## **Data Visualisation using Python Matplotlib Pyplot**

**Data Visualisation using Python Matplotlib Pyplot**

**Data visualisation means graphical or pictorial representation of the data using graph, chart, etc. The purpose of plotting data is to visualise variation or show relationships between variables.**

**In this article, we will learn how to visualise data using Matplotlib library of Python by plotting charts such as line, bar with respect to the various types of data.**

## **Plotting using Matplotlib**

**Matplotlib library is used for creating static, animated, and interactive 2D- plots in Python. The command to install matplotlib is given below**

### **pip install matplotlib**

**For plotting using Matplotlib, we need to import its Pyplot module using the following command:**

### **import matplotlib.pyplot as plt**

**NOTE: plt is an alias or an alternative name for matplotlib.pyplot**

## **Different components of plot**

1. **Chart Title**
2. **Legend**
3. **X-Axis label**
4. **Y-Axis label**
5. **X-ticks**
6. **Y-ticks**

## **plot( ) function of pyplot module**

**The plot() function of the pyplot module is used to create a chart. It is always expected that the data presented through charts easily understood. Hence, while presenting data we should always give a chart title, label the axis of the chart and provide legend in case we have more than one plotted data.**

## **show( ) function of pyplot module**

**The show() function is used to display the figure created using the plot() function.**

**Let we discuss a program to demonstrate the use of plot( ) and show( ) function.**

**#Example 1: program to show number of students vs marks obtained (Line Plot)**

**import matplotlib.pyplot as plt**

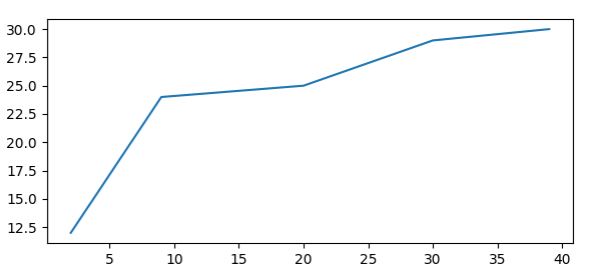
**nos = [2, 9, 20, 25, 30, 39]**

**marks = [12, 24, 25, 27,29, 30]**

**plt.plot(nos, marks)**

**plt.yticks(nos)**

**plt.show()**

**Data Visualisation line plot**

**plot() is provided with two parameters(nos, marks), which indicates values for x-axis and y-axis, respectively. The x and y ticks are displayed accordingly. As shown in Figure 4.2, the plot() function by default plots a line chart. We can click on the save button on the output window and save the plot as an image. A figure can also be saved by using savefig() function.**

**for example: plt.savefig(‘marks.png’).**

**#Example 2: program to show number of students vs marks obtained (Bar Plot)**

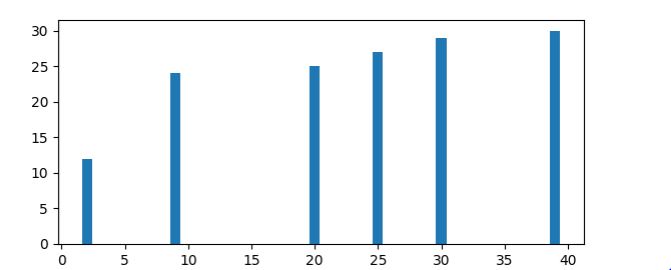
**import matplotlib.pyplot as plt**

**nos = [2, 9, 20, 25, 30, 39]**

**marks = [12, 24, 25, 27,29, 30]**

**plt.bar(nos, marks)**

**plt.show()**

**Data Visualisation – Bar Plot**

**#Example 3: program to show number of students vs marks obtained (Horizontal Bar Plot)**

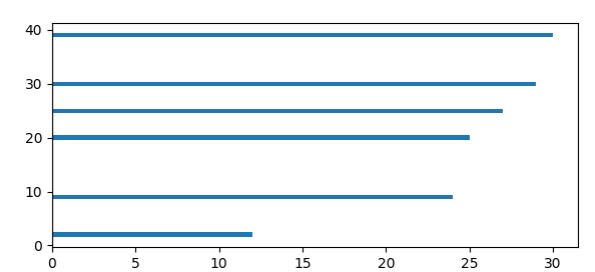
**import matplotlib.pyplot as plt**

**nos = [2, 9, 20, 25, 30, 39]**

**marks = [12, 24, 25, 27,29, 30]**

**plt.barh(nos, marks)**

**plt.show()**

**** Data Visualisation – Barh Plot

## 

## **Customisation of Plots :**

**Pyplot library gives us numerous functions, which can be used to customise charts such as adding titles or  
legends. Some of the options are listed below**

| **Options** | **Explanation** |
| --- | --- |
| **grid** | **shows the grid lines on plot.** |
| **legend** | **Place a legend on the axes.** |
| **savefig** | **used to save the figure(plot)** |
| **show** | **Display the figure** |
| **title** | **Display the title for the plot** |
| **xlabel** | **Set the label for the X-axis.** |
| **ylabel** | **Set the label for the y-axis.** |
| **xticks** | **Get or set the current tick locations and labels of the X-axis.** |
| **yticks** | **Get or set the current tick locations and labels of the Y-axis.** |

**#Example 4: Plotting a line chart of “Month Name” versus “Monthly Saving” given below and adding label on X and Y axis, and adding a title and grids to the chart.**

**month =[“Jan”, “Feb”, “Mar”, “Apr”, “May”, “Jun”]**

**msaving=[1000, 500, 700, 550, 600, 800 ]**

**import matplotlib.pyplot as plt**

**month =["Jan", "Feb", "Mar", "Apr", "May", "Jun"]**

**msaving=[1000, 500, 700, 550, 600, 800 ]**

**plt.plot(month, msaving)**

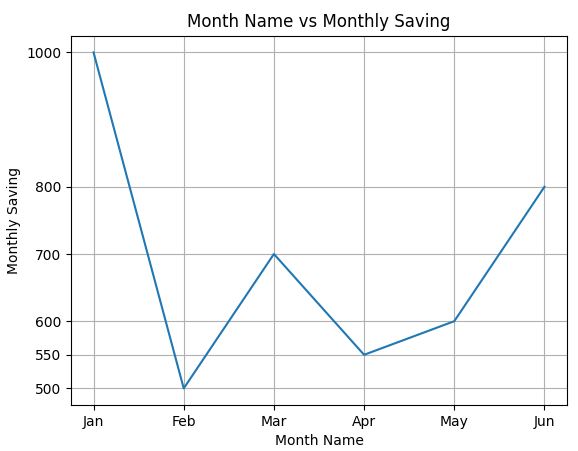
**plt.xlabel("Month Name") #add the Label on x-axis**

**plt.ylabel("Monthly Saving") #add the Label on y-axis**

**plt.title("Month Name vs Monthly Saving") #add the title to the chart**

**plt.grid(True) #add gridlines to the background**

**plt.show()**

**Data Visualisation – Month Name Vs Monthly Saving**

## **Other Attributes for Customisation of Plots :**

**1. Marker : A marker is any symbol that represents a data value in a line chart. We can specify each point in the line chart through a marker.**

**2. Colour : We can format the plot further by changing the colour of the plotted data. We can either use character codes or the color names as values to the parameter color in the plot().** Following table shows aome colour characters.

| **Character** | **Colour** |
| --- | --- |
| **‘b’** | **Blue** |
| **‘g’** | **Green** |
| **‘r’** | **Red** |
| **‘c’** | **Cyan** |
| **‘m’** | **Magenta** |
| **‘y’** | **Yellow** |
| **‘k’** | **Black** |
| **‘w’** | **White** |

**3. Linewidth and Line Style : The linewidth and linestyle property can be used to change the width and the style of the line chart. Linewidth is specified in pixels. The default line width is 1 pixel. We can also set the line style of a line chart using the linestyle parameter. It can take a value such as “solid”, “dotted”, “dashed” or “dashdot”.**

**#Example 5:** **Plotting a line chart of “Month Name” versus “Monthly Saving” given below**

**month =[“Jan”, “Feb”, “Mar”, “Apr”, “May”, “Jun”]**

**msaving=[1000, 500, 700, 550, 600, 800 ]**

**Let us plot a line chart where:**

**i. x axis will represent Month  
ii. y axis will represent Monthly Saving  
iii. x axis label should be “Month Name”  
iv. y axis label should be “Monthly Saving”  
v. colour of the line should be red  
vi. use \* as marker  
vii. Marker size as10  
viii. The title of the chart should be “Month Names vs Monthly Saving”.  
ix. Line style should be dashed  
x. Linewidth should be 2.**

**Ans.**

**import matplotlib.pyplot as plt**

**month =["Jan", "Feb", "Mar", "Apr", "May", "Jun"]**

**msaving=[1000, 500, 700, 550, 600, 800 ]**

**plt.plot(month, msaving,marker="\*", markersize=10, color="red", linewidth=2, linestyle='dashdot')**

**plt.xlabel("Month Name") #add the Label on x-axis**

**plt.ylabel("Monthly Saving") #add the Label on y-axis**

**plt.title("Month Name vs Monthly Saving") #add the title to the chart**

**plt.grid(True) #add gridlines to the background**

**plt.show()**

## **OR**

**we can also create the DataFrame using 2 lists, and in the plot function we can passed the Month Name and Monthly Saving of the DataFrame**

**Ans.**

**import matplotlib.pyplot as plt**

**import pandas as pd**

**month =["Jan", "Feb", "Mar", "Apr", "May", "Jun"]**

**msaving=[1000, 500, 700, 550, 600, 800 ]**

**df=pd.DataFrame({"Month" : month, "MonthlySaving" : msaving})**

**plt.plot(df.Month, df.MonthlySaving, marker="\*", markersize=10, color="red", linewidth=2, linestyle='dashdot')**

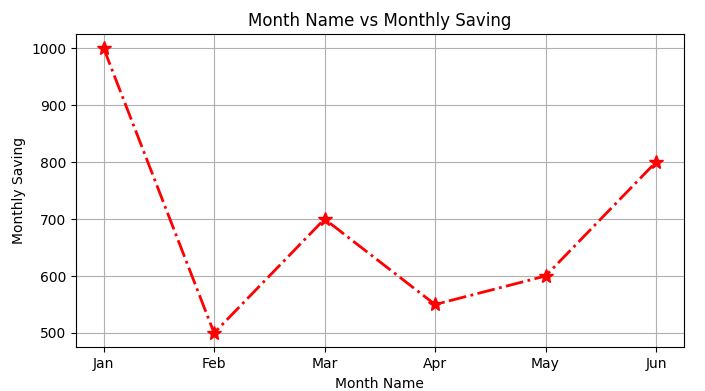
**plt.xlabel("Month Name") #add the Label on x-axis**

**plt.ylabel("Monthly Saving") #add the Label on y-axis**

**plt.title("Month Name vs Monthly Saving") #add the title to the chart**

**plt.grid(True) #add gridlines to the background**

**plt.show()**

**Data Visualisation – Customisation of Line Plot**

## **The Pandas Plot Function :**

**In above Programs, we learnt that the plot() function of the pyplot module of matplotlib can be used to plot a chart. However, starting from version 0.17.0, Pandas objects Series and DataFrame come equipped with their own .plot() methods. Thus, if we have a Series or DataFrame type object (let’s say ‘s’ or ‘df’) we can call the plot method by writing:**

**s.plot() or df.plot()**

**The plot() method of Pandas accepts an arguments “kind” that can be used to plot a variety of graphs. The general syntax is:**

**plt.plot(kind)**

| **kind =** | **Plot type** |
| --- | --- |
| **line** | **Line plot (default)** |
| **bar** | **Vertical bar plot** |
| **barh** | **Horizontal bar plot** |
| **hist** | **Histogram** |
| **pie** | **pie plot** |

## **Plotting a Line chart :**.

**A line plot is a graph that shows the frequency of data along a number line. It is used to show continuous  
dataset. A line plot is used to visualise growth or decline in data over a time interval.**

**#Example 6:** **The file “monthsales.csv” have stored the sales (in Rs) made in first six months for four years.**

| **2018** | **2019** | **2020** | **2021** |
| --- | --- | --- | --- |
| **2500** | **2600** | **2450** | **3250** |
| **2200** | **1800** | **2000** | **3200** |
| **3000** | **2500** | **3000** | **3800** |
| **2800** | **3000** | **3100** | **2800** |
| **2750** | **3200** | **1950** | **2900** |
| **3500** | **2000** | **3550** | **2300** |

**Year-wise sales of first six months**

**Draw the line chart for the data given above with following details.**

1. **Chart title as “Year-Wise Sales”**
2. **X-axis label as “Months”**
3. **Y-axis label as “Sales”**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# reads "monthsales.csv" to df by giving path to the file**

**df=pd.read\_csv("monthsales.csv")**

**#create a line plot of different color for each week**

**df.plot(kind='line', color=['red', 'blue', 'brown', 'Yellow'])**

**# Set title to "Year-Wise Sales"**

**plt.title('Year-Wise Sales')**

**# Label x axis as "Months"**

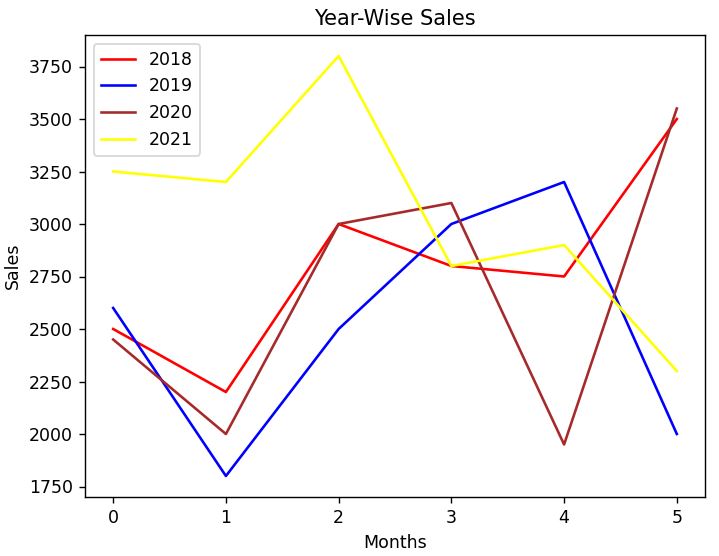
**plt.xlabel('Months')**

**# Label y axis as "Sales"**

**plt.ylabel('Sales')**

**#Display the figure**

**plt.show()**

**Data Visualisation: Line chart from csv**

## **Customising a Line chart :**.

**We can substitute the ticks at x axis with a list of values of our choice by using plt.xticks(ticks,label) where  
ticks is a list of locations(locs) on x axis at which ticks should be placed, label is a list of items to place at the  
given ticks.**

**#Example 7: Assuming the same CSV file, i.e., monthsales.csv, plot the line chart with following customisations. Chart should have Month name on X-axis instead of numbers.  
Maker =”\*”  
Marker size=10  
linestyle=”–“  
Linewidth =3**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# reads "monthsales.csv" to df by giving path to the file**

**df=pd.read\_csv("monthsales.csv")**

**df["Months"]=["Jan","Feb","Mar","Apr","May","June"]**

**#create a line plot of different color for each week**

**df.plot(kind='line', color=['red', 'blue', 'brown', 'Yellow'],marker="\*", markersize=10, linewidth=3, linestyle="--")**

**# Set title to "Year-Wise Sales"**

**plt.title('Year-Wise Sales')**

**# Label x axis as "Months"**

**plt.xlabel('Months')**

**# Label y axis as "Sales"**

**plt.ylabel('Sales')**

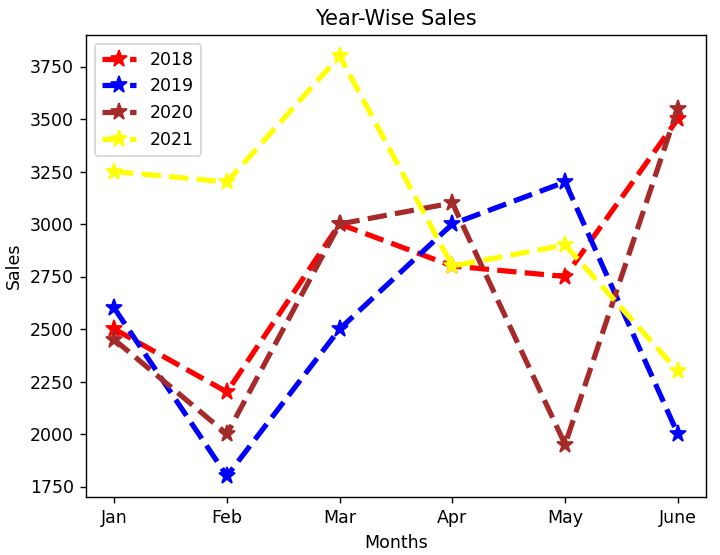
**ticks = df.index.tolist()**

**#displays corresponding Month name on x axis**

**plt.xticks(ticks,df.Months)**

**#Display the figure**

**plt.show()**

**Data Visualisation: Line chart from csv with Month Name**

## **Plotting a bar chart :**.

**In above example lines are unable to show comparison between the years for which the sales data is plotted. In order to show comparisons, we prefer Bar charts. Unlike line plots, bar charts can plot strings on the x axis. To plot a bar chart, we will specify kind=’bar’.**

**#Example 8: Let us take the same data as shown in Example 6 in file “monthsales.csv”**

| **Month** | **2018** | **2019** | **2020** | **2021** |
| --- | --- | --- | --- | --- |
| **Jan** | **2500** | **2600** | **2450** | **2900** |
| **Feb** | **2200** | **1800** | **2000** | **3200** |
| **Mar** | **3000** | **2500** | **3000** | **3800** |
| **Apr** | **2800** | **3000** | **3100** | **2800** |
| **May** | **2750** | **3200** | **1950** | **2900** |
| **June** | **3500** | **2000** | **3550** | **2300** |

**Year-wise sales of first six months**

**To plot the bar chart of above data, use the same code as given in Example 6, just make a small change use**

**kind = ‘bar’ instead of kind = ‘line’**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# reads "monthsales.csv" to df by giving path to the file**

**df=pd.read\_csv("monthsales.csv")**

**#create a line plot of different color for each week**

**df.plot(kind='bar', color=['red', 'blue', 'brown', 'Yellow'])**

**# Set title to "Year-Wise Sales"**

**plt.title('Year-Wise Sales')**

**# Label x axis as "Months"**

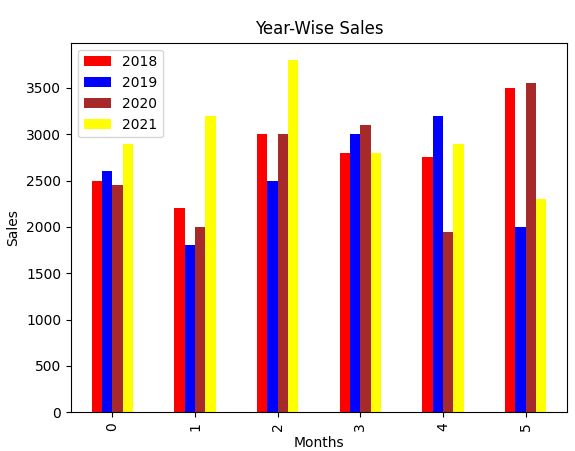
**plt.xlabel('Months')**

**# Label y axis as "Sales"**

**plt.ylabel('Sales')**

**#Display the figure**

**plt.show()**

**Data Visualisation: Bar chart year wise sales**

**#Example 9: Let us add a new column “Month” in the file “monthsales.csv” as shown below**

| **Month** | **2018** | **2019** | **2020** | **2021** |
| --- | --- | --- | --- | --- |
| **Jan** | **2500** | **2600** | **2450** | **2900** |
| **Feb** | **2200** | **1800** | **2000** | **3200** |
| **Mar** | **3000** | **2500** | **3000** | **3800** |
| **Apr** | **2800** | **3000** | **3100** | **2800** |
| **May** | **2750** | **3200** | **1950** | **2900** |
| **June** | **3500** | **2000** | **3550** | **2300** |

**Year-wise sales of first six months**

**To plot the bar chart of above data and to show the month name in X-axis just add the following attribute in plot()**

**x = Month #Month is the column name**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# reads "monthsales.csv" to df by giving path to the file**

**df=pd.read\_csv("monthsales.csv")**

**#create a line plot of different color for each month**

**df.plot(kind='bar', x='Month', color=['red', 'blue', 'brown', 'Yellow'])**

**# Set title to "Year-Wise Sales"**

**plt.title('Year-Wise Sales')**

**# Label x axis as "Months"**

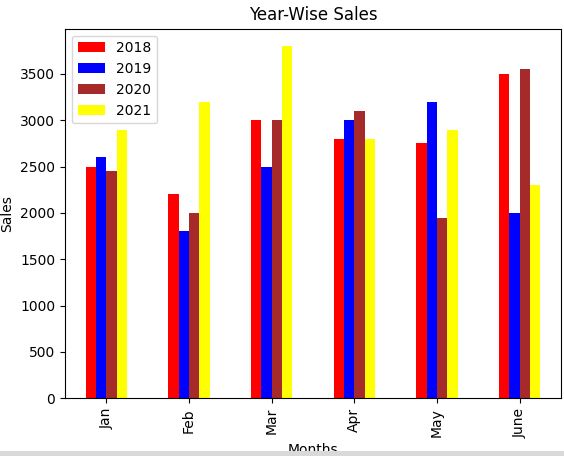
**plt.xlabel('Months')**

**# Label y axis as "Sales"**

**plt.ylabel('Sales')**

**#Display the figure**

**plt.show()**

**Data Visualisation: Bar chart year wise with month name**

## **Customising a Bar chart :**.

**We can also customise the bar chart by adding certain parameters to the plot function. We can control the  
edgecolor of the bar, linestyle and linewidth. We can also control the color of the lines.**

**#Example 10: Write a Program to display Bar plot for the “monthsales.csv” file with column ‘Month’ on x axis, and having the following customisation:**

**● Edgecolor to green  
● Linewidth as 2  
● Line style as “–“**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# reads "monthsales.csv" to df by giving path to the file**

**df=pd.read\_csv("monthsales.csv")**

**#create a line plot of different color for each week**

**df.plot(kind = 'bar', x ='Month', color = ['red', 'blue', 'brown', 'Yellow'], edgecolor = 'Green', linewidth=2, linestyle ='--')**

**# Set title to "Year-Wise Sales"**

**plt.title('Year-Wise Sales')**

**# Label x axis as "Months"**

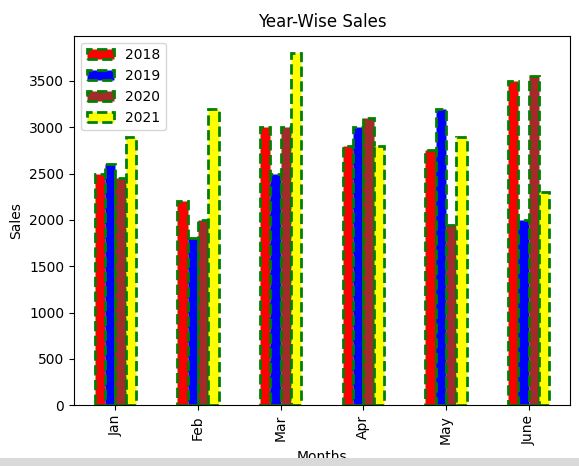
**plt.xlabel('Months')**

**# Label y axis as "Sales"**

**plt.ylabel('Sales')**

**#Display the figure**

**plt.show()**

**Data Visualisation: Bar chart with Customisation**

## **Customising a Bar chart :**.

**Histograms are column-charts, where each column represents a range of values, and the height of a column  
corresponds to how many values are in that range.**

**The df.plot(kind=’hist’) function automatically selects the size of the bins based on the spread of values in  
the data.**

**#Example 11:** **Plot a histogram to show the bin value calculated by plot( ) function.**

**import pandas as pd**

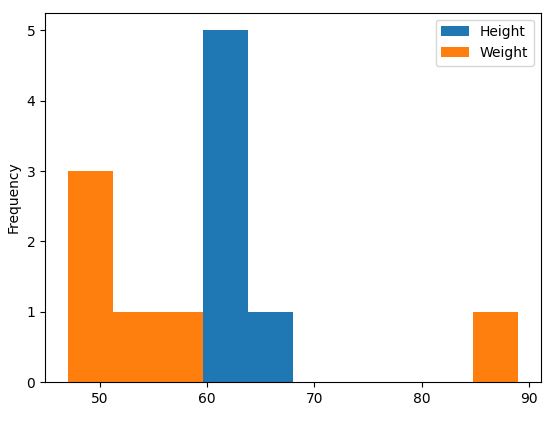
**import matplotlib.pyplot as plt**

**data = {'Height' : [60, 61, 63, 65, 61, 60], 'Weight' : [47, 89, 52, 58, 50, 47]}**

**df=pd.DataFrame(data)**

**df.plot(kind='hist')**

**plt.show()**

**Data Visualisation: Histogram**

**It is also possible to set value for the bins parameter, for example,**

**df.plot(kind=’hist’,bins=20)  
df.plot(kind=’hist’,bins=[18,19,20,21,22])  
df.plot(kind=’hist’,bins=range(18,25))**

**#Example 12:** **Plot a histogram with user specified bin values.**

**[45, 50, 55, 60, 65, 70, 75, 80, 85]**

**import pandas as pd**

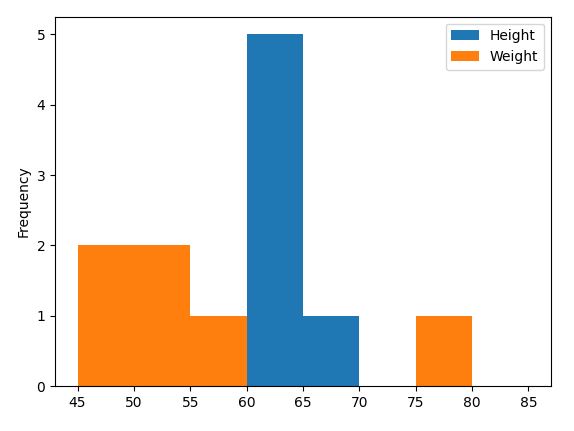
**import matplotlib.pyplot as plt**

**data = {'Height' : [60, 61, 63, 65, 61, 60], 'Weight' : [47, 89, 52, 58, 50, 47]}**

**df=pd.DataFrame(data)**

**df.plot(kind='hist', bins=[45, 50, 55, 60, 65, 70, 75, 80, 85])**

**plt.show()**

**Data Visualisation: Histogram with bin value**

## **Customising a Histogram :**.

**Let we create the same histogram as created above with the following customisation**

1. **Edgecolor: It is the border of each hist and let make it green**
2. **line style: Let make line style to ‘:’**
3. **line width: Let make it to 3**

**Let us try another property called fill, which takes boolean values. The default True means each hist will be filled with color and False means each hist will be empty. Another property called hatch can be used to fill to each hist with pattern ( ‘-‘, ‘+’, ‘x’, ‘\’, ‘\*’, ‘o’, ‘O’, ‘.’).**

**#Example 13:**

**import pandas as pd**

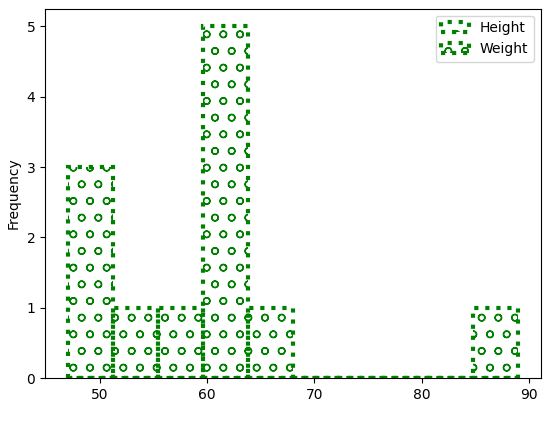
**import matplotlib.pyplot as plt**

**data = {'Height' : [60, 61, 63, 65, 61, 60], 'Weight' : [47, 89, 52, 58, 50, 47]}**

**df=pd.DataFrame(data)**

**df.plot(kind='hist', edgecolor = 'Green', linewidth=3, linestyle=':', fill=False, hatch='o')**

**plt.show()**

**Data Visualisation: Customised Histogram**

## **Reading specific column from CSV file:**.

**Let us consider the ‘monthsales.csv’ file that we created above**

| **Month** | **2018** | **2019** | **2020** | **2021** |
| --- | --- | --- | --- | --- |
| **Jan** | **2500** | **2600** | **2450** | **2900** |
| **Feb** | **2200** | **1800** | **2000** | **3200** |
| **Mar** | **3000** | **2500** | **3000** | **3800** |
| **Apr** | **2800** | **3000** | **3100** | **2800** |
| **May** | **2750** | **3200** | **1950** | **2900** |
| **June** | **3500** | **2000** | **3550** | **2300** |

**Year-wise sales of first six months**

**#Example 14: Now we are going to read specific columns (2018 and 2021) from “monthsales.csv” and plot the line chart**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# reads "monthsales.csv" to df by giving path to the file**

**df=pd.read\_csv("monthsales.csv", usecols=['2018', '2021'])**

**#create a line plot of different color for each week**

**df.plot(kind='line', color=['red', 'blue'])**

**# Set title to "Year-Wise Sales"**

**plt.title('Year-Wise Sales')**

**# Label x axis as "Months"**

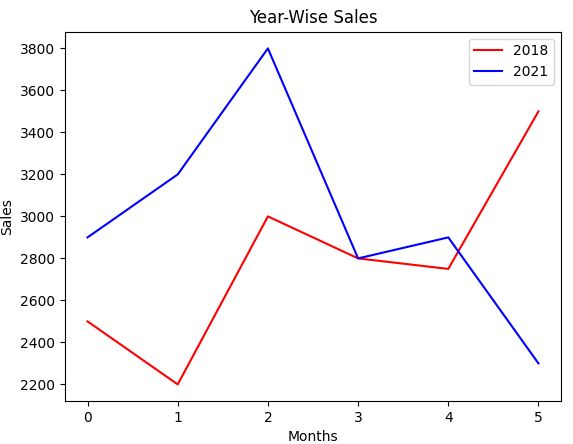
**plt.xlabel('Months')**

**# Label y axis as "Sales"**

**plt.ylabel('Sales')**

**#Display the figure**

**plt.show()**

**Data Visualisation: Specific column from CSV**

**#Example 15:** **If you want to show the names of the month along X-axis then make few changes in the above code. Changes are highlighted in the code given below.**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**df=pd.read\_csv("monthsales.csv", usecols=['Month', '2018', '2021'])**

**df.plot(kind='line', x='Month', color=['red', 'blue'])**

**plt.title('Year-Wise Sales')**

**plt.xlabel('Months')**

**plt.ylabel('Sales')**

**plt.show()**

**OR**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**df=pd.read\_csv("monthsales.csv", usecols=['Month', '2018', '2021'])**

**df.plot(kind='line', color=['red', 'blue'])**

**plt.title('Year-Wise Sales')**

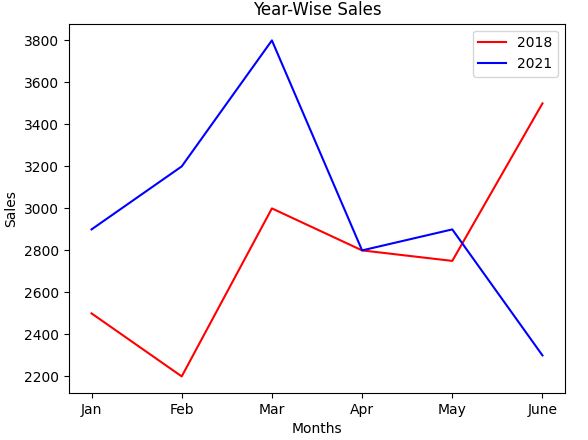
**plt.xlabel('Months')**

**plt.ylabel('Sales')**

**ticks = df.index.tolist()**

**plt.xticks(ticks,df.Month)**

**plt.show()**

**Data Visualisation: Specific column from CSV with Month name**