import numpy as np

import pandas as pd

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

from sklearn import preprocessing

from sklearn import utils

df = pd.read\_csv("D:/python data set/insurance.csv")

print(df)

x=df.iloc[:,0]

y=df.iloc[:,1]

X\_train,X\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.25,random\_state=0)

print(X\_train)

print(y\_train)

lab\_enc=preprocessing.LabelEncoder()

encoded=lab\_enc.fit\_transform(y\_train)

print(encoded)

encoded=np.array(encoded)

#print(encoded)

encoded= encoded.reshape(1,-1)

def models(X\_train,Y\_train):

  #Using Logistic Regression

  from sklearn.linear\_model import LogisticRegression

  log = LogisticRegression(random\_state = 0)

  log.fit(X\_train, Y\_train)

  #Using KNeighborsClassifier

  from sklearn.neighbors import KNeighborsClassifier

  knn = KNeighborsClassifier(n\_neighbors = 5, metric = 'minkowski', p = 2)

  knn.fit(X\_train, Y\_train)

  #Using SVC linear

  from sklearn.svm import SVC

  svc\_lin = SVC(kernel = 'linear', random\_state = 0)

  svc\_lin.fit(X\_train, Y\_train)

  #Using SVC rbf

  from sklearn.svm import SVC

  svc\_rbf = SVC(kernel = 'rbf', random\_state = 0)

  svc\_rbf.fit(X\_train, Y\_train)

  #Using GaussianNB

  from sklearn.naive\_bayes import GaussianNB

  gauss = GaussianNB()

  gauss.fit(X\_train, Y\_train)

  #Using DecisionTreeClassifier

  from sklearn.tree import DecisionTreeClassifier

  tree = DecisionTreeClassifier(criterion = 'entropy', random\_state = 0)

  tree.fit(X\_train, Y\_train)

  #Using RandomForestClassifier method of ensemble class to use Random Forest Classification algorithm

  from sklearn.ensemble import RandomForestClassifier

  forest = RandomForestClassifier(n\_estimators = 10, criterion = 'entropy', random\_state = 0)

  forest.fit(X\_train, Y\_train)

  #print model accuracy on the training data.

  print('[0]Logistic Regression Training Accuracy:', log.score(X\_train, Y\_train))

  print('[1]K Nearest Neighbor Training Accuracy:', knn.score(X\_train, Y\_train))

  print('[2]Support Vector Machine (Linear Classifier) Training Accuracy:', svc\_lin.score(X\_train, Y\_train))

  print('[3]Support Vector Machine (RBF Classifier) Training Accuracy:', svc\_rbf.score(X\_train, Y\_train))

  print('[4]Gaussian Naive Bayes Training Accuracy:', gauss.score(X\_train, Y\_train))

  print('[5]Decision Tree Classifier Training Accuracy:', tree.score(X\_train, Y\_train))

  print('[6]Random Forest Classifier Training Accuracy:', forest.score(X\_train, Y\_train))

models(X\_train, encoded)