Python for Machine Learning

scikit-learn it is open source project most prominent python library for machine learning scikit-learn depends on two other packages numpy and scipy. For plotting and interactive development we need matplotlib, Ipython and jupyter notebook

any one among the list

1. Anaconda
2. Enthough Canopy
3. Python(x,y)

Essential Libraries and tools

scikit-learn build on top of Numpy and scipy pandas matplotlib jupyter notebook browser based interactive programming environment

```
#NumPy stands for Numerical Python.
NumPy - fundamental package for scientific Computing in Python

'''
contains functionality for multidimentional array,high level mathematical function such as
linear algebra,
fourier transform and pseudorandom number generator

in scikit-learn numpy array is fundamental data structure
scikit-learn takes in data in the form of numpy array
core functionality of numpy is the ndarray ( n-dimensional array)

NumPy aims to provide an array object that is up to 50x faster than traditional Python
lists
```

In [1]:

```
print("Welcome to python for Machine Learning")
```

```
Welcome to python for Machine Learning
```

In [2]:

```
#numpy exmple 1
import numpy
arr = numpy.array([1, 2, 3, 4, 5])
print(arr)
```

```
[1 2 3 4 5]
```

In [3]:

```
#numpy exmple 2
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

```
[1 2 3 4 5]
```

In [4]:

```
#numpy example 4
#2-D Arrays
import numpy as np
arr = np.array([[1, 2, 3], [4, 5, 6]])
print(arr)
```

```
[[1 2 3]
 [4 5 6]]
```

In [5]:

```
#numpy Example 6
#3-D arrays
#An array that has 2-D arrays (matrices) as its elements is called 3-D array.
import numpy as np

arr = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(arr)
```

```
[[[1 2 3]
  [4 5 6]]

 [[1 2 3]
  [4 5 6]]]
```

In [6]:

```
#numpy Example 7
#Check Number of Dimensions
#NumPy Arrays provides the ndim attribute that returns an integer that tells us how many di
import numpy as np

a = np.array(42)
b = np.array([1, 2, 3, 4, 5])
c = np.array([[1, 2, 3], [4, 5, 6]])
d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(a.ndim)
print(b.ndim)
print(c.ndim)
print(d.ndim)
```

```
0
1
2
3
```

In [7]:

```
# to test whether none of the elements of a given array is zero.
import numpy as np
x = np.array([1, 2, 3, 4])
print("Original array:")
print(x)
print("Test if none of the elements of the said array is zero:")
print(np.all(x))
x = np.array([0, 1, 2, 3])
print("Original array:")
print(x)
print("Test if none of the elements of the said array is zero:")
print(np.all(x))
```

```
Original array:
[1 2 3 4]
Test if none of the elements of the said array is zero:
True
Original array:
[0 1 2 3]
Test if none of the elements of the said array is zero:
False
```

In [12]:

```
#to generate a random number between 0 and 1
import numpy as np
rand_num = np.random.normal(0,1,1)
print("Random number between 0 and 1:")
print(rand_num)
```

```
Random number between 0 and 1:
[-0.89075346]
```

In [13]:

```
##to generate 10 random number between 0 and 1
import numpy as np
rand_num = np.random.normal(0,1,10)
print("Random number between 0 and 1:")
print(rand_num)
```

```
Random number between 0 and 1:
[ 0.08764494 -0.108397   -0.49951839 -0.56855244  1.59411412  0.59785653
  0.49309307 -0.5451371   0.020466   -2.06458713]
```

```
#scipy
#SciPy stands for Scientific Python.
SciPy is a scientific computation library that uses NumPy underneath.

Sparse Data: is a data set where most of the item values are zero.

Dense Array: is the opposite of a sparse array: most of the values are not zero.
'''
Collection of functions for scientific computing in python
provides
1. Aadvanced linear algebra routines
2. Mathematicals function optimization
```

```
3. Statistical distributions

most important part of scipy is sparse matrices scipy.sparse

sparse martix are used whenever we want to store a 2d array that contain mostly zeros
```

In [14]:

```
#parse Example 1
#CSR - Compressed Sparse Row. For fast row slicing, faster matrix vector products
import numpy as np
import scipy
from scipy.sparse import csr_matrix

arr = np.array([0, 0, 0, 0, 0, 1, 1, 0, 2])

print(csr_matrix(arr))
```

```
  (0, 5)        1
  (0, 6)        1
  (0, 8)        2
```

In [15]:

```
#parse Example 2
#create a 2d numpy array with diagonal of ones and zro everywhere
from scipy import sparse
eye=np.eye(4)
print(eye)
```

```
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
```

In [16]:

```
#parse Example 3
#convert the numpy array to a scipy sparse matrix in CSR format
#only non zero entries are stored

sparse_matrix = sparse.csr_matrix(eye)
print(sparse_matrix)
```

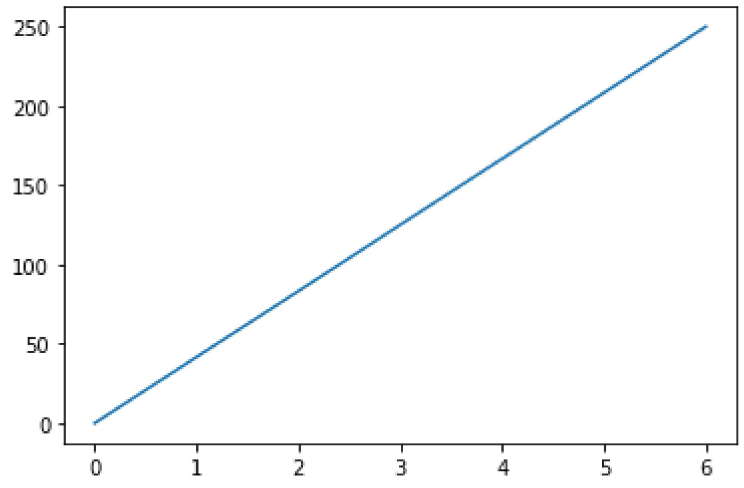
```
  (0, 0)        1.0
  (1, 1)        1.0
  (2, 2)        1.0
  (3, 3)        1.0
```

```
#matplotlib
#matplotlib is the primary scientific plotting library in python
Most of the Matplotlib utilities lies under the pyplot submodule, and are usually imported
under the plt alias:
'''
It provides functions for quality visualization like
1. line chart
2. histogram
3. scatter plot

when working in jupyter notebook we can visualize figures directly in the browser by using
%matplotlib inline commands
```

```
'''
```

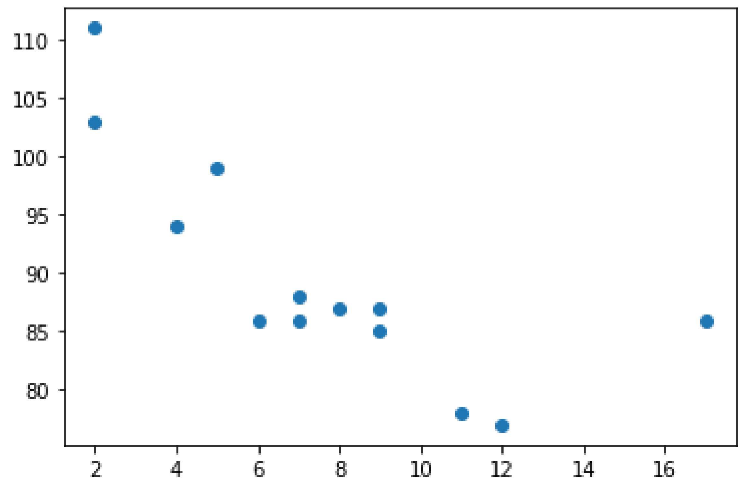
In [19]:

```
#matplot Example 1
#Draw a line in a diagram from position (0,0) to position (6,250):
import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([0, 6])
ypoints = np.array([0, 250])

plt.plot(xpoints, ypoints)
plt.show()
```



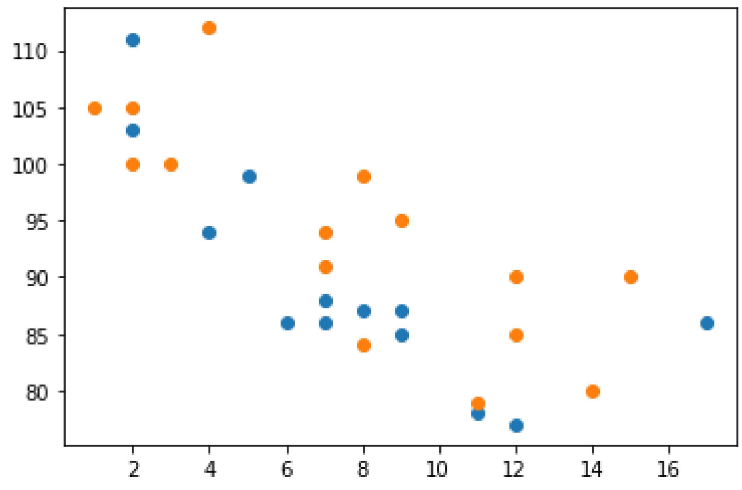
In [22]:

```
#matplot Example 2
#A simple scatter plot
#he scatter() function plots one dot for each observation.
#It needs two arrays of the same length, one for the values of the x-axis, and one for valu
import matplotlib.pyplot as plt
import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

plt.scatter(x, y)
plt.show()
```



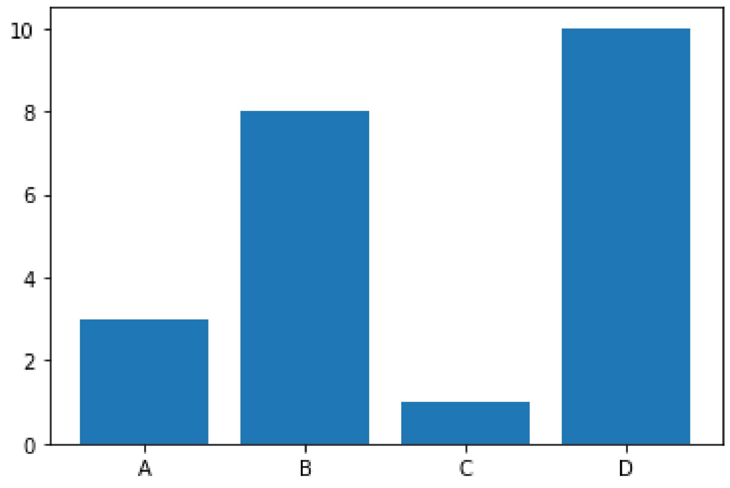
In [23]:

```
#matplot Example 3
#The observation in the example above is the result of 13 cars passing by.

#The X-axis shows how old the car is.

#The Y-axis shows the speed of the car when it passes.

#Are there any relationships between the observations?

import matplotlib.pyplot as plt
import numpy as np

#day one, the age and speed of 13 cars:
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])
plt.scatter(x, y)

#day two, the age and speed of 15 cars:
x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])
y = np.array([100,105,84,105,90,99,90,95,94,100,79,112,91,80,85])
plt.scatter(x, y)

plt.show()
```



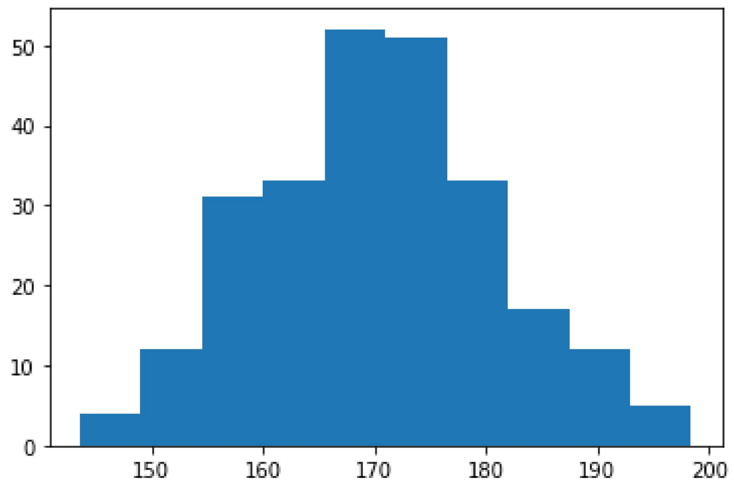
In [24]:

```
#matplot Example 4
#With Pyplot, you can use the bar() function to draw bar graphs:
import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.bar(x,y)
plt.show()
```



In [25]:

```
#The hist() function will use an array of numbers to create a histogram,
#the array is sent into the function as an argument.

#For simplicity we use NumPy to randomly generate an array with 250 values,
#where the values will concentrate around 170, and the standard deviation is 10.
import matplotlib.pyplot as plt
import numpy as np

x = np.random.normal(170, 10, 250)
#print(x)
plt.hist(x)
plt.show()
```

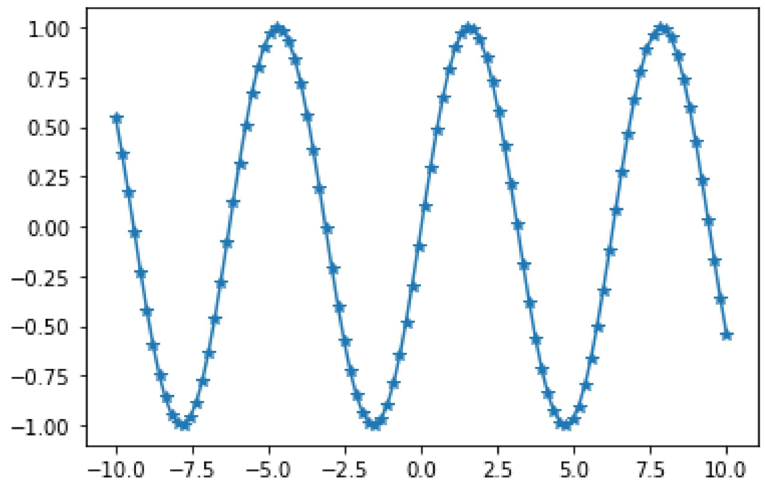


In [26]:

```
#matplot Example 6

import matplotlib.pyplot as plt
#generate a sequence of numbers from -10 to 10 with 100 steps in between
x=np.linspace(-10,10,100)
#create secondary array using sine
y=np.sin(x)
#plot function males a line chart of one array agaist another
plt.plot(x,y,marker="*")
```

Out[26]:

```
[<matplotlib.lines.Line2D at 0x25ffcd1f3a0>]
```



```
#pandas
pandas is a python library for data wrangling and nalysis build around datastructure
called dataframe
panda allowes each column to have seperate type
```

In [43]:

```
 import pandas as pd
#create a simple dataset of people
data ={'Name':["Ashwin","Vikas","John","Venkat","Balaji","Pradeep"],'Location':["Vellore","
df=pd.DataFrame(data)
print(df)
```

```
      Name   Location  Age
0   Ashwin    Vellore   39
1    Vikas    Chennai   18
2     John  Bangalore   34
3   Venkat      Delhi   15
4   Balaji       Pune   50
5  Pradeep     Mumbai   35
```

In [44]:

```
display(df[df.Age>30])
```

	Name	Location	Age
0	Ashwin	Vellore	39
2	John	Bangalore	34
4	Balaji	Pune	50
5	Pradeep	Mumbai	35

In [45]:

```
#displays top five rows
df.head()
```

Out[45]:

	Name	Location	Age
0	Ashwin	Vellore	39
1	Vikas	Chennai	18
2	John	Bangalore	34
3	Venkat	Delhi	15
4	Balaji	Pune	50

In [46]:

```
#displays bottom five rows
df.tail()
```

Out[46]:

	Name	Location	Age
1	Vikas	Chennai	18
2	John	Bangalore	34
3	Venkat	Delhi	15
4	Balaji	Pune	50
5	Pradeep	Mumbai	35

In [ ]:

```

```