Data Science using Python

Dr. Kamini Solanki, Assistant Professor
Parul Institute of Computer Application - BCA

Parul® University

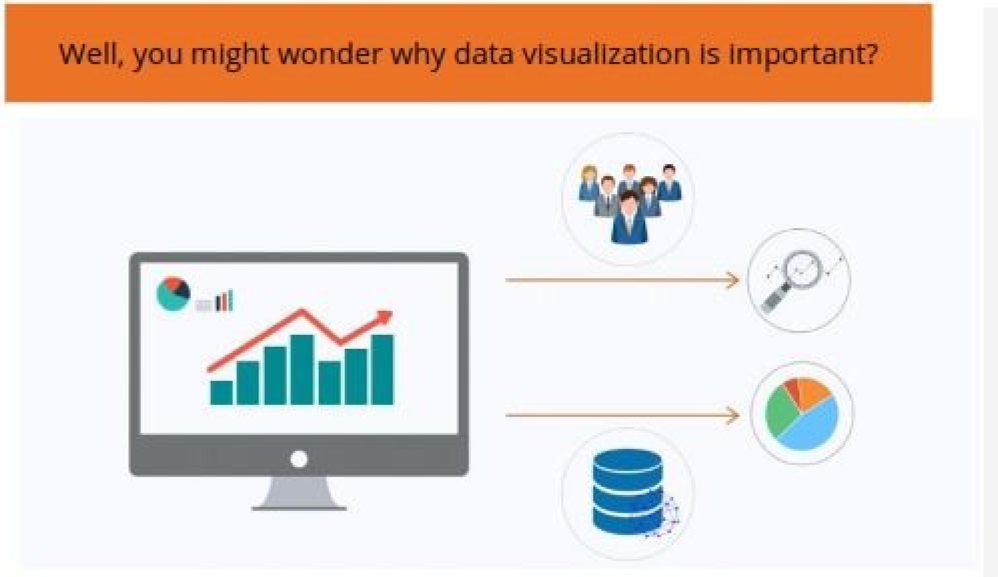
DIGITAL LEARNING CONTENT

CHAPTER - 8

Data Visualization in Python using Matplotlib

Data Visualization

- Data visualization is a technique to present the data in a pictorial or graphical format.



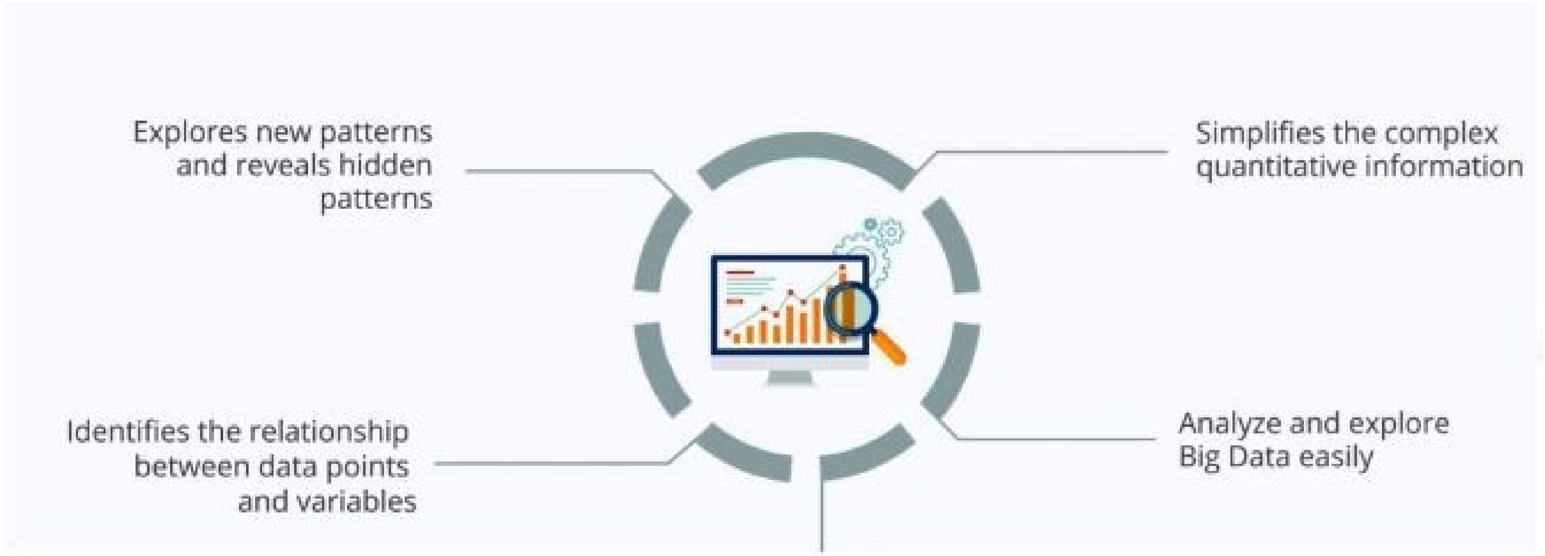
Data Visualization

- You are a Sales Manager in a leading global organization. The organization plans to study the sales details of each product across all regions and countries. This is to identify the product which has the highest sales in a particular region and up the production. This research will enable the organization to increase the manufacturing of that product in that particular region.

● Product A
● Product B
● Product C
● Product D

Data Visualization

- The main benefits of data visualization are:


Explores new patterns and reveals hidden patterns
Simplifies the complex quantitative information
Identifies the relationship between data points and variables
Analyze and explore Big Data easily

Considerations of Data Visualization

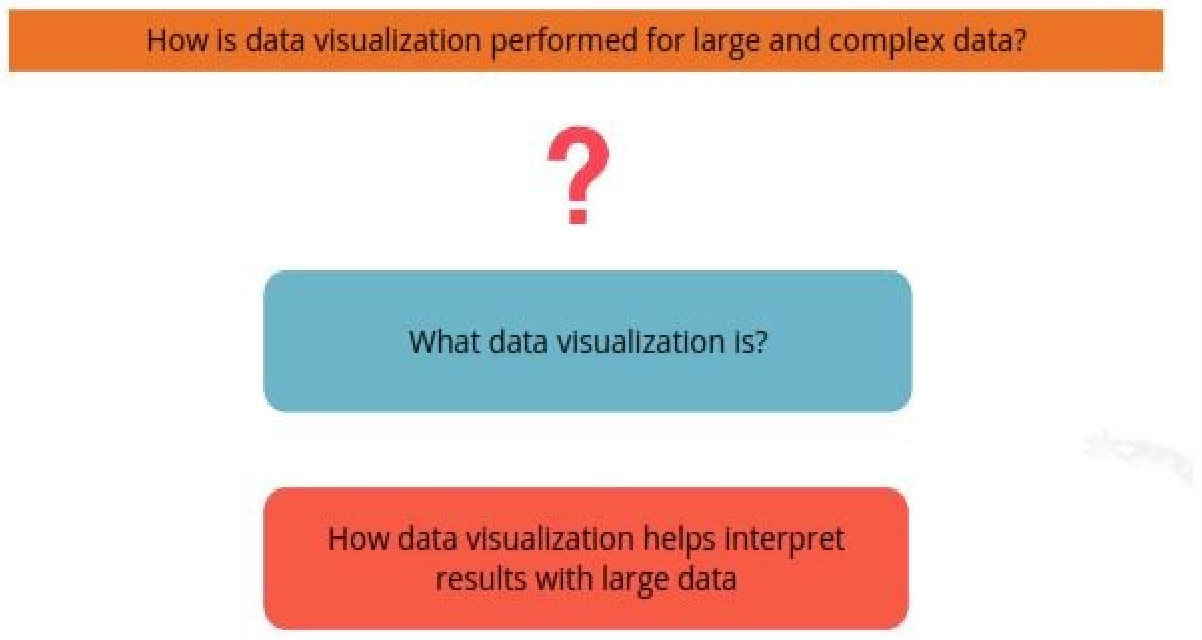
- Three major considerations for data visualization:

- Clarity :- Ensure the dataset is complete and relevant. This enables the Data Scientist to use the new patterns obtained from the data in the relevant places.
- Accuracy :- Ensure you use appropriate graphical representation to convey the intended message.
- Efficiency :- Use efficient visualization techniques that highlight all the data points.

Factors of Data Visualization

- There are some basic factors that one needs to be aware of before visualizing the data:
- The visual effect includes the usage of appropriate shapes, colors, and sizes to represent the analyzed data.
- The coordinate system helps organize the data points within the provided coordinates.
- The data types and scale choose the type of data, for example, numeric or categorical.
- The informative interpretation helps create visuals in an effective and easily interpretable manner using labels, title, legends, and pointers.

Data Visualization Tool: Python



How is data visualization performed for large and complex data?

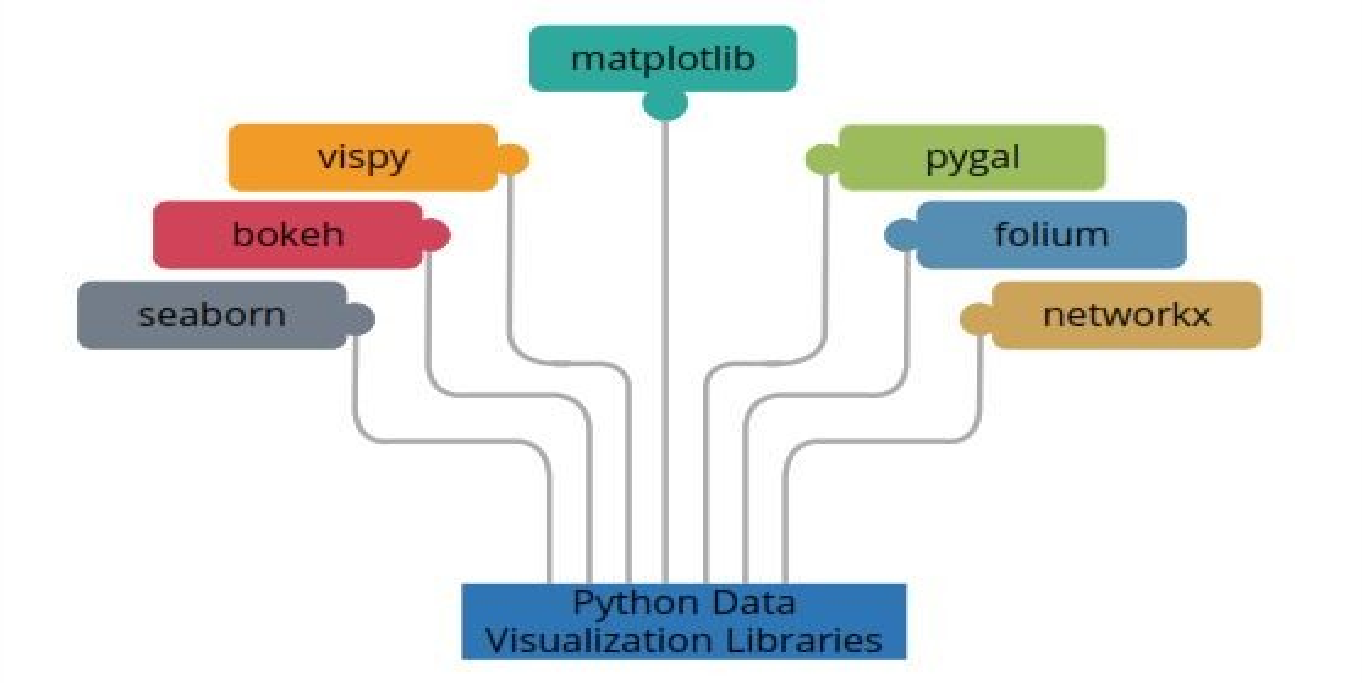
?

What data visualization is?

How data visualization helps interpret results with large data

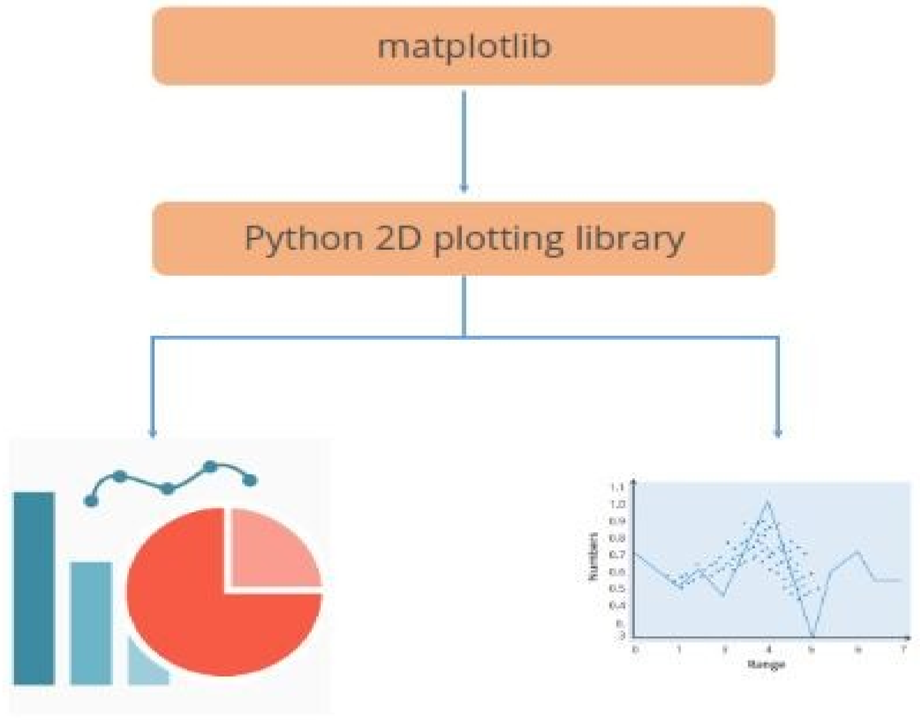
Python Libraries

- Many new Python data visualization libraries are introduced recently, such as:



Python Libraries: matplotlib

- Using Python's matplotlib, the data visualization of large and complex data becomes easy.

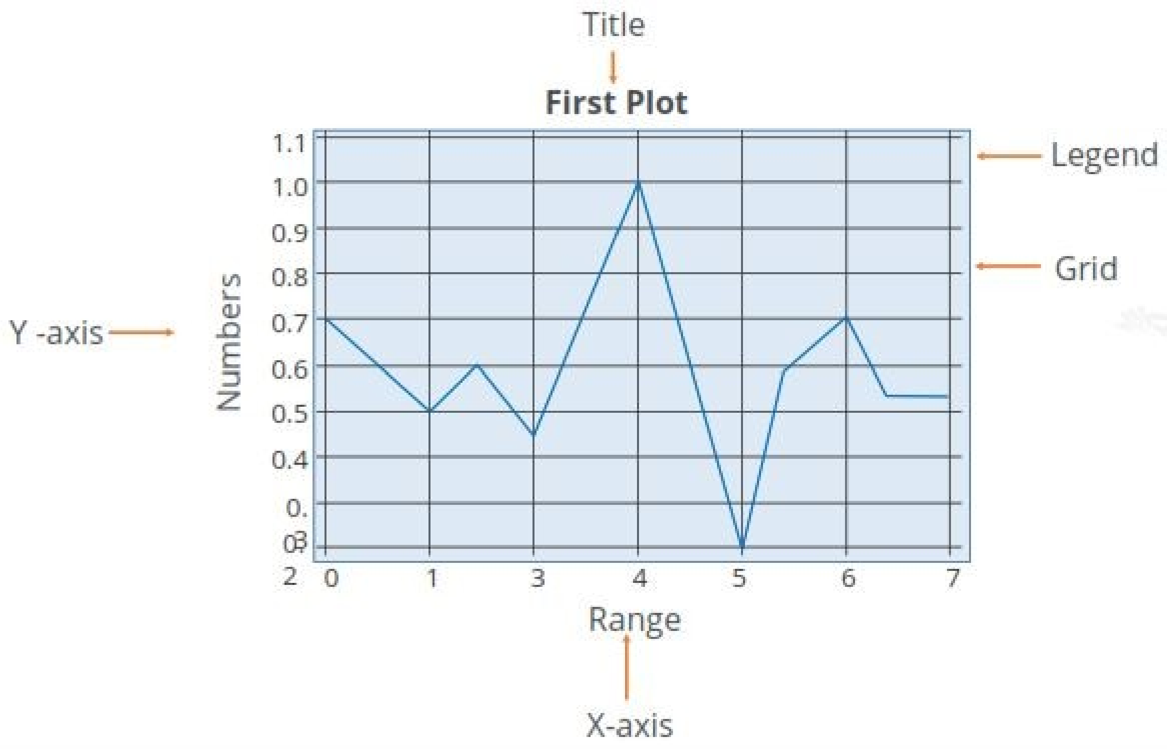


Python Libraries: matplotlib

- There are several advantages of using matplotlib to visualize data. They are as follows:
- Is a multi-platform data visualization tool; therefore, it is fast and efficient.
- Can work well with many operating systems and graphics back ends
- Has high-quality graphics and plots to print and view a range of graphs
- With Jupyter notebook integration, the developers are free to spend their time implementing features
- Has large community support and cross platform support as it is an open source tool
- Has full control over graphs or plot styles

The Plot

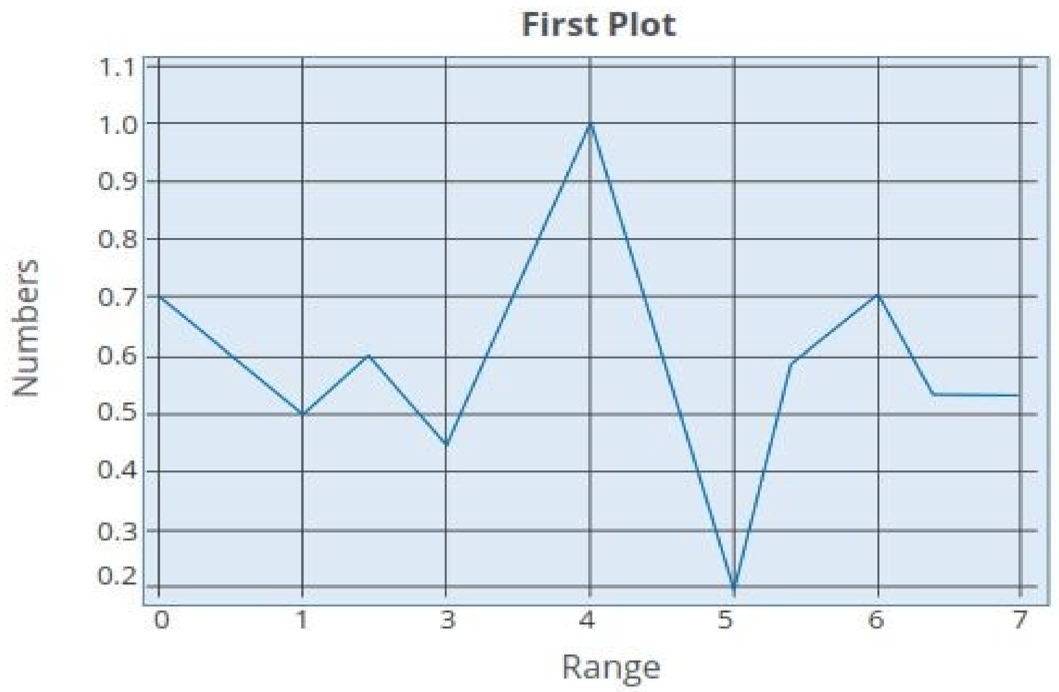
- A plot is a graphical representation of data, which shows the relationship between two variables or the distribution of data.



Parul® University

DIGITAL LEARNING CONTENT

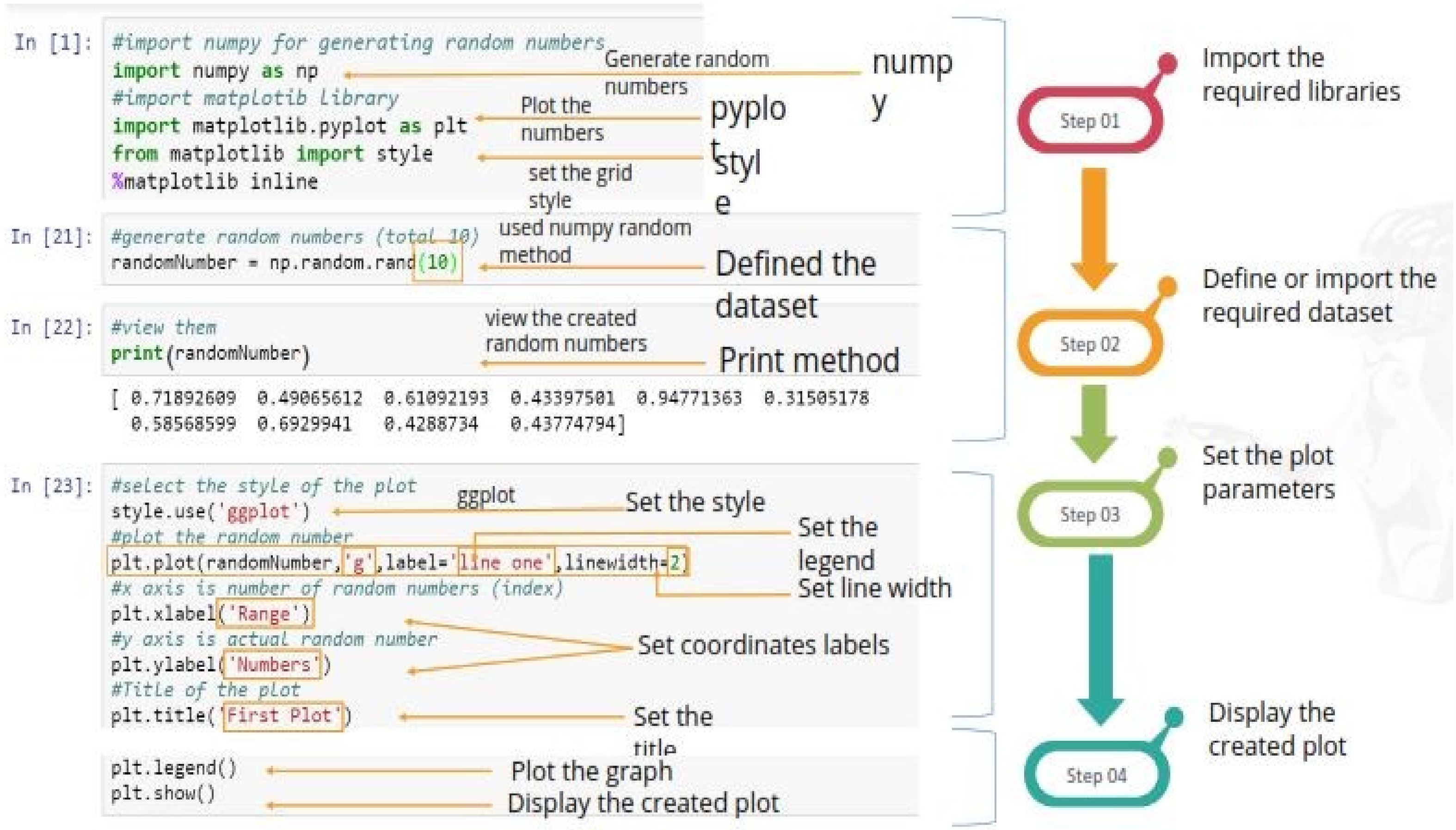
Steps to Create a Plot : Example



Parul® University

DIGITAL LEARNING CONTENT

Steps to Create a Plot : Example



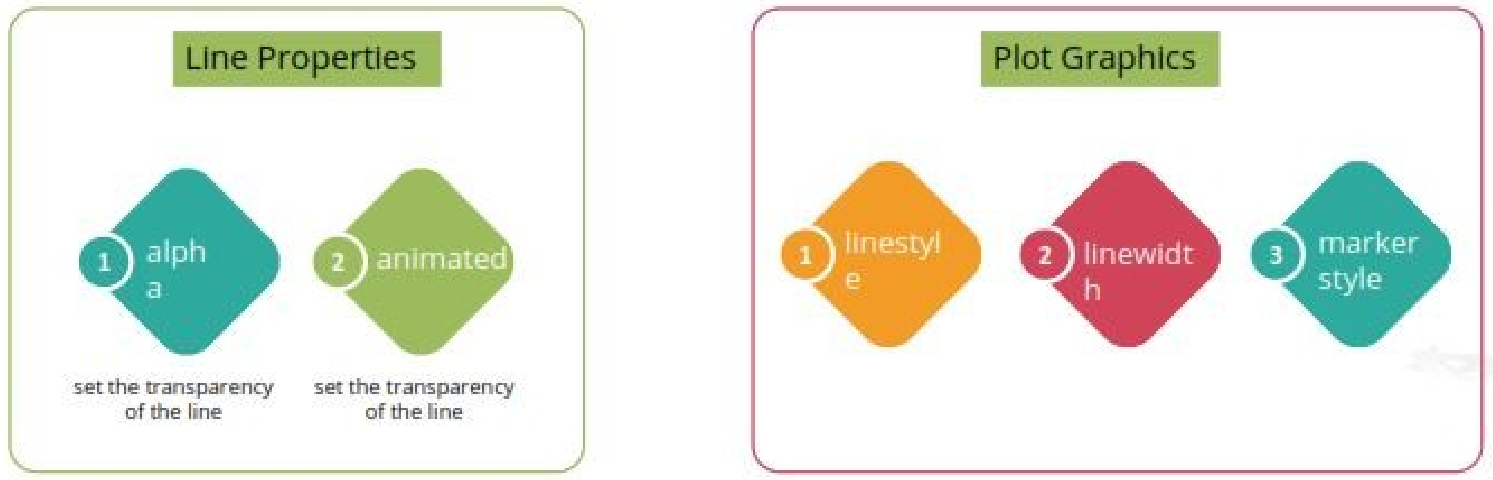
```
In [1]:  #import numpy for generating random numbers
         import numpy as np
         #import matplotlib library
         import matplotlib.pyplot as plt
         from matplotlib import style
         %matplotlib inline

In [21]: #generate random numbers (total 10)
         randomNumber = np.random.rand(10)

In [22]: #view them
         print(randomNumber)

         [ 0.71892609  0.49065612  0.61092193  0.43397501  0.94771363  0.31505178
           0.58568599  0.6929941   0.4288734   0.43774794]

In [23]: #select the style of the plot
         style.use('ggplot')
         #plot the random number
         plt.plot(randomNumber,'g',label='line one',linewidth=2)
         #x axis is number of random numbers (index)
         plt.xlabel('Range')
         #y axis is actual random number
         plt.ylabel('Numbers')
         #Title of the plot
         plt.title('First Plot')

         plt.legend()
         plt.show()
```

Parul® University

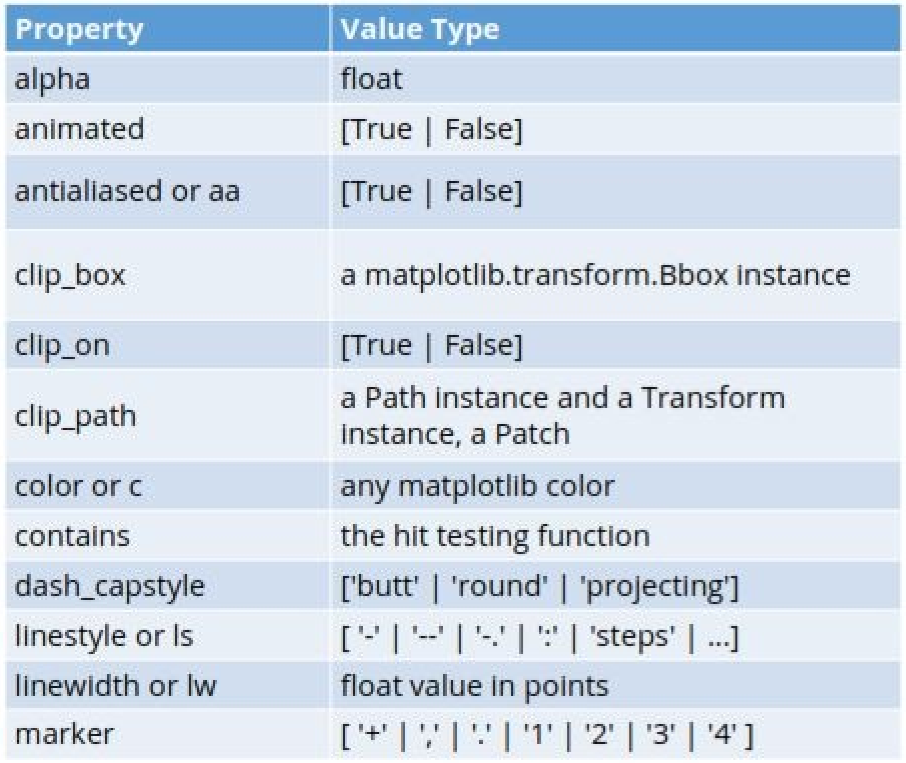
DIGITAL LEARNING CONTENT

Line Properties

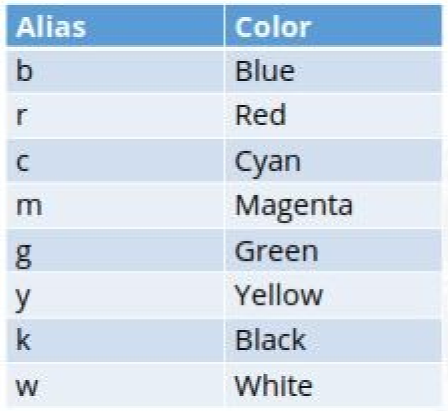
- matplotlib also offers various line colors.



Line Properties



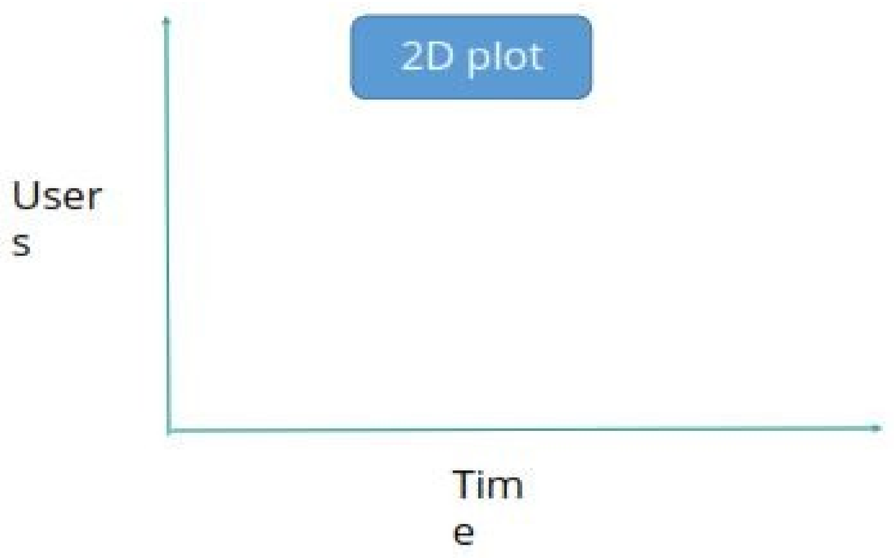
Property	Value Type
alpha	float
animated	[True | False]
antialiased or aa	[True | False]
clip_box	a matplotlib.transform.Bbox instance
clip_on	[True | False]
clip_path	a Path instance and a Transform instance, a Patch
color or c	any matplotlib color
contains	the hit testing function
dash_capstyle	['butt' | 'round' | 'projecting']
linestyle or ls	['-' | '--' | '-.' | ':' | 'steps' | ...]
linewidth or lw	float value in points
marker	['+' | ',' | '.' | '1' | '2' | '3' | '4']



Alias	Color
b	Blue
r	Red
c	Cyan
m	Magenta
g	Green
y	Yellow
k	Black
w	White

Plot with (X,Y)

- A leading global organization wants to know how many people visit its website in a particular time. This analysis helps it control and monitor the website traffic.

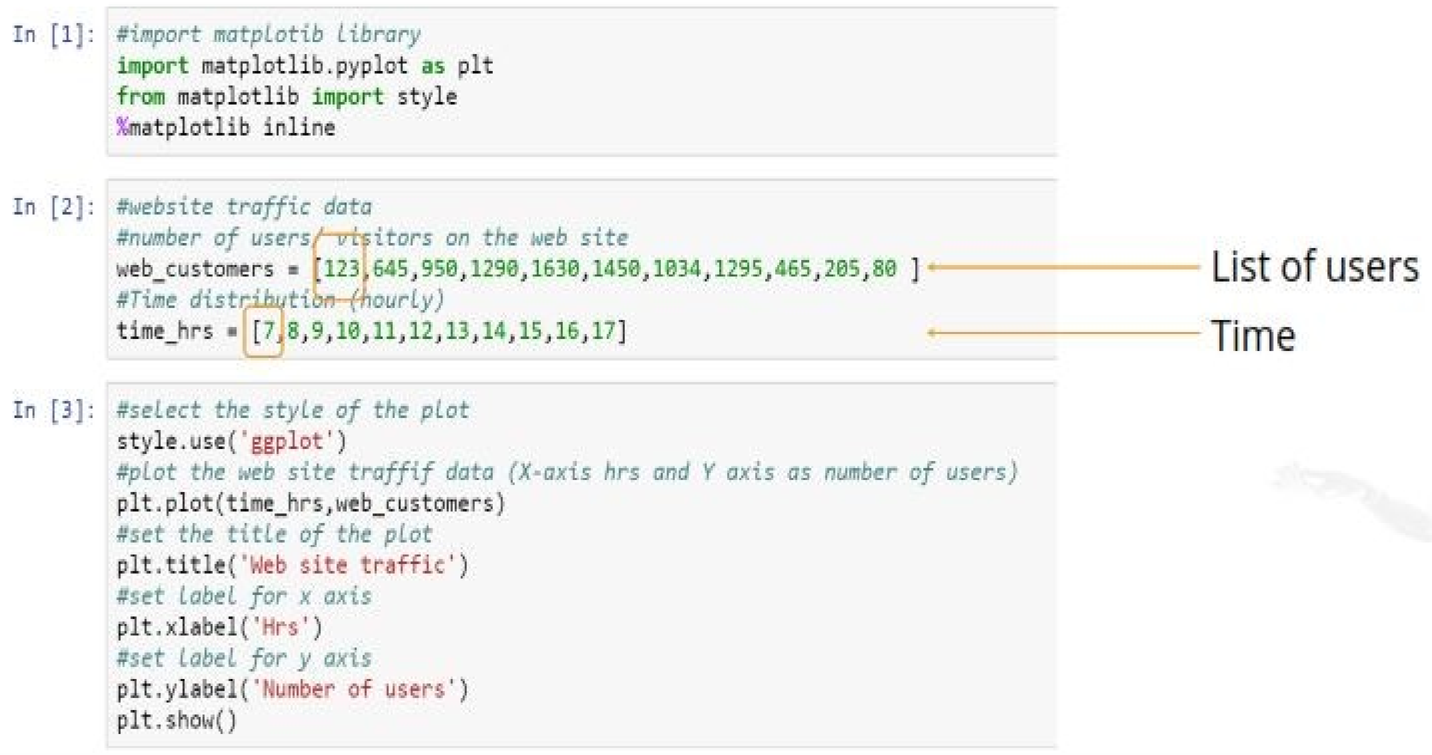


Parul® University

DIGITAL LEARNING CONTENT

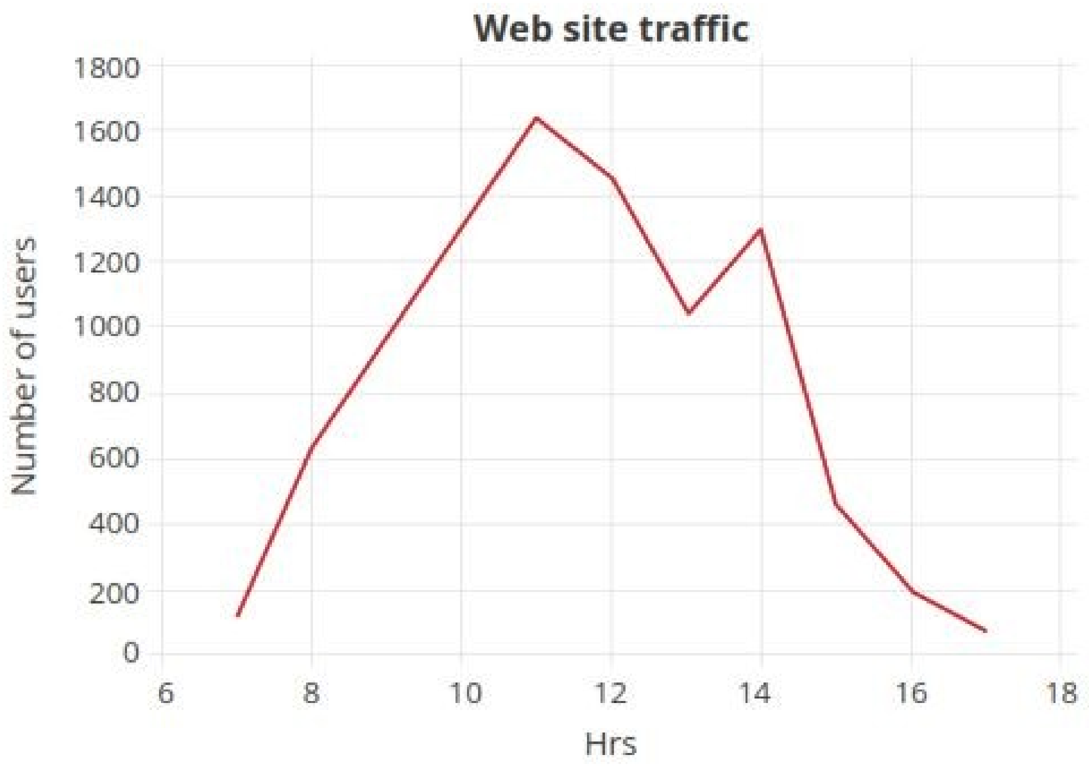
Plot with (X,Y)

- Use %matplotlib inline to display or view the plot on Jupyter notebook.

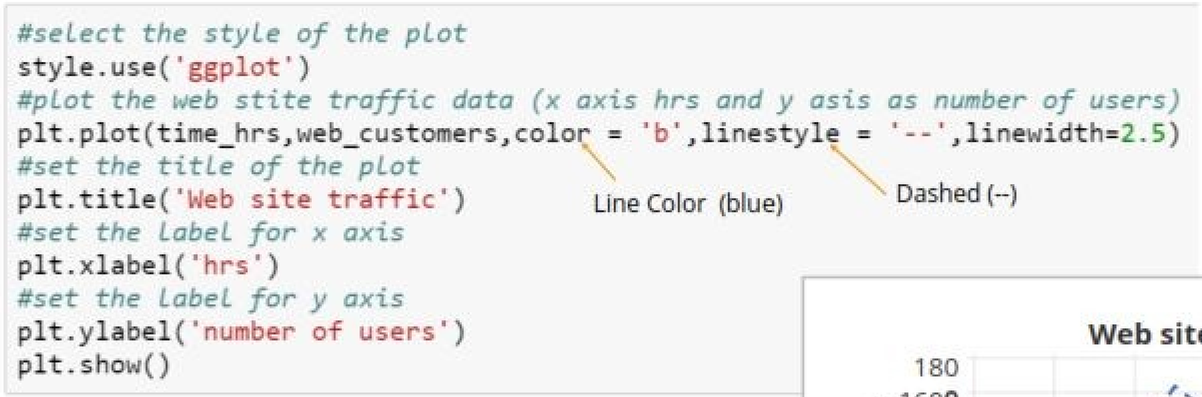


```
In [1]: #import matplotlib library
        import matplotlib.pyplot as plt
        from matplotlib import style
        %matplotlib inline

In [2]: #website traffic data
        #number of users/ visitors on the web site
        web_customers = [123,645,950,1290,1630,1450,1034,1295,465,205,80 ]
        #Time distribution (hourly)
        time_hrs = [7,8,9,10,11,12,13,14,15,16,17]

In [3]: #select the style of the plot
        style.use('ggplot')
        #plot the web site traffif data (X-axis hrs and Y axis as number of users)
        plt.plot(time_hrs,web_customers)
        #set the title of the plot
        plt.title('Web site traffic')
        #set label for x axis
        plt.xlabel('Hrs')
        #set label for y axis
        plt.ylabel('Number of users')
        plt.show()
```

Plot with (X,Y)


Web site traffic
Number of users
Hrs
0
200
400
600
800
1000
1200
1400
1600
1800
6
8
10
12
14
16
18

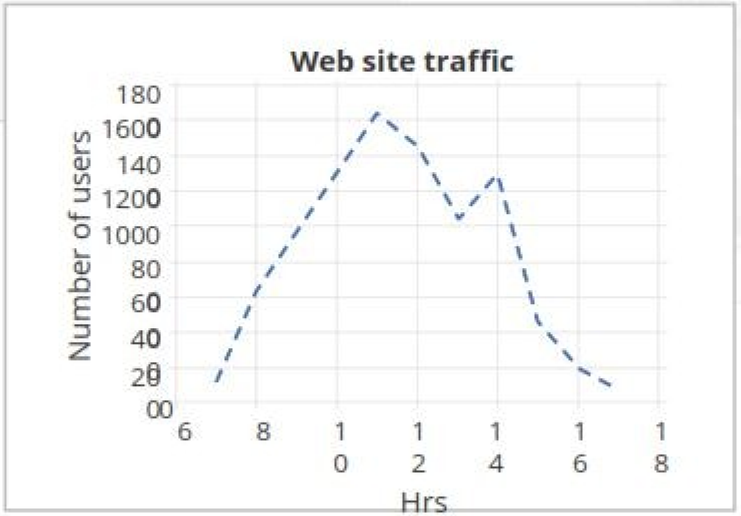
Controlling Line Patterns and Colors



```
#select the style of the plot
style.use('ggplot')
#plot the web stite traffic data (x axis hrs and y asis as number of users)
plt.plot(time_hrs,web_customers,color = 'b',linestyle = '--',linewidth=2.5)
#set the title of the plot
plt.title('Web site traffic')
#set the label for x axis
plt.xlabel('hrs')
#set the label for y axis
plt.ylabel('number of users')
plt.show()
```

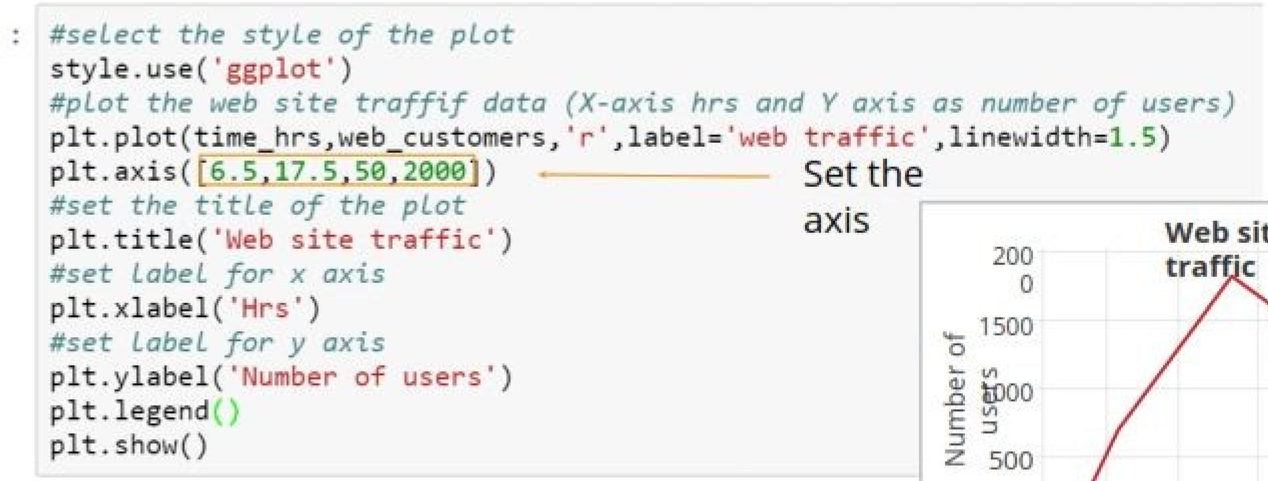
Line Color (blue)

Dashed (--)



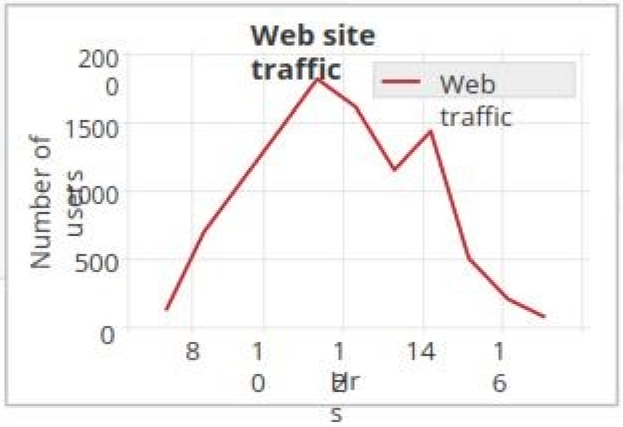
Set Axis, Labels, and Legend Property

- Using matplotlib, it is also possible to set the desired axis to interpret the result.
- Axis is used to define the range on the x axis and y axis.



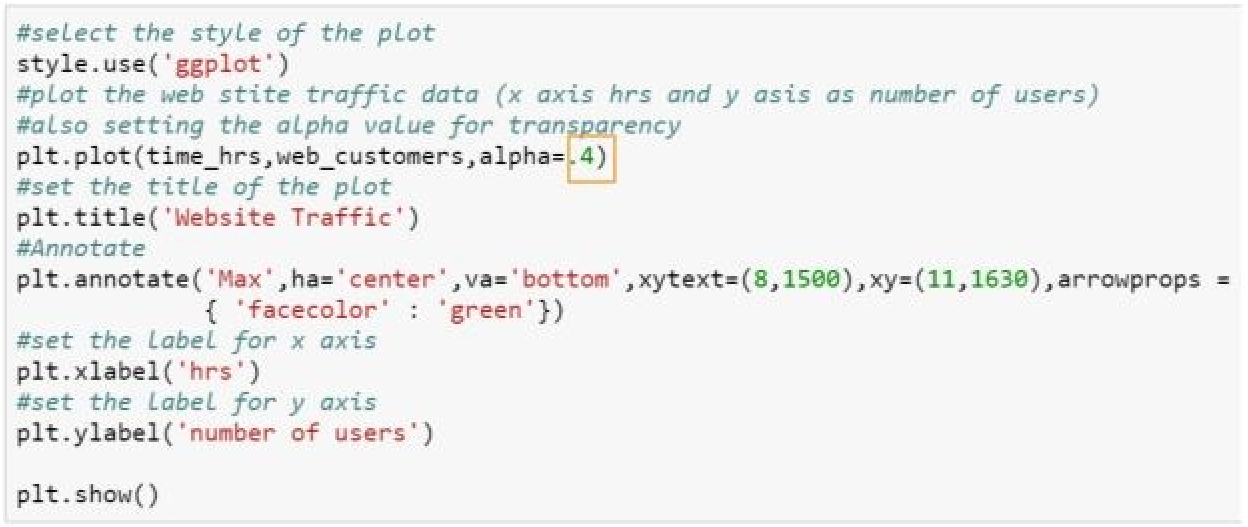
```
#select the style of the plot
style.use('ggplot')
#plot the web site traffif data (X-axis hrs and Y axis as number of users)
plt.plot(time_hrs,web_customers,'r',label='web traffic',linewidth=1.5)
plt.axis([6.5,17.5,50,2000])
#set the title of the plot
plt.title('Web site traffic')
#set label for x axis
plt.xlabel('Hrs')
#set label for y axis
plt.ylabel('Number of users')
plt.legend()
plt.show()
```

Set the axis



Alpha and Annotation

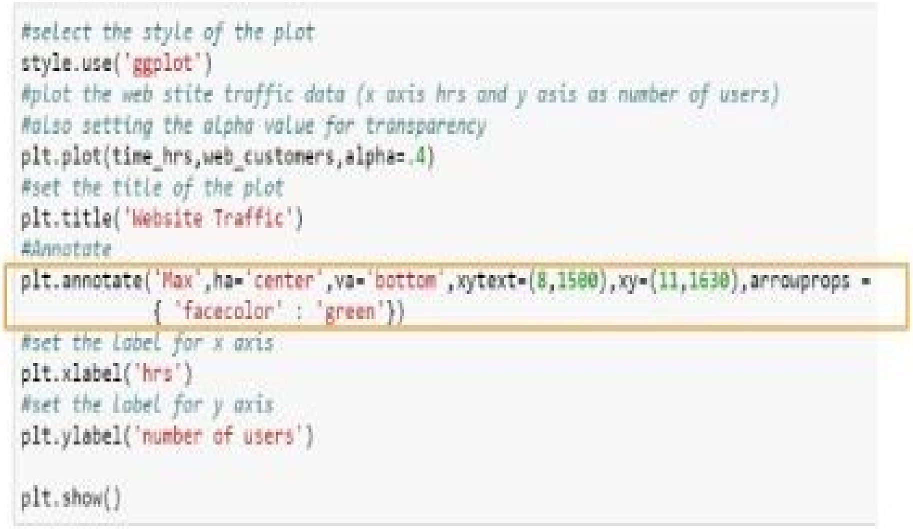
- Alpha is an attribute that controls the transparency of the line.
- The lower the alpha value, the more transparent the line is.



```
#select the style of the plot
style.use('ggplot')
#plot the web stite traffic data (x axis hrs and y asis as number of users)
#also setting the alpha value for transparency
plt.plot(time_hrs,web_customers,alpha=.4)
#set the title of the plot
plt.title('Website Traffic')
#Annotate
plt.annotate('Max',ha='center',va='bottom',xytext=(8,1500),xy=(11,1630),arrowprops =
             { 'facecolor' : 'green'})
#set the label for x axis
plt.xlabel('hrs')
#set the label for y axis
plt.ylabel('number of users')

plt.show()
```

Alpha and Annotation

- Annotate() method is used to annotate the graph. It has several attributes which help annotate the plot.



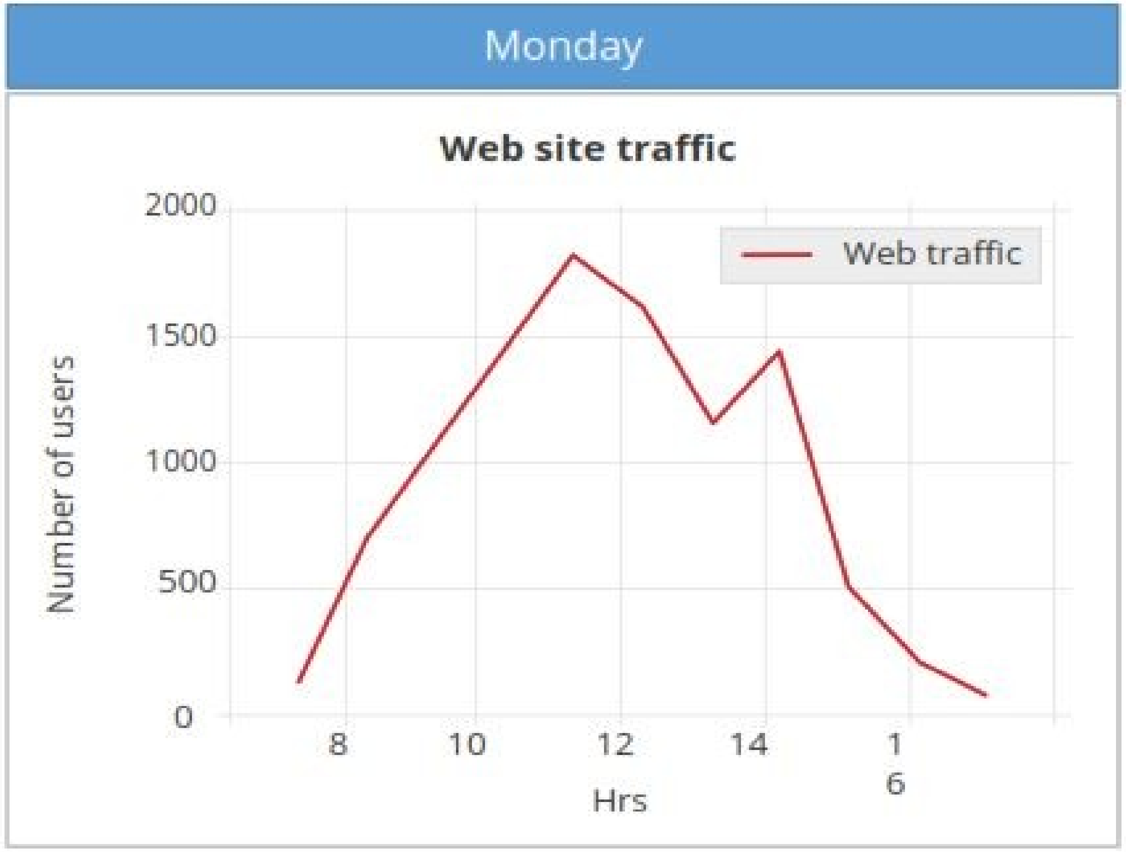
```
#select the style of the plot
style.use('ggplot')
#plot the web stite traffic data (x axis hrs and y asis as number of users)
#also setting the alpha value for transparency
plt.plot(time_hrs,web_customers,alpha=.4)
#set the title of the plot
plt.title('Website Traffic')
#Annotate
plt.annotate('Max',ha='center',va='bottom',xytext=(8,1500),xy=(11,1630),arrowprops =
            { 'facecolor' : 'green'})
#set the label for x axis
plt.xlabel('hrs')
#set the label for y axis
plt.ylabel('number of users')

plt.show()
```

"Max" denotes the annotation text,
"ha" indicates the horizontal alignment,
"va" indicates the vertical alignment,
"xytext" indicates the text position,
"xy" indicates the arrow position, and
"arrowprops" indicates the properties of the arrow.

Parul® University

DIGITAL LEARNING CONTENT

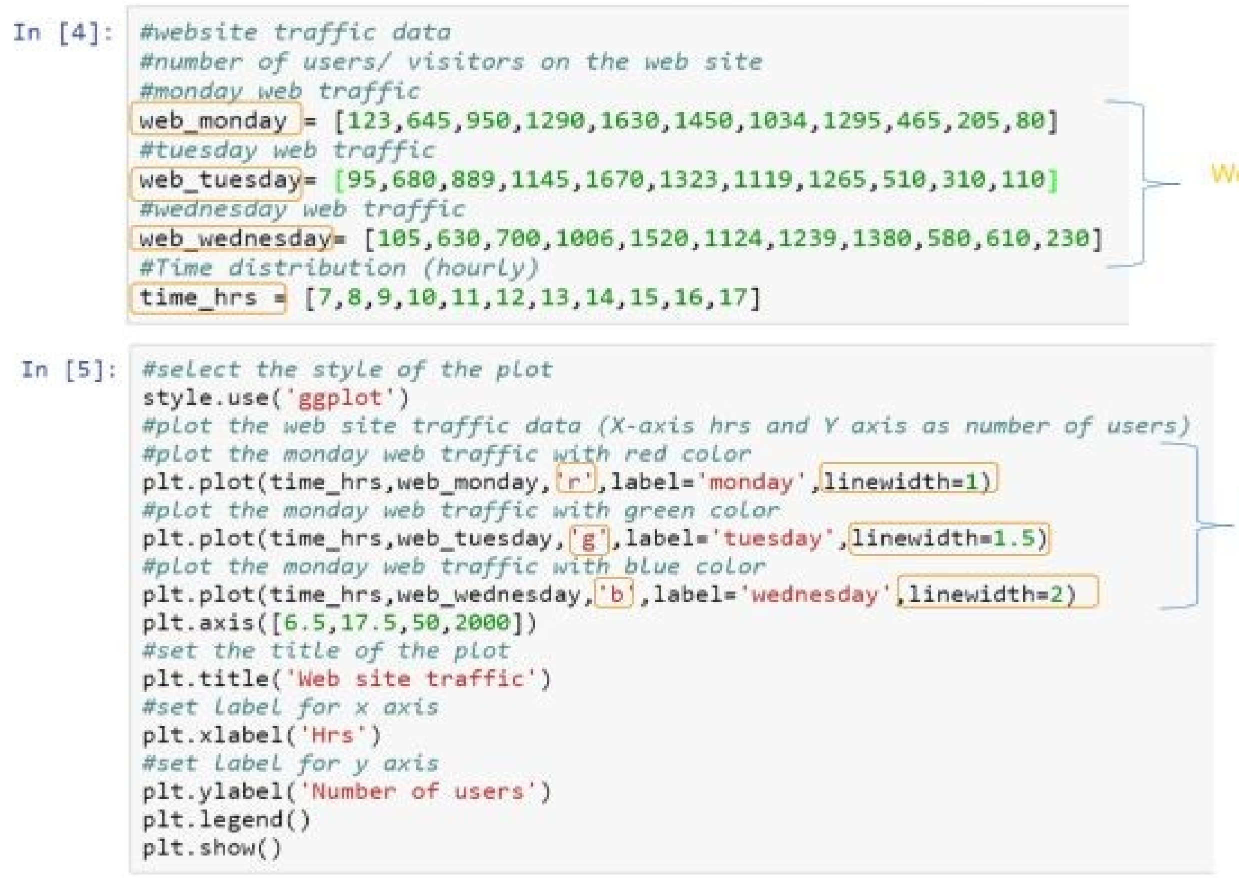
Multiple Plots



Parul® University

DIGITAL LEARNING CONTENT

Multiple Plots



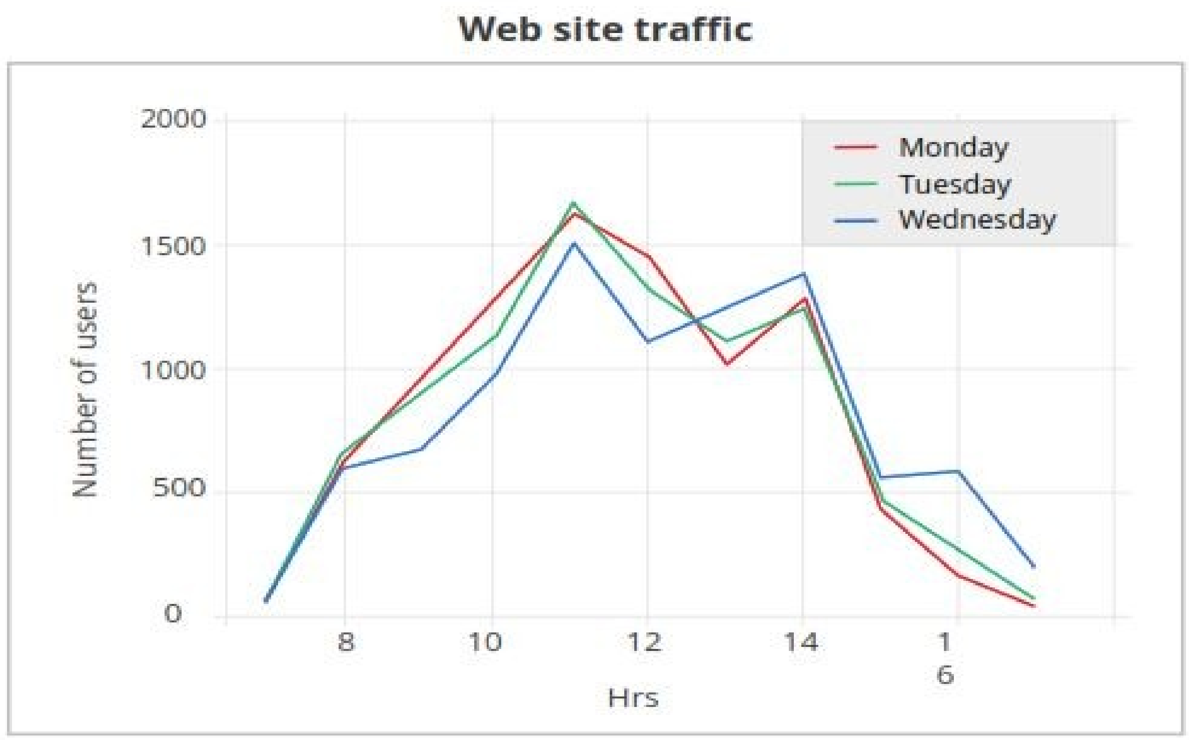
```
In [4]: #website traffic data
        #number of users/ visitors on the web site
        #monday web traffic
        web_monday = [123,645,950,1290,1630,1450,1034,1295,465,205,80]
        #tuesday web traffic
        web_tuesday= [95,680,889,1145,1670,1323,1119,1265,510,310,110]
        #wednesday web traffic
        web_wednesday= [105,630,700,1006,1520,1124,1239,1380,580,610,230]
        #Time distribution (hourly)
        time_hrs = [7,8,9,10,11,12,13,14,15,16,17]
```

Web traffic data

```
In [5]: #select the style of the plot
        style.use('ggplot')
        #plot the web site traffic data (X-axis hrs and Y axis as number of users)
        #plot the monday web traffic with red color
        plt.plot(time_hrs,web_monday,'r',label='monday',linewidth=1)
        #plot the monday web traffic with green color
        plt.plot(time_hrs,web_tuesday,'g',label='tuesday',linewidth=1.5)
        #plot the monday web traffic with blue color
        plt.plot(time_hrs,web_wednesday,'b',label='wednesday',linewidth=2)
        plt.axis([6.5,17.5,50,2000])
        #set the title of the plot
        plt.title('Web site traffic')
        #set label for x axis
        plt.xlabel('Hrs')
        #set label for y axis
        plt.ylabel('Number of users')
        plt.legend()
        plt.show()
```

Set different colors and line widths for different days

Multiple Plots



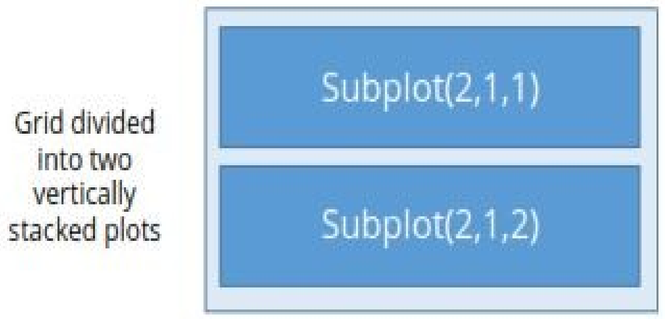
Subplots

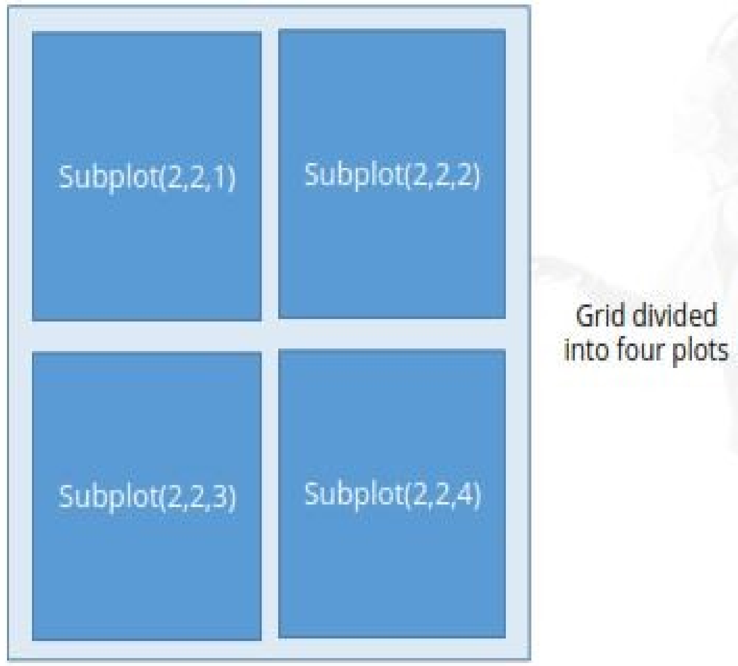
- Subplots are used to display multiple plots in the same window.
- With subplot, you can arrange plots in a regular grid.
- The syntax for subplot is
- subplot(m,n,p). →

It divides the current window into an m-by-n grid and creates an axis for a subplot in the position specified by p.

For example,
subplot(2,1,2) creates two subplots which are stacked vertically on a grid.
subplot(2,1,4) creates four subplots in one window.

Subplots


Grid divided into two vertically stacked plots
Subplot(2,1,1)
Subplot(2,1,2)


Subplot(2,2,1)
Subplot(2,2,2)
Subplot(2,2,3)
Subplot(2,2,4)
Grid divided into four plots

Layout

- Layout and spacing adjustments are two important factors to be considered while creating subplots.
- Use the plt.subplots_adjust() method with the parameters hspace and wspace to adjust the distances between the subplots and move them around on the grid.

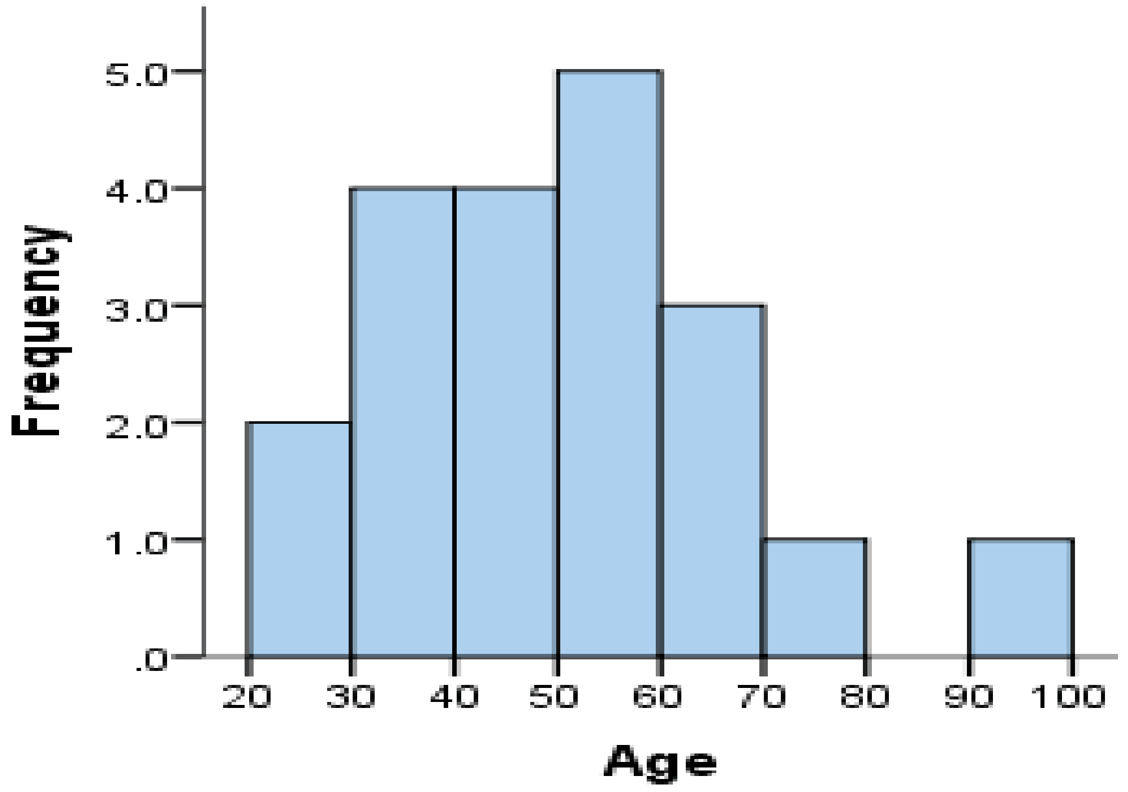
Types of Plots

- You can create different types of plots using matplotlib:
- Histogram:-
- Histograms are graphical representations of a Histogram probability distribution. A histogram is a kind of a bar chart.
- Using matplotlib and its bar chart function, you can create histogram charts.
- Advantages of Histogram charts:
- They display the number of values within a specified interval.
- They are suitable for large datasets as they can be grouped within the intervals.

Parul® University

DIGITAL LEARNING CONTENT

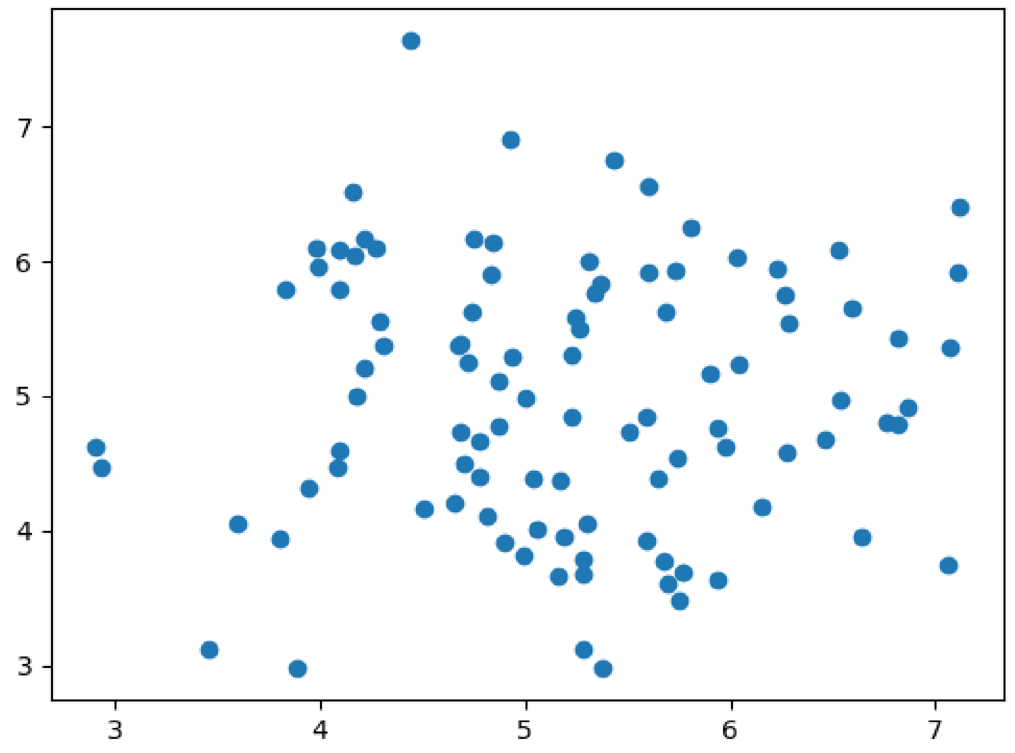
Histogram: Example



Scatter Plot

- A scatter plot is used to graphically display the relationships between variables.
- However, to control a plot, it is recommended to use scatter() method.
- It has several advantages:
- Shows the correlation between variables
- Is suitable for large datasets
- Is easy to find clusters
- Is possible to represent each piece of data as a point on the plot

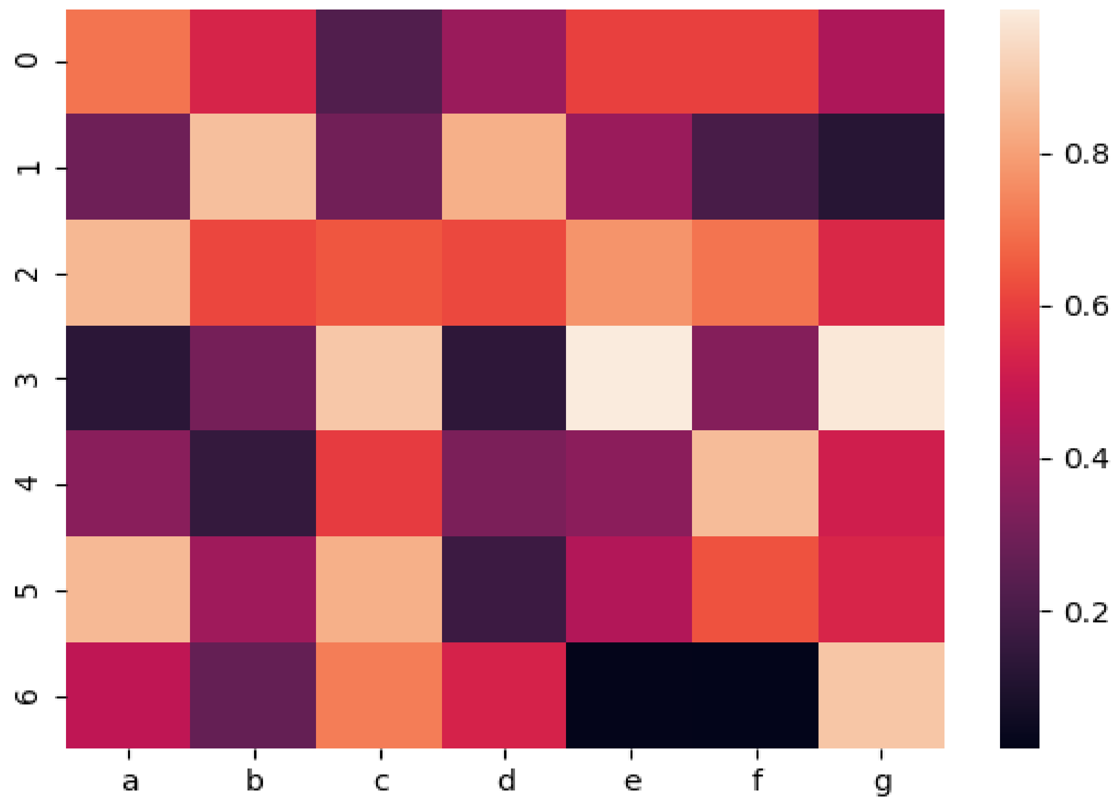
Scatter Plot: Example



Heat Map

- A heat map is a way to visualize two-dimensional data. Using heat maps, you can gain deeper and faster insights about data than other types of plots.
- It has several advantages:
- Draws attention to the risk-prone area
- Uses the entire dataset to draw meaningful insights
- Is used for cluster analysis and can deal with large datasets

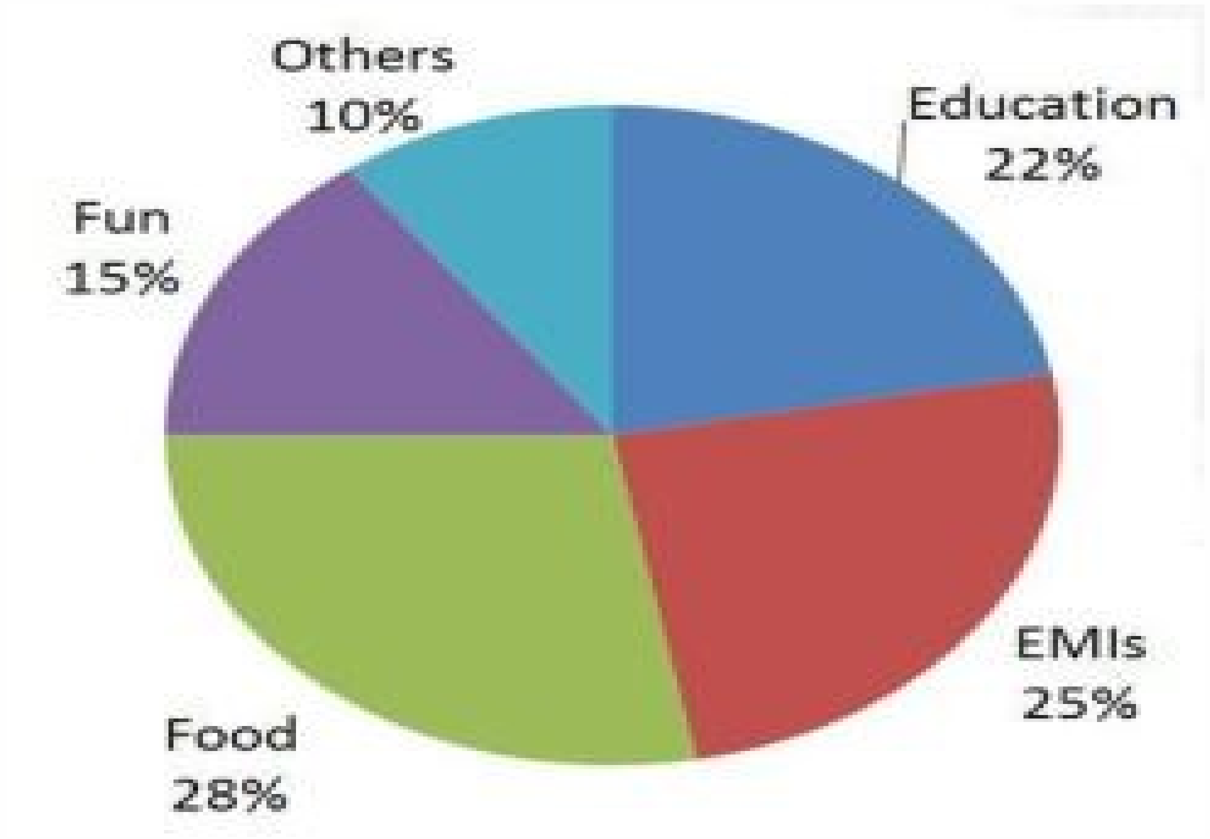
Heat Map: Example



Pie Chart

- Pie charts are used to show percentage or proportional data. matplotlib provides the pie() method to create pie charts.
- It has several advantages:
- Summarizes a large dataset in visual form
- Displays the relative proportions of multiple classes of data
- Size of the circle is made proportional to the total quantity

Pie Chart : Example



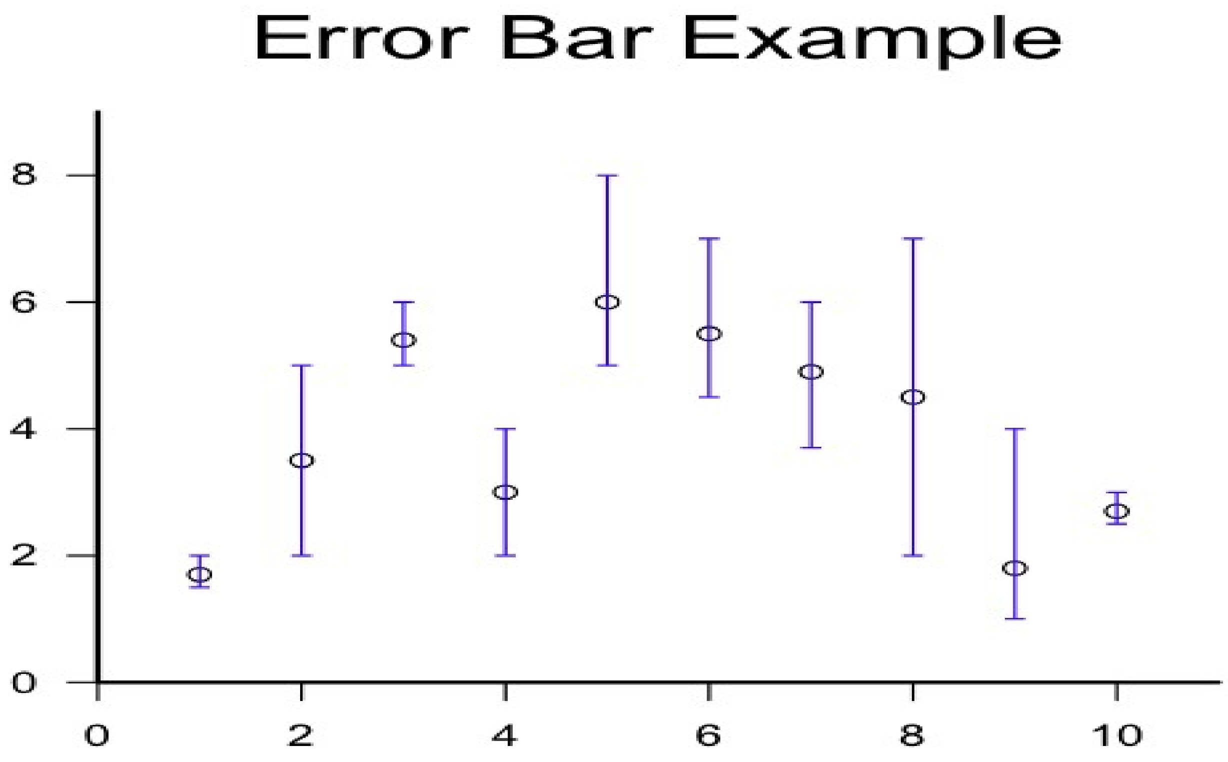
Error Bar

- An error bar is used to graphically represent the variability of data. It is used mainly to identify errors. It builds confidence about the data analysis by revealing the statistical difference between the two groups of data.
- It has several advantages:
- Shows the variability in data and indicates the errors.
- Depicts the precision in the data analysis.
- Demonstrates how well a function and model are used in the data analysis.
- Describes the underlying data.

Parul® University

DIGITAL LEARNING CONTENT

Error Bar: Example



Seaborn

- Seaborn is a Python visualization library based on matplotlib. It provides a high-level interface to draw attractive statistical graphics.
- There are several advantages:
- Possesses built-in themes for better visualizations
- Has built-in statistical functions which reveal hidden patterns in the dataset
- Has functions to visualize matrices of data

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.in