# Seat No : 363102

# 1.Write a program to implement Naive Bayes Classifier Using Python

import pandas as pd

import numpy as np

df = pd.read\_csv('User\_Data.csv')

print('Dataset sample :\n',df.head())

x = df.iloc[:,3:4].values

y = df.iloc[:,-1].values

print('Features :\n',x)

print('Label :\n',y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest = train\_test\_split(x,y,test\_size=0.25,random\_state=1)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

xtrain = sc.fit\_transform(xtrain)

xtest = sc.transform(xtest)

from sklearn.naive\_bayes import GaussianNB

clf = GaussianNB()

clf.fit(xtrain,ytrain)

sc = clf.score(xtest,ytest)

print('Score :\n',sc)

ypred = clf.predict(xtest)

print('Actual values : \n',ytest)

print('Predicted values : \n',ypred)

from sklearn import metrics

Accuracy = metrics.accuracy\_score(ytest,ypred)

print('Accuracy : \n',Accuracy)

CM = metrics.confusion\_matrix(ytest,ypred)

print('Confusion Matrix : \n',CM)

CR = metrics.classification\_report(ytest,ypred)

print('Classification Report : \n',CR)

r2\_score = metrics.r2\_score(ytest,ypred)

print('r2\_score : \n',r2\_score)

print('\nMean Absolute Error:', metrics.mean\_absolute\_error(ytest, ypred))

print('\nMean Squared Error:', metrics.mean\_squared\_error(ytest, ypred))

print('\nRoot Mean Squared Error:', np.sqrt(metrics.mean\_absolute\_error(ytest, ypred)))

# Seat No : 363102

# 2.Write a program to implement Multiple Regression Using Python

import pandas as pd

import numpy as np

df = pd.read\_csv('house\_data.csv')

print('Dataset sample :\n',df.head())

x = df.iloc[:,:-1].values

y = df.iloc[:,-1].values

print('Features :\n',x)

print('Label :\n',y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest = train\_test\_split(x,y,test\_size=0.25,random\_state=1)

from sklearn.linear\_model import LinearRegression

clf = LinearRegression()

clf.fit(xtrain,ytrain)

sc = clf.score(xtest,ytest)

print('\nScore :\n',sc)

ypred = clf.predict(xtest)

print('\nActual values : \n',ytest)

print('\nPredicted values : \n',ypred)

from sklearn import metrics

print('\nMean Absolute Error:', metrics.mean\_absolute\_error(ytest, ypred))

print('\nMean Squared Error:', metrics.mean\_squared\_error(ytest, ypred))

print('\nRoot Mean Squared Error:', np.sqrt(metrics.mean\_absolute\_error(ytest, ypred)))