import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

from sklearn import datasets,linear\_model

from sklearn.metrics import mean\_squared\_error,r2\_score

df = pd.read\_csv('E:\\Latha\\JG\_MCA\\MachineLearning\\KNN\\shirtsize.csv')

print(df.columns)

df

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

df['T Shirt Size'] = le.fit\_transform(df['T Shirt Size'])

df

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

Y = df.iloc[:, 2].values

X

Y

#Split traing and testing

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)

#K nearest neighbor

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n\_neighbors = 2) # n\_neighbors means k

knn.fit(X\_train, Y\_train)

prediction = knn.predict(X\_test)

#accuracies['KNN'] = acc

print("{} NN Score: {:.2f}%".format(2, knn.score(X\_test, Y\_test)\*100))

print(knn.score(X\_test,Y\_test))

#Predict for single

print(knn.predict([[158,58]]))

print(knn.predict([[170,68]]))

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

result = confusion\_matrix(Y\_test, prediction)

print("Confusion Matrix:")

print(result)

result1 = classification\_report(Y\_test, prediction)

print("Classification Report:",)

print (result1)

result2 = accuracy\_score(Y\_test,prediction)

print("Accuracy:",result2)

#Decision Tree

from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier().fit(X\_train, Y\_train)

prediction1 = clf.predict(X\_test)

print('Accuracy of Decision Tree classifier on training set: {:.2f}'

.format(clf.score(X\_train, Y\_train)))

print('Accuracy of Decision Tree classifier on test set: {:.2f}'

.format(clf.score(X\_test, Y\_test)))

result = confusion\_matrix(Y\_test, prediction1)

print("Confusion Matrix:")

print(result)

result1 = classification\_report(Y\_test, prediction1)

print("Classification Report:",)

print (result1)

result2 = accuracy\_score(Y\_test,prediction1)

print("Accuracy:",result2)

#SVM

from sklearn.svm import SVC

svm = SVC()

svm.fit(X\_train, Y\_train)

prediction2 = svm.predict(X\_test)

print('Accuracy of SVM classifier on training set: {:.2f}'

.format(svm.score(X\_train, Y\_train)))

print('Accuracy of SVM classifier on test set: {:.2f}'

.format(svm.score(X\_test, Y\_test)))

result = confusion\_matrix(Y\_test, prediction2)

print("Confusion Matrix:")

print(result)

result1 = classification\_report(Y\_test, prediction2)

print("Classification Report:",)

print (result1)

result2 = accuracy\_score(Y\_test,prediction2)

print("Accuracy:",result2)

#Naive Bayes

from sklearn.naive\_bayes import GaussianNB

gnb = GaussianNB()

gnb.fit(X\_train, Y\_train)

prediction3 = gnb.predict(X\_test)

print('Accuracy of GNB classifier on training set: {:.2f}'

.format(gnb.score(X\_train, Y\_train)))

print('Accuracy of GNB classifier on test set: {:.2f}'

.format(gnb.score(X\_test, Y\_test)))

result = confusion\_matrix(Y\_test, prediction3)

print("Confusion Matrix:")

print(result)

result1 = classification\_report(Y\_test, prediction3)

print("Classification Report:",)

print (result1)

result2 = accuracy\_score(Y\_test,prediction3)

print("Accuracy:",result2)