import numpy as np

import pandas as pd

import sklearn

from sklearn.svm import SVC

from sklearn.neural\_network import MLPClassifier

from sklearn.ensemble import RandomForestClassifier

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

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import accuracy\_score

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA

HD\_Data = pd.read\_csv("/content/heart disease classification dataset.csv")

print(HD\_Data.head())

HD\_Data = HD\_Data.drop(['Unnamed: 0'],axis =1)

HD\_Data.isnull().sum()

HD\_Data = HD\_Data.dropna(axis = 0, subset = ['trestbps', 'chol', 'thalach'])

HD\_Data.isnull().sum()

HD\_Data.info()

HD\_Data['sex'] = HD\_Data['sex'].map({'female':2, 'male':1})

HD\_Data['target'] = HD\_Data['target'].map({'yes':1, 'no':0})

print(HD\_Data[['sex', 'target']])

X = HD\_Data[['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang','oldpeak','slope','ca','thal']]

Y = HD\_Data[['target']]

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y,stratify=Y,test\_size=0.20, random\_state=0)

scaler = MinMaxScaler()

scaler.fit(X\_train)

X\_train\_scaled = scaler.transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

svc = SVC(kernel="linear")

svc.fit(X\_train\_scaled, Y\_train)

svc\_score = svc.score(X\_test\_scaled, Y\_test)

print("Support Vector Machine accuracy pre-PCA: {:.2f}".format(svc\_score))

rf = RandomForestClassifier(n\_estimators=50)

rf.fit(X\_train\_scaled, Y\_train)

rf\_score = rf.score(X\_test\_scaled, Y\_test)

print("Random Forest accuracy pre-PCA: {:.2f}".format(rf\_score))

nn = MLPClassifier(hidden\_layer\_sizes=(7), activation="relu", max\_iter=10000)

nn.fit(X\_train\_scaled, Y\_train)

nn\_score = nn.score(X\_test\_scaled, Y\_test)

print("Neural Network accuracy pre-PCA: {:.2f}".format(nn\_score))

pca = PCA(n\_components=7)

principal\_components= pca.fit\_transform(X)

principal\_df = pd.DataFrame(data=principal\_components, columns=["principle component 1", "principle component 2", "principle component 3", "principle component 4", "principle component 5", "principle component 6", "principle component 7"])

print(principal\_df.head())

main\_df=pd.concat([principal\_df, HD\_Data[["target"]]], axis=1)

print(main\_df.head())

main\_df = main\_df.dropna(axis = 0)

new\_X = main\_df.drop("target", axis=1)

new\_Y = main\_df[["target"]]

new\_X\_train, new\_X\_test, new\_Y\_train, new\_Y\_test = train\_test\_split(new\_X, new\_Y,stratify=new\_Y, test\_size=0.2, random\_state=0)

new\_scaler = MinMaxScaler()

new\_scaler.fit(new\_X\_train)

new\_X\_train\_scaled = new\_scaler.transform(new\_X\_train)

new\_X\_test\_scaled = new\_scaler.transform(new\_X\_test)

new\_svc = SVC(kernel="linear")

new\_svc.fit(new\_X\_train\_scaled, new\_Y\_train)

new\_svc\_score = new\_svc.score(new\_X\_test\_scaled, new\_Y\_test)

print("Support Vector Machine accuracy post-PCA: {:.2f}".format(new\_svc\_score))

new\_rf = RandomForestClassifier(n\_estimators=50)

new\_rf.fit(new\_X\_train\_scaled, new\_Y\_train)

new\_rf\_score = new\_rf.score(new\_X\_test\_scaled, new\_Y\_test)

print("Random Forest accuracy post-PCA: {:.2f}".format(new\_rf\_score))

new\_nn = MLPClassifier(hidden\_layer\_sizes=(7), activation="relu", max\_iter=10000)

new\_nn.fit(new\_X\_train\_scaled, new\_Y\_train)

new\_nn\_score = new\_nn.score(new\_X\_test\_scaled, new\_Y\_test)

print("Neural Network accuracy post-PCA: {:.2f}".format(new\_nn\_score))

plt.bar(['Before PCA', 'After PCA'], [svc\_score, new\_svc\_score])

plt.title('Support Vector Machine')

plt.show()

plt.bar(['Before PCA', 'After PCA'],[rf\_score, new\_rf\_score])

plt.title('Random Forest')

plt.show()

plt.bar(['Before PCA', 'After PCA'],[nn\_score, new\_nn\_score])

plt.title('Neural Network')

plt.show()