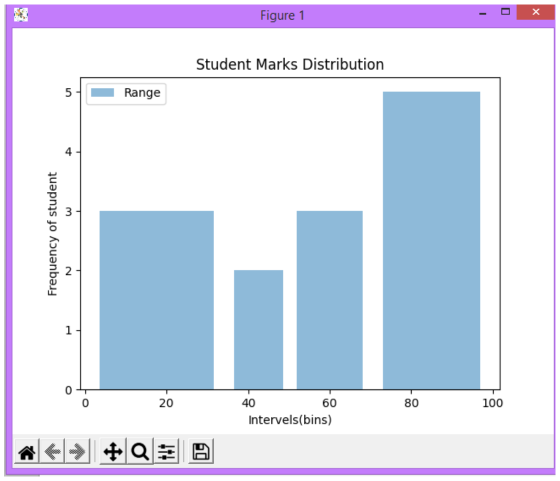


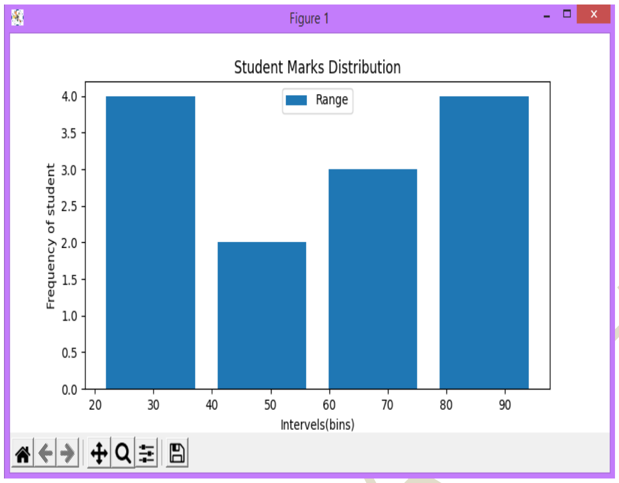


Example2: Drawing simple histogram using DataFrame(Taking user defined and default bin)

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

Marks=[40,60,55,88,96,93,20,35,70,60,89,20,33]
df=pd.DataFrame(Marks)
#print(df)
df.plot.hist(Marks,bins=[0,35,50,70,100],rwidth=0.8,alpha=0.5)
#df.plot.hist(Marks,bins=4,rwidth=0.8,figsize=(8,4)) #user defined intervals(bins)
plot.xlabel("Intervels(bins)")
plot.ylabel("Frequency of student")
plot.title("Student Marks Distribution")
plot.legend(['Range'])
plot.show()
```

OUTPUT:





5.5 Bar Chart : Concepts of Bar chart, bar(),set title, xlabel and ylabel

- A bar chart displays a set of categories in one axis and the percentage or frequencies of a variable for those categories in another axis.
- The height of the bar is either less or more depending upon the frequency value.
- In a Vertical Bar Chart, the X-axis will represent categories and Y-axis will represent frequencies. In a Horizontal Bar Chart, it is the inverse.
- In a Vertical Bar Chart, the bars grow downwards below the X-axis for negative values.
- In a Horizontal Bar Chart, the bars grow leftwards from the Y-axis for negative values.

Plotting Bar charts using pandas DataFrame:

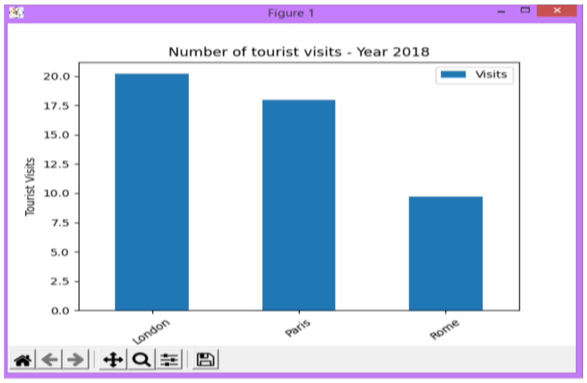
- While a bar chart can be drawn directly using matplotlib, it can be drawn for the DataFrame columns using the DataFrame class itself.
- The pandas DataFrame class in Python has a member plot. Using the plot instance various diagrams for visualization can be drawn including the Bar Chart.
- The bar() method draws a vertical bar chart and the barh() method draws a horizontal bar chart.
- The bar() and barh() of the plot member accepts X and Y parameters. By default, X takes the index of the DataFrame and all the numeric columns are drawn as Y.
- Any keyword argument supported by the method DatFrame.plot() can be passed to the bar() method in order to customize the bar chart. For example, the keyword argument title places a title on top of the bar chart.

Example: Bar Chart of a pandas DataFrame: one column as X-axis and another as Y-axis(bar1.py)

```
import pandas as pd
import matplotlib.pyplot as plot
# A python dictionary
data = {"City":["London", "Paris", "Rome"], "Visits":[20.20,17.95,9.7]}
# Dictionary loaded into a DataFrame
dataFrame = pd.DataFrame(data=data)
```

```
# Draw a vertical bar chart
dataFrame.plot.bar(x="City", y="Visits", rot=40, title="Number of tourist visits - Year 2018")
plot.xlabel('City')
plot.ylabel('Tourist Visits')
plot.show()
```

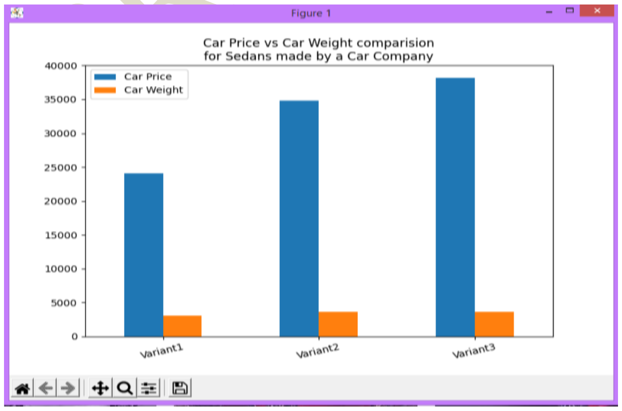
OUTPUT:



Compound Bar Chart or Nested Bar Chart:
The following Python code plots a compound bar chart combining two variables Car Price, Car Weight for the sedan variants produced by a car company.
Example Python program to plot a complex bar chart (bar2.py)

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

data = {"Car Price":[24050, 34850, 38150],"Car Weight":[3045, 3572, 3638]}

index= ["Variant1", "Variant2", "Variant3"]

# Dictionary loaded into a DataFrame
dataFrame = pd.DataFrame(data=data, index=index)

#Draw a vertical bar chart
dataFrame.plot.bar(rot=15, title='''Car Price vs Car Weight comparision
for Sedans made by a Car Company''')
plot.show()
```

OUTPUT:



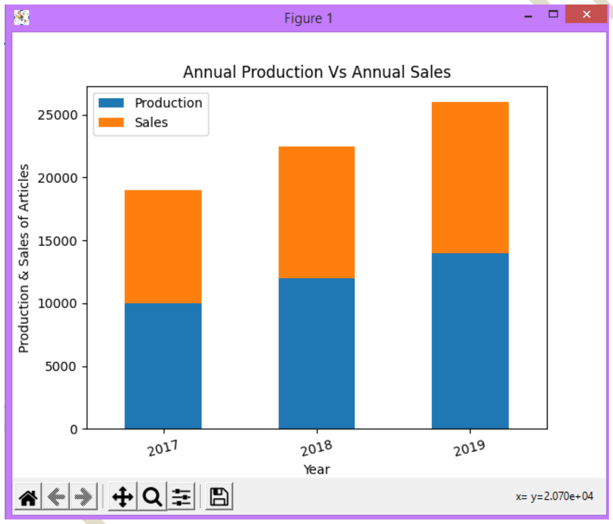
Stacked vertical bar chart:

A stacked bar chart illustrates how various parts contribute to a whole. The example Python code plots a pandas DataFrame as a stacked vertical bar chart. The Python code plots two variables - number of articles produced and number of articles sold for each year as stacked bars. The years are plotted as categories on which the plots are stacked.

Example: Python program to plot a stacked vertical bar chart (bar3.py)

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

data = {"Production":[10000, 12000, 14000],
      "Sales":[9000, 10500, 12000]}

index=["2017", "2018", "2019"]

# Dictionary loaded into a DataFrame
dataFrame = pd.DataFrame(data=data, index=index)
# Draw a vertical bar chart
dataFrame.plot.bar(stacked=True, rot=15,title="Annual Production Vs Annual Sales")

plot.xlabel('Year')

plot.ylabel(‘Production & Sales of Articles’)

plot.show()
```

OUTPUT:



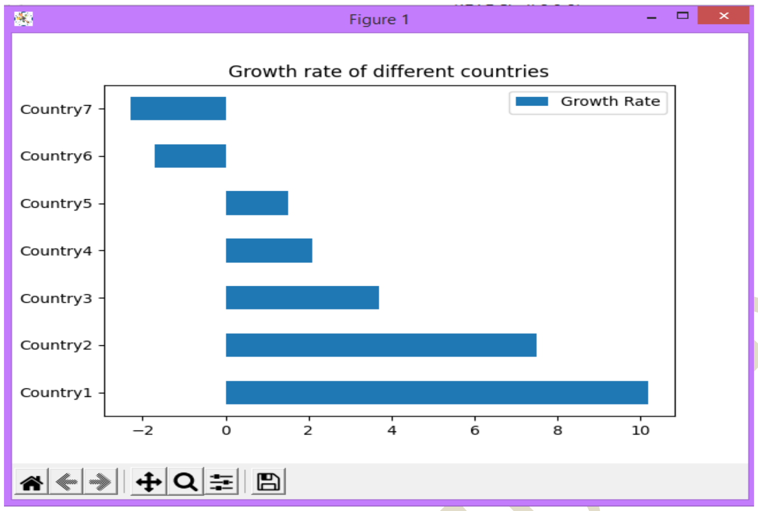
Horizontal Bar Chart:

In a horizontal bar chart categories are drawn in Y-axis and the frequencies of the variables for the categories are drawn in X-axis.
The Python example code plots the Growth Rate of different countries from a pandas DataFrame into a horizontal bar chart.

Example: python program to plot a horizontal bar chart(barh.py)

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

growthData = {"Countries": ["Country1", "Country2", "Country3", "Country4", "Country5",
"Country6", "Country7"], "Growth Rate":[10.2, 7.5, 3.7, 2.1, 1.5, -1.7, -2.3]}

dataFrame  = pd.DataFrame(data = growthData)

dataFrame.plot.barh(x='Countries', y='Growth Rate',title='Growth rate of different
countries')
plot.show()
```

OUTPUT:

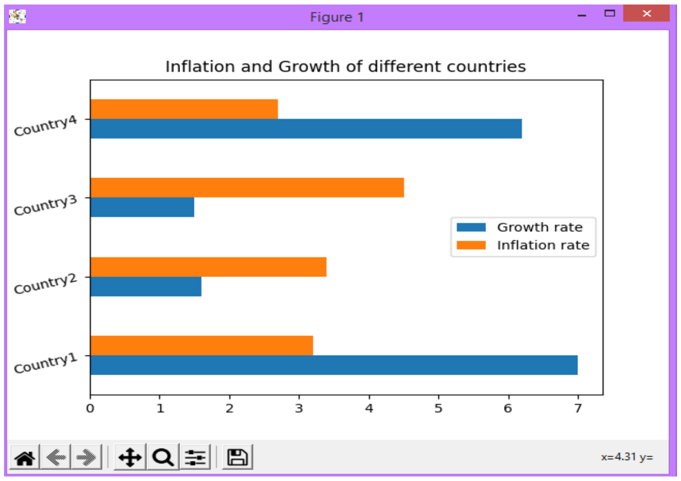


Compound horizontal bar chart:
A compound horizontal bar chart is drawn for more than one variable. For each variable a horizontal bar is drawn in the corresponding category. The example Python code plots Inflation and Growth for each year as a compound horizontal bar chart.

Example: Python program to plot a compound horizontal bar chart using pandas DataFrame

```
import pandas as pd
import matplotlib.pyplot as plot
inflationAndGrowth  = {"Growth rate": [7, 1.6, 1.5, 6.2],
                "Inflation rate":[3.2, 3.4, 4.5, 2.7]}
index=["Country1", "Country2", "Country3", "Country4"]

 #Python dictionary into a pandas DataFrame
dataFrame= pd.DataFrame(data = inflationAndGrowth)
dataFrame.index = index
dataFrame.plot.barh(rot=15, title="Inflation and Growth of different countries")
plot.show()
```

Output:



Example: Write a python program to implement different types of graphs.(allcharts.py)

```
import matplotlib.pyplot as plt
#bar graph
x = [10,20,30,40,50]
y = [10,50,60,20,30]
plt.bar(x,y)
plt.show()

'''
#histogram
z= [10, 5, 8, 4, 2]
plt.hist(z)
plt.show()
'''

'''
#line graph
a = [5, 2, 9, 4, 7]
b = [10, 5, 8, 4, 2]
plt.plot(a,b)
plt.show()
'''

'''
#scatter graph
x_values = [0,1,2,3,4,5]
squares = [0,1,4,9,16,25]
plt.scatter(x_values,squares, s=10, color = "blue")
plt.show()
'''
```