import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

dataset=pd.read\_csv('diabetes.csv')

X = dataset.iloc[:, :-1].values

y = dataset.iloc[:,[8]].values.ravel()

from sklearn.preprocessing import Imputer

imputer = Imputer(missing\_values = 0, strategy = 'median', axis = 0)

imputer = imputer.fit(X[:, 0:6])

X[:, 0:6] = imputer.transform(X[:, 0:6])

# Test Train set split

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 0)

# Feature Scaling

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.transform(X\_test)

'''NAIVE BAYES'''

from sklearn.naive\_bayes import GaussianNB

classifier1 = GaussianNB()

classifier1.fit(X\_train, y\_train)

# Predicting the Test set results

y\_pred1 = classifier1.predict(X\_test)

from sklearn.metrics import confusion\_matrix

cm1=confusion\_matrix(y\_test,y\_pred1)

accuracy\_naivebayes = (cm1[0][0]+cm1[1][1])/154

print(classifier1.predict([[6,148,72,35,125,33.6,0.625,50]])[0])

'''RANDOMFOREST '''

from sklearn.ensemble import RandomForestClassifier

classifier3 = RandomForestClassifier(n\_estimators =100, criterion = 'entropy', random\_state = 0)

classifier3.fit(X\_train, y\_train)

# Predicting the Test set results

y\_pred3 = classifier3.predict(X\_test)

from sklearn.metrics import confusion\_matrix

cm3=confusion\_matrix(y\_test,y\_pred3)

accuracy\_forest = (cm3[0][0]+cm3[1][1])/154

''' K NEIGHBORS '''

from sklearn.neighbors import KNeighborsClassifier

classifier4 = KNeighborsClassifier(n\_neighbors = 25, metric = 'minkowski', p = 2)

classifier4.fit(X\_train, y\_train)

# Predicting the Test set results

y\_pred4 = classifier4.predict(X\_test)

# Making the Confusion Matrix

from sklearn.metrics import confusion\_matrix

cm4 = confusion\_matrix(y\_test, y\_pred4)

accuracy\_kneighbors = (cm4[0][0]+cm4[1][1])/154

print(classifier4.predict([[2,140,60,25,200,30,0.230,45]]))

'''LOGISTIC REGRESSION'''

from sklearn.linear\_model import LogisticRegression

classifier5= LogisticRegression(C=0.3)

classifier5.fit(X\_train, y\_train)

# Predicting the Test set results

y\_pred5 = classifier5.predict(X\_test)

# Making the Confusion Matrix

from sklearn.metrics import confusion\_matrix

cm5 = confusion\_matrix(y\_test, y\_pred5)

accuracy\_Lregression= (cm5[0][0]+cm5[1][1])/154

''' accuracies of Algorithms '''

print("accuracy of Logistic Regression \t" + str(accuracy\_Lregression))

print("accuracy of Naive Bayes \t\t" + str(accuracy\_naivebayes))

print("accuracy of KNN \t\t\t" + str(accuracy\_kneighbors))

print("accuracy of Random Forest \t\t" + str(accuracy\_forest))