Data Visualization using Pyplot

As per CBSE curriculum
Class 12

Chapter- 08

What is Data Visualization?

- As we know it is an era of Big Data,

- And this Data is very important for any organization for decision making.

- Visualization techniques of such big data are very important for the purpose of analysis of data.

- "Data Visualization basically refers to the graphical or visual representation of data using visual elements like chart, graph and map etc.

Data Visualization

"Data Visualization basically refers to the graphical or visual representation of information and data using visual elements like charts, graphs or maps.

- In this chapter we will come to know about Pyplot in Python.

- We will also come to know about the visualization of data using Pyplot.

Use of Pyplot of MATPLOTLIB Library

- The Matplotlib is a python library that provides many interfaces and functionality for 2D-graphics similar to MATLAB.

- We can call it as high quality ploting library of python.

- Matplotlib library offers many different named collections of methods; Pyplot is one such interface.

- Pyplot is a collection of methods within matplotlib which allow us to construct 2D plots easily and interactively.

Installing and importing Matplotlib


```
Command Prompt

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

C:\Users\SHAURYA>pip install matplotlib
Collecting matplotlib
  Using cached matplotlib-1.5.1-cp27-none-win32.whl
Collecting numpy>=1.6 (from matplotlib)
  Using cached numpy-1.11.1-cp27-none-win32.whl
Collecting python-dateutil (from matplotlib)
  Using cached python_dateutil-2.5.3-py2.py3-none-any.whl
Collecting pytz (from matplotlib)
  Using cached pytz-2016.6.1-py2.py3-none-any.whl
Collecting cycler (from matplotlib)
  Using cached cycler-0.10.0-py2.py3-none-any.whl
Collecting pyparsing!=2.0.4,>=1.5.6 (from matplotlib)
  Using cached pyparsing-2.1.5-py2.py3-none-any.whl
Collecting six>=1.5 (from python-dateutil->matplotlib)
  Using cached six-1.10.0-py2.py3-none-any.whl
Installing collected packages: numpy, six, python-dateutil, pytz, cycler, pypars
ing, matplotlib
```

padhaaii

Importing Pyplot

- Following syntax need to write to import Pyplot

import matplotlib.pyplot

OR

import matplotlib.pyplot as pl

- We will use commands afterwards using pl with (.).

- Before proceeding we need to know something about numpy.

- Numpy provides very useful functions for ploting.

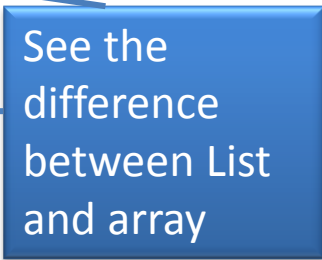- Numpy also supports vectorized functions.

NumPy Arrays

- NumPy ("Numerical Python" or Numeric Python") is an open source module of Python which provides functions for arrays and matrices.

- NumPy is needed to import for its use. The statements for the same is as follows-

  >>>import numpy as np

  (np is another name for numpy which is optional.

- NumPy arrays is of two types-

  - 1-D array – also known as Vectors.

 - Multidimentional arrays –
   also known as Matrices.

```
>>> import numpy as np
>>> lst = [1,2,3,4]
>>> a1=np.array(lst)
>>> lst
[1, 2, 3, 4]
>>> print(a1)
[1 2 3 4]
>>> a1
array([1, 2,
```

See the difference between List and array

Basics of Simple Plotting

- Graphical representation of compiled data is known as data visualization.

- Chart and Graph are very important tools for data visualization.

- Pyplot can be used for developing various types of graphs and charts.

- We will go through following charts in syllabus-
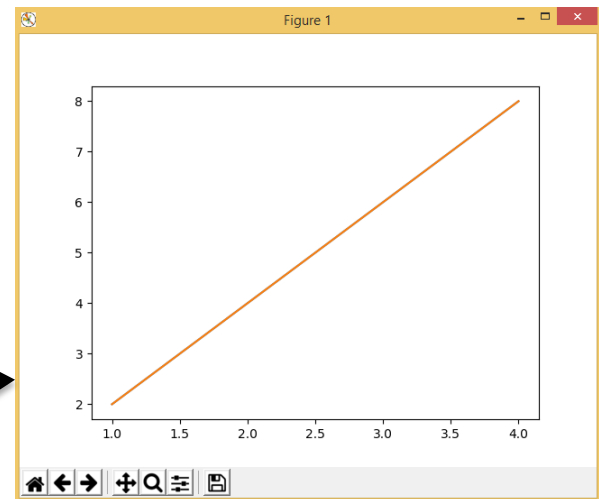 - Line chart
  - Bar Chart
  - Pie Chart
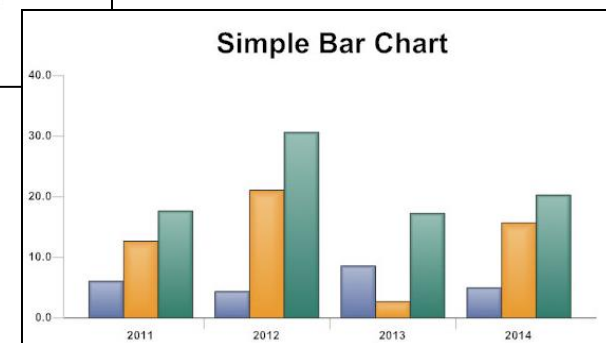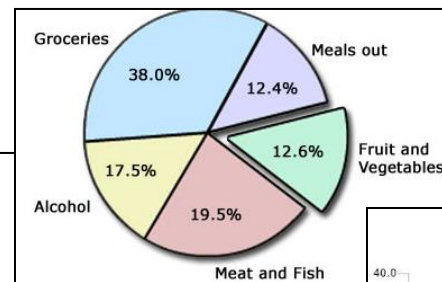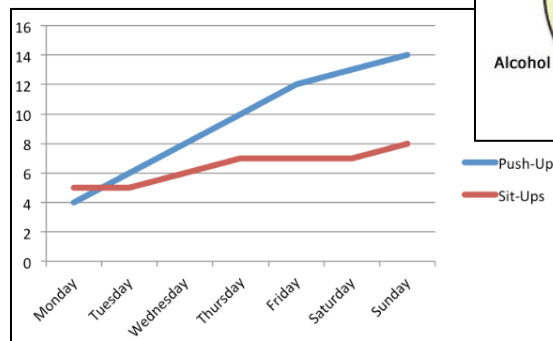
Creating Line Chart

- A line chart or line graph is a type of chart which displays information as a series of data points called 'markers' connected by a straight line segments.

- The pyplot interface offers plot() function for creating a line graph. -

```
>>> import matplotlib.pyplot as pl
>>> a=[1,2,3,4]
>>> b=[2,4,6,8]
>>> pl.plot(a,b)
```

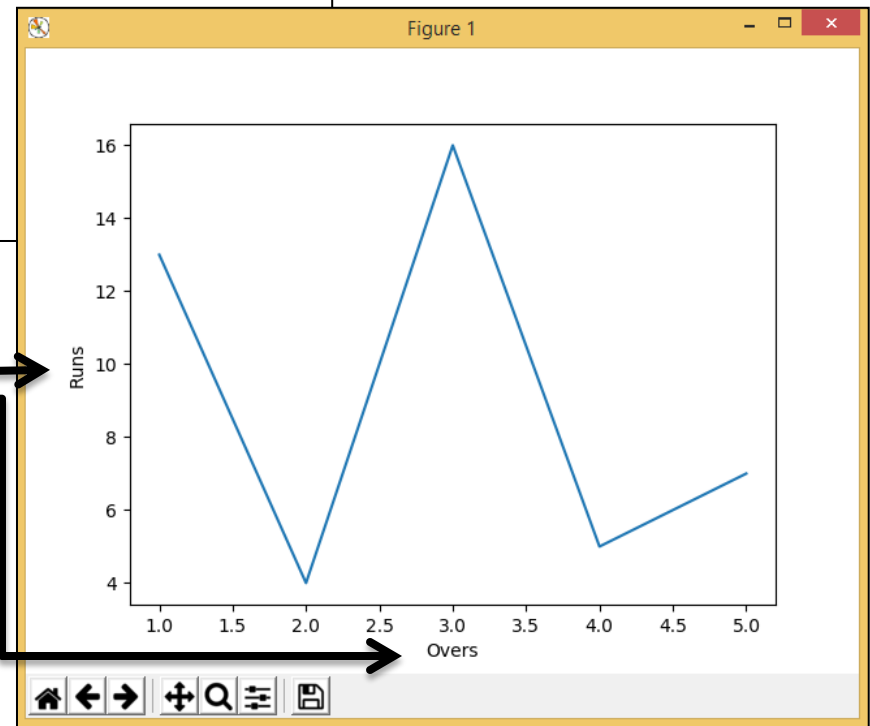`[<matplotlib.lines.Line2D object at 0x00000021D8F979E8>]`

```
>>> pl.show()
```

Creating Line Chart

- Let's take an example – here we have data of runs made in 5 overs. We will name X axis as overs and Y axis as runs.

```python
import matplotlib.pyplot as pl
over = [1,2,3,4,5]
run = [13,4,16,5,7]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.plot(over,run)
pl.show()
```

We will use these functions for labeling.

Labels are shown in resultant chart.

Figure 1

Setting of Line color, width and style
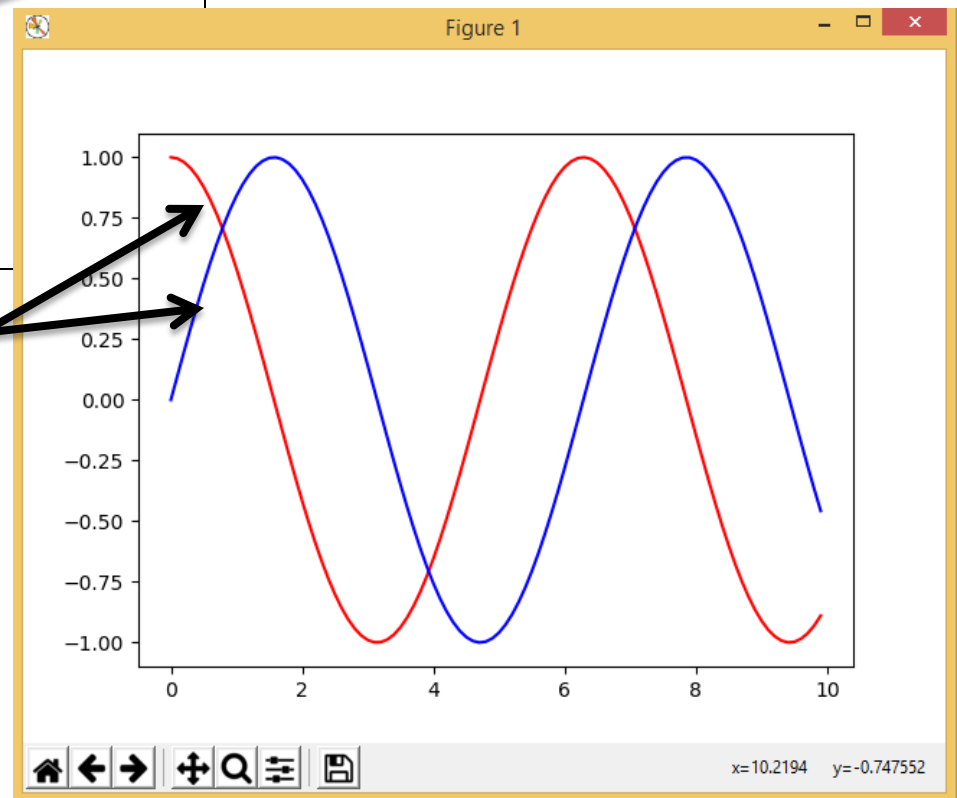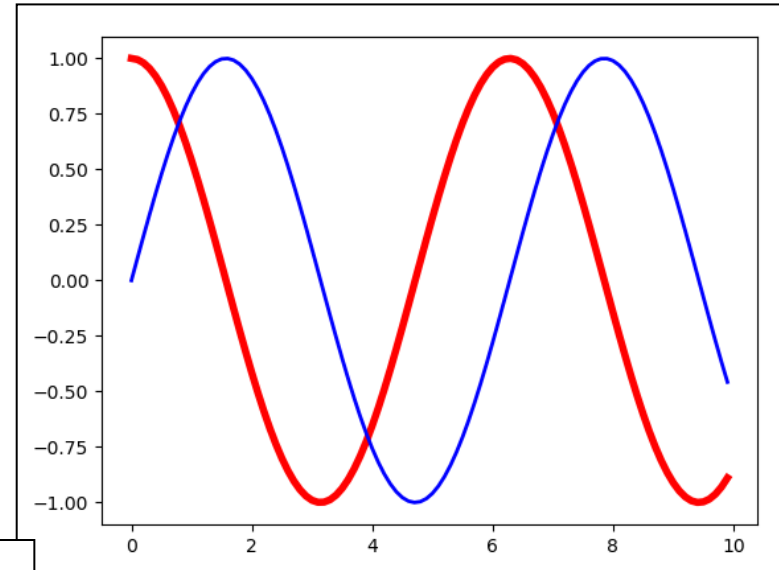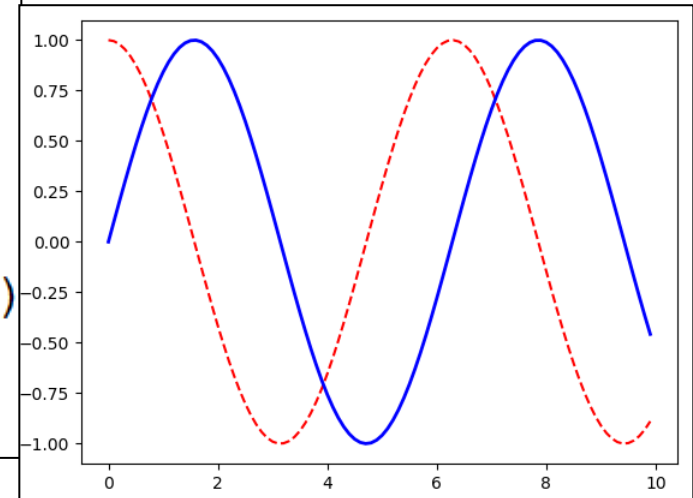
- It has following syntax -

  matplotlib.pyplot.plot(<data1>,<data2>,<color code>)

```
import matplotlib.pyplot as pl
import numpy as np
x=np.arange(0,10,0.1)
a=np.cos(x)
b=np.sin(x)
pl.plot(x,a,'r')
pl.plot(x,b,'b')
pl.show()
```

'r' is used for Red color and 'b' is used for blue color.

Both the colors are shown in resultant chart.

character	color
'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

Figure 1

x=10.2194   y=-0.747552

padhaaii

Changing Line color, width and style

- It has following syntax -

  matplotlib.pyplot.plot(<data1>,<data2>,linewidth=<val>…)

```python
import matplotlib.pyplot as pl
import numpy as np
x=np.arange(0,10,0.1)
a=np.cos(x)
b=np.sin(x)
pl.plot(x,a,'r',linewidth=4)
pl.plot(x,b,'b',linewidth=2)
pl.show()
```



```python
import matplotlib.pyplot as pl
import numpy as np
x=np.arange(0,10,0.1)
a=np.cos(x)
b=np.sin(x)
pl.plot(x,a,'r',linestyle='dashed')
pl.plot(x,b,'b',linewidth=2)
pl.show()
```

> Use ':' , '-' , '--', '-.' for different line styles.

Changing Marker type, size and color

-It has following syntax -

matplotlib.pyplot.plot(<data1>,<data2>,linestyle=<val>…)

```
import matplotlib.pyplot as pl
over=[1,2,3,4,5]
run=[13,5,7,16,4]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.plot(over,run,'r',marker='d', markersize=6,markeredgecolor='red')
pl.show()
```

https://matplotlib.org/2.1.1/api/_as_gen/matplotlib.pyplot.plot.html   A use full Link to understand pyplot

character	description
'-'	solid line style
'--'	dashed line style
'-.'	dash-dot line style
':'	dotted line style
'.'	point marker
','	pixel marker
'o'	circle marker
'v'	triangle_down marker
'^'	triangle_up marker
'<'	triangle_left marker
'>'	triangle_right marker
'1'	tri_down marker
'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
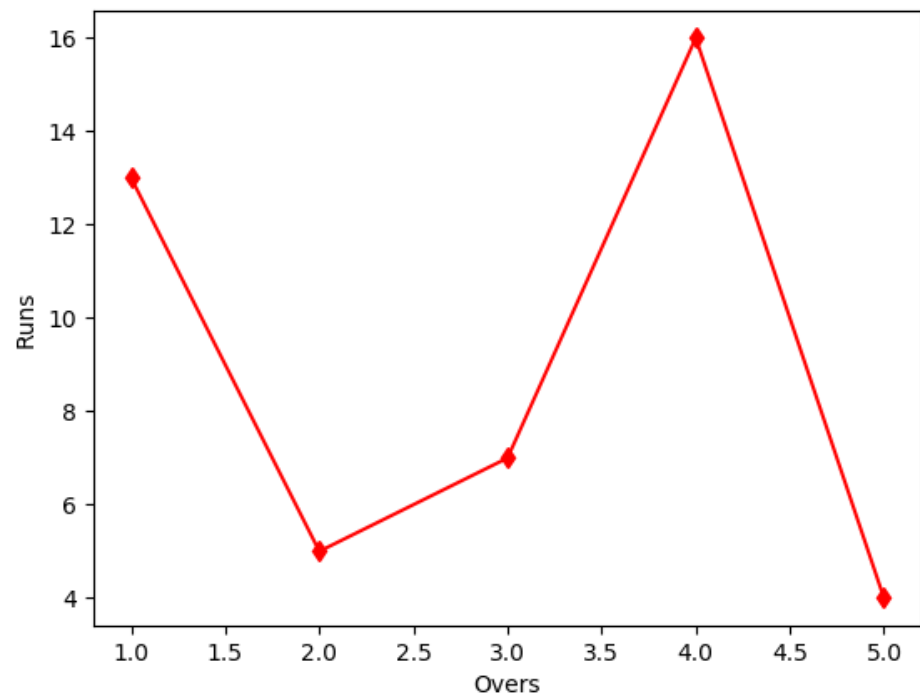
Creating Bar Chart

- A Bar Graph /Chart a graphical display of data using bars of different heights. Syntax is– matplotlib.pyplot.bar(a,b)

```python
import matplotlib.pyplot as pl
over=[1,2,3,4,5]
run=[13,5,7,16,4]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.bar(over,run)
pl.show()
```

These functions are used for labeling.
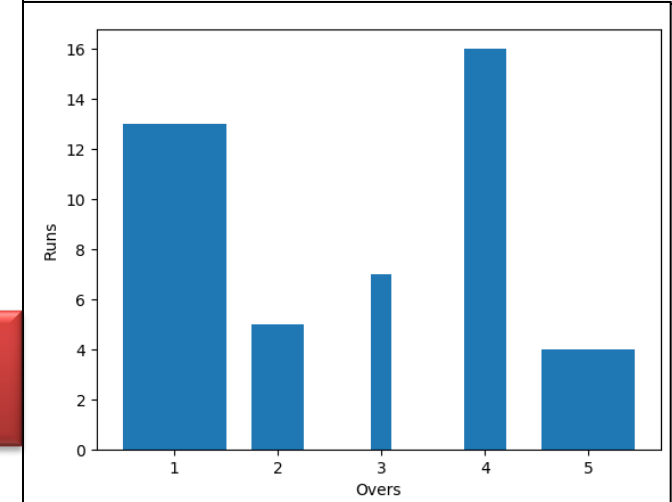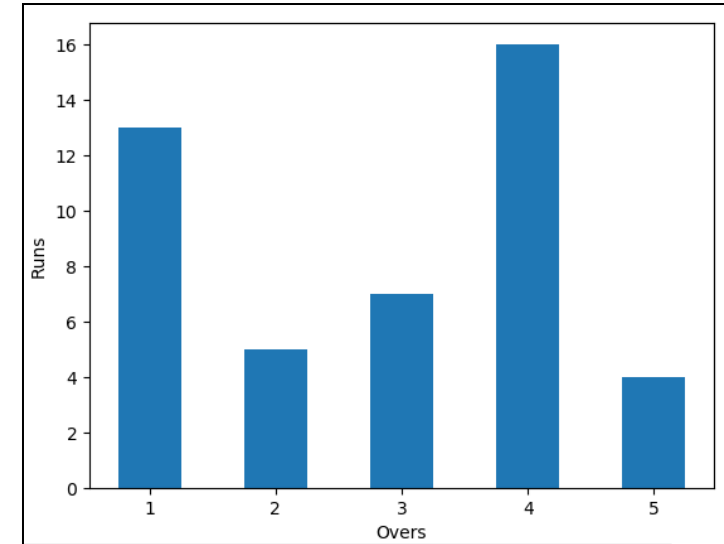
Labels are shown in resultant chart.

padhaaii
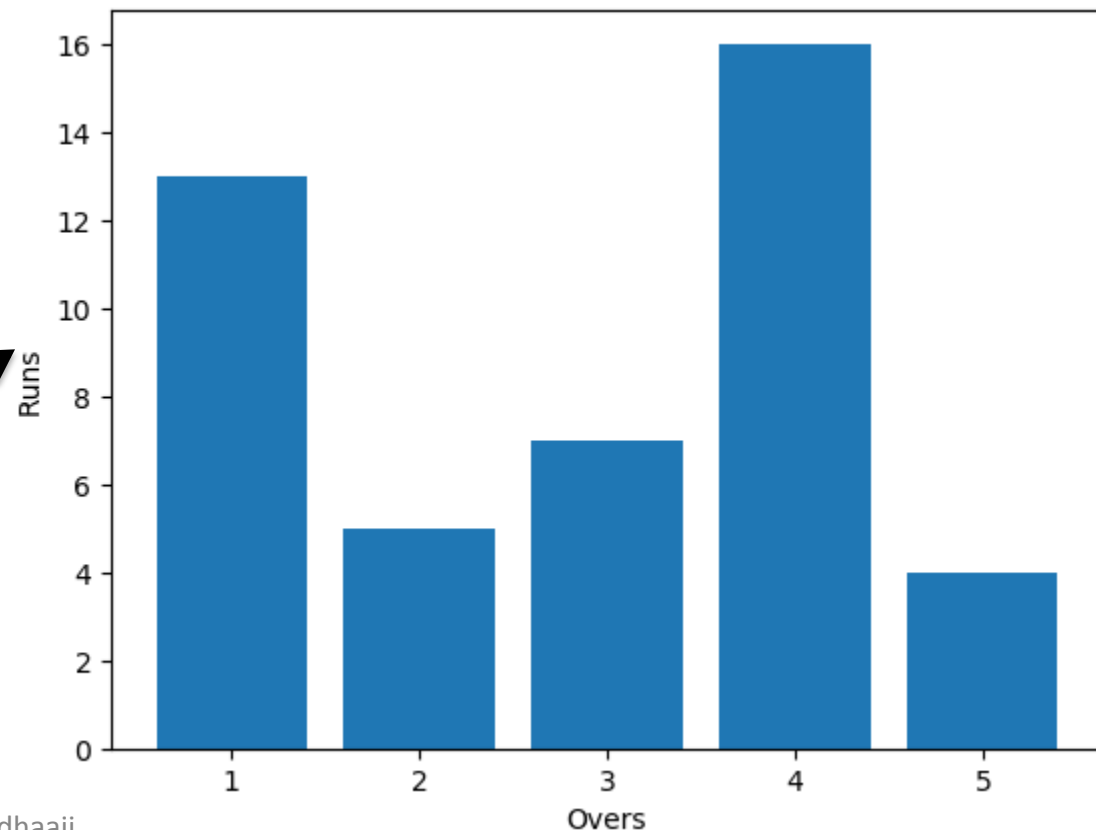
Changing Bar width

- A Bar Graph /Chart a graphical display of data using bars of different heights. Syntax is–

matplotlib.pyplot.bar(a, b, width=<Value>)

```python
import matplotlib.pyplot as pl
over=[1,2,3,4,5]
run=[13,5,7,16,4]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.bar(over,run,width=1/2)
pl.show()
```



```python
import matplotlib.pyplot as pl
over=[1,2,3,4,5]
run=[13,5,7,16,4]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.bar(over,run,width=[1,0.5,0.2,0.4,0.9])
pl.show()
```


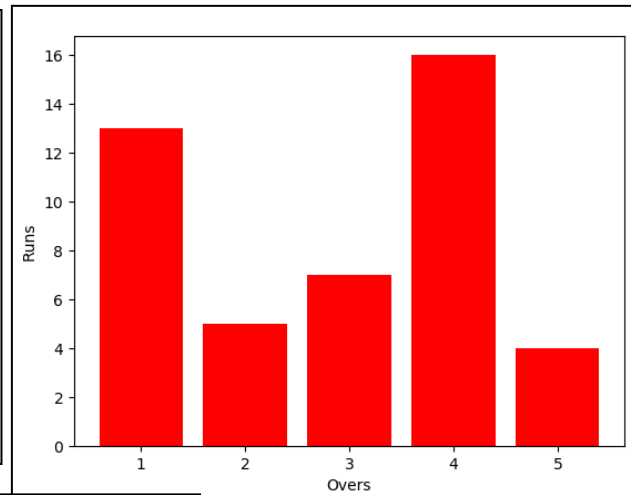It is also possible to set the different width of bars for different data.

padhaaii

# Changing Bar color

- A Bar Graph /Chart a graphical display of data using bars of different heights. Syntax is–

matplotlib.pyplot.bar(a, b, color=<code>)

```python
import matplotlib.pyplot as pl
over=[1,2,3,4,5]
run=[13,5,7,16,4]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.bar(over,run,color ='red')
pl.show()
```
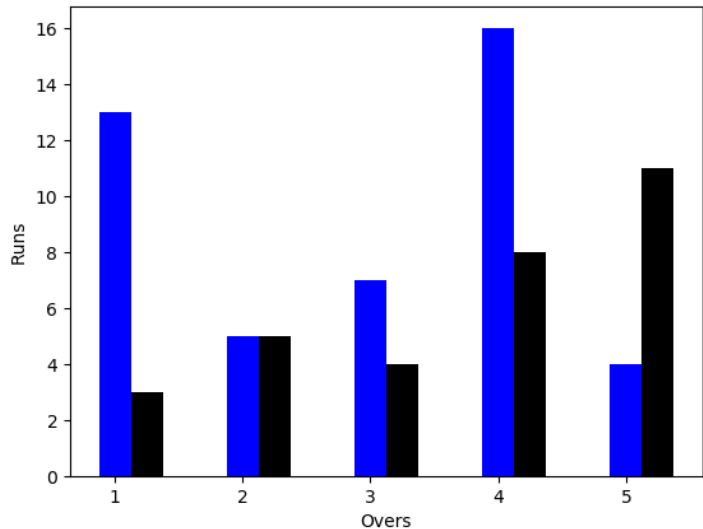
```python
import matplotlib.pyplot as pl
over=[1,2,3,4,5]
run=[13,5,7,16,4]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.bar(over,run,color =['r','g','b','k','c'])
pl.show()
```
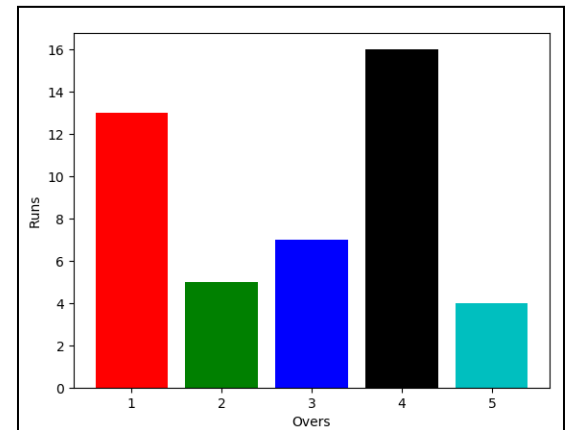
Creating Multiple Bar Chart

```python
import matplotlib.pyplot as pl
import numpy as np
over=np.arange(1.0,6.0,1.0)
Ind=[13,5,7,16,4]
Nz=[3,5,4,8,11]
pl.xlabel("Overs")
pl.ylabel("Runs")
pl.bar(over,Ind,color ='b',width=0.25)
pl.bar(over+0.25,Nz,color='k',width=0.25)
pl.show()
```


The fact to notice here is that the number of times you use bar () function before calling Show() Function, it will be added to the same chart.

padhaaii

Creating Horrizontal Bar Chart

```python
import matplotlib.pyplot as pl
import numpy as np
over=np.arange(1.0,6.0,1.0)
Ind=[13,5,7,16,4]
pl.xlabel("Runs")
pl.ylabel("Overs")
pl.barh(over,Ind,color ='b')
pl.show()
```
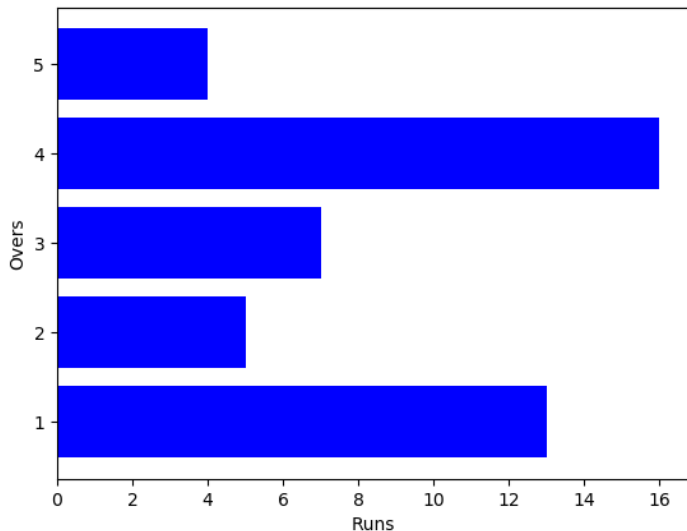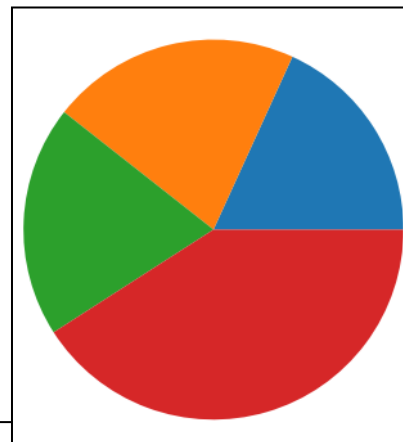
barh() function is being used here.

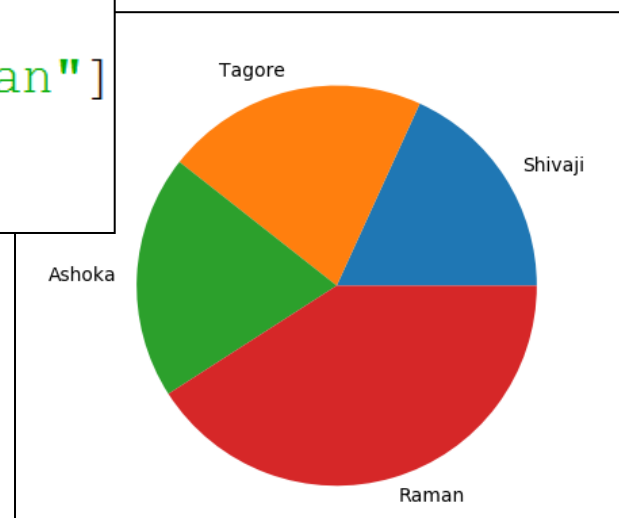# Creating Pie Chart

- This kind of chart is made up of different parts of a circle where each part shows a particular ratio of data. Syntax is–
  matplotlib.pyplot.pie(a,labels)

```
import matplotlib.pyplot as pl
import numpy as np
contri=[12.0,14.0,13.0,27.0]
Houses=["Shivaji","Tagore","Ashoka","Raman"]
pl.pie(contri)
pl.show()
```
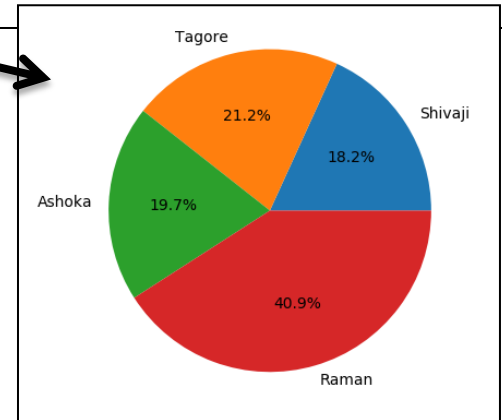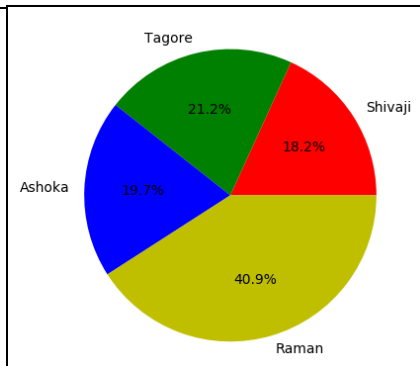


```
import matplotlib.pyplot as pl
import numpy as np
contri=[12.0,14.0,13.0,27.0]
Houses=["Shivaji","Tagore","Ashoka","Raman"]
pl.pie(contri,labels=Houses)
pl.show()
```

|for labeling, code given lower side will be used.

Formatting Pie Chart

```python
import matplotlib.pyplot as pl
import numpy as np
contri=[12.0,14.0,13.0,27.0]
Houses=["Shivaji","Tagore","Ashoka","Raman"]
pl.pie(contri,labels=Houses, autopct="%1.1f%%")
pl.show()
```



```python
import matplotlib.pyplot as pl
import numpy as np
contri=[12.0,14.0,13.0,27.0]
Houses=["Shivaji","Tagore","Ashoka","Raman"]
clr=['r','g','b','Y']
pl.pie(contri,labels=Houses, colors=clr,autopct="%1.1f%%")
pl.show()
```

padhaaii

Slice explode

```python
import matplotlib.pyplot as pl
import numpy as np
contri=[12.0,14.0,13.0,27.0]
expl=[0,0,0.3,0]
Houses=["Shivaji","Tagore","Ashoka","Raman"]
clr=['r','g','b','Y']
pl.pie(contri,labels=Houses, explode=expl,colors=clr,autopct="%1.1f%%")
pl.show()
```

The slice to be explode need to see carefully for the variable name expl. For multiple slicing, valeus are to be started from 0.
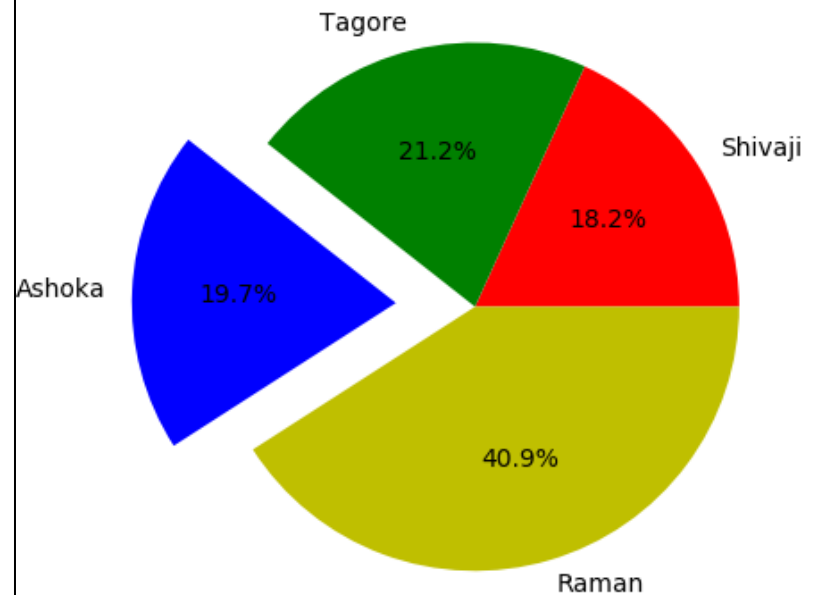
Anatomy of Chart

- Chart has a structure. See the following points-

- Figure – Any chart will be made under this area only. This is the area of plot.

- Axes – This is that area which has actual ploting.
  - Axis Label – This is made up of x-axis and y-axis.
  - Limits – This is the limit of values marked on x-axis and y-axis.
  - Tick Marks – This is the individual value on x-axis and y-axis.

- Title – It is the text to be shown at the top of plot.

- Legends – This is the set of data of different color which is to be used during plotting.

Adding Title and setting xlimit & ylimit

```python
import matplotlib.pyplot as pl
import numpy as np
over=[1,2,3,4,5]
run=[10,3,14,15,4]
pl.xlim(0,10)
pl.title("Cricket Analysis")
pl.bar(over,run,width=1/2)
pl.show()
```

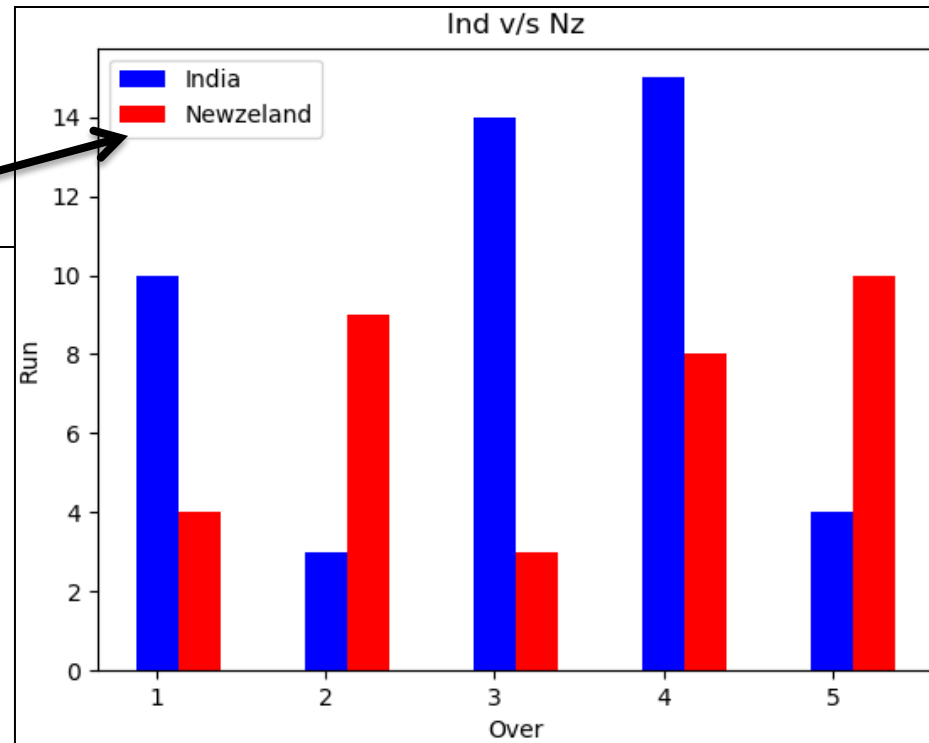pl.title () and pl.xlim () functions are used here.
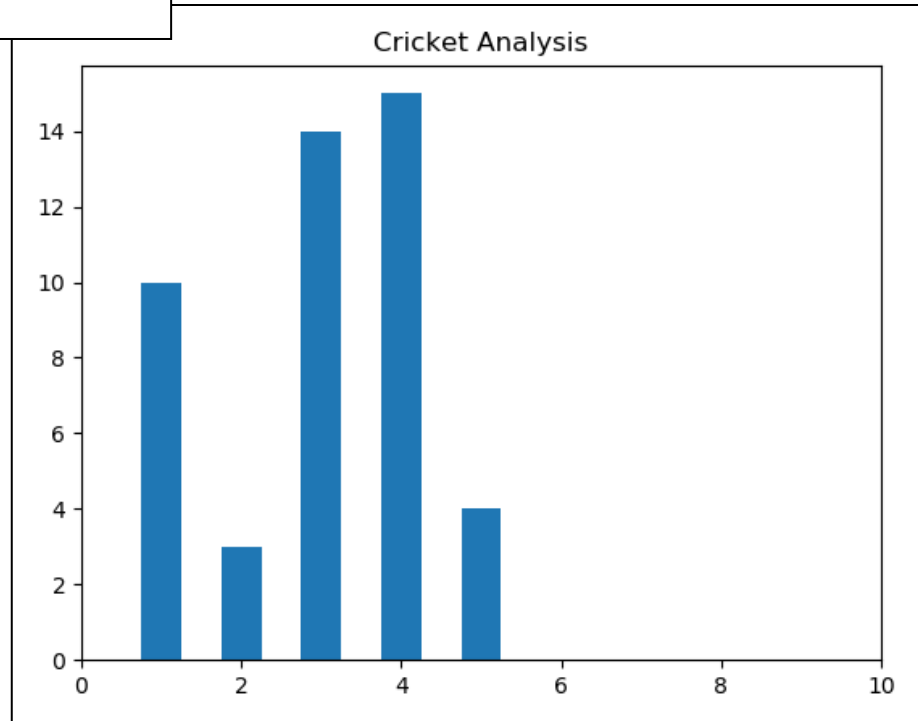

Cricket Analysis

Adding Legends

```python
import matplotlib.pyplot as pl
import numpy as np
over=np.arange(1.0,6.0,1.0)
ind=[10,3,14,15,4]
nz=[4,9,3,8,10]
pl.title("Ind v/s Nz")
pl.bar(over,ind,color='b',width=0.25,label='India')
pl.bar(over+0.25,nz,color='r',width=0.25,label='Newzeland')
pl.legend(loc='upper left')
pl.xlabel("Over")
pl.ylabel("Run")
pl.show()
```

Legends

Saving a Figure

```python
import matplotlib.pyplot as pl
import numpy as np
over=np.arange(1.0,6.0,1.0)
ind=[10,3,14,15,4]
nz=[4,9,3,8,10]
pl.title("Ind v/s Nz")
pl.bar(over,ind,color='b',width=0.25,label='India')
pl.bar(over+0.25,nz,color='r',width=0.25,label='Newzeland')
pl.legend(loc='upper left')
pl.xlabel("Over")
pl.ylabel("Run")
pl.savefig("C:\\MyData\\myfig.png")
pl.show()
```
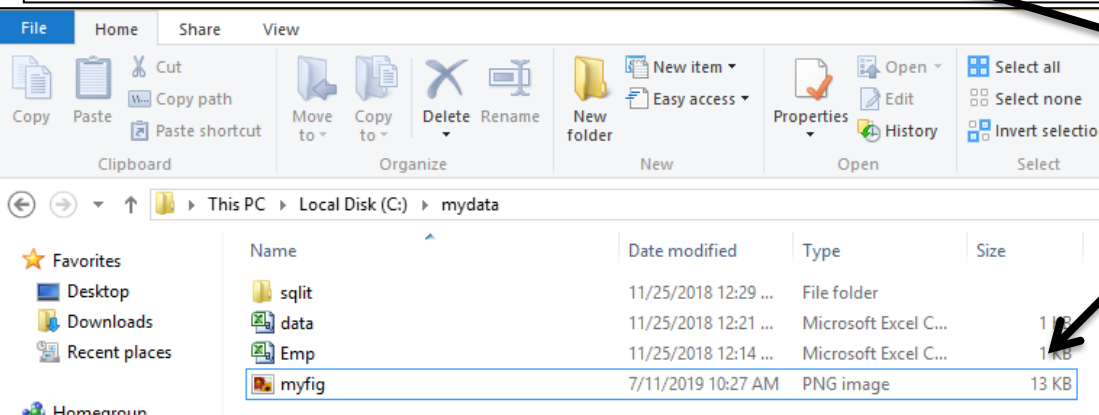
This line will be written to save the figure of Plotting.

Thank you

Please follow us on our blog

www.padhaaii.wordpress.com