*CSE422 Lab08 Sklearn Library*

*Assignment 06*

Name: Subhi Bhuiyan

ID: 17201116

Section: 03

#importing libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

%matplotlib inline

heart\_dataset = pd.read\_csv('/content/heart failur classification dataset.csv')

heart\_dataset.head(3)

heart\_dataset.shape

heart\_dataset.isnull().sum()

heart\_dataset = heart\_dataset.drop(['time'], axis = 1)

heart\_dataset = heart\_dataset.drop(['serum\_sodium'], axis = 1)

heart\_dataset.shape

heart\_dataset.isnull().sum()

heart\_dataset['sex']=heart\_dataset['sex'].map({'Male':0,'Female':1})

heart\_dataset['smoking']=heart\_dataset['smoking'].map({'No':0,'Yes':1})

pip install -U scikit-learn

from sklearn.metrics import accuracy\_score

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

x = heart\_dataset.iloc[:, :-1]

y = heart\_dataset.iloc[:, -1]

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

print("Training set: x->{} , y->{} \n Testing set: x->{} , y->{}".format(x\_train.shape, y\_train.shape, x\_test.shape, y\_test.shape))

Support Vector Machine (SVM)

from sklearn.svm import SVC

svc = SVC(kernel="linear")

svc.fit(x\_train, y\_train)

print("Training accuracy of the model is {:.2f}".format(svc.score(x\_train, y\_train)))

print("Testing accuracy of the model is {:.2f}".format(svc.score(x\_test, y\_test)))

predictions = svc.predict(x\_test)

print(predictions)

scoreSVC = accuracy\_score(y\_test, predictions)

from sklearn.metrics import confusion\_matrix

mat=confusion\_matrix(predictions, y\_test)

print(mat)

Neural Network (MLPClassifier)

from sklearn.neural\_network import MLPClassifier

nnc=MLPClassifier(hidden\_layer\_sizes=(8), activation="relu", max\_iter=100000)

nnc.fit(x\_train, y\_train)

print("The Training accuracy of the model is {:.2f}".format(nnc.score(x\_train, y\_train)))

print("The Testing accuracy of the model is {:.2f}".format(nnc.score(x\_test, y\_test)))

predictions = nnc.predict(x\_test)

print(predictions)

scoreMLP = accuracy\_score(y\_test, predictions)

from sklearn.metrics import confusion\_matrix

mat=confusion\_matrix(predictions, y\_test)

print(mat)

Random Forest

from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(n\_estimators=50)

rfc.fit(x\_train, y\_train)

print("The Training accuracy of the model is {:.2f}".format(rfc.score(x\_train, y\_train)))

print("The Testing accuracy of the model is {:.2f}".format(rfc.score(x\_test, y\_test)))

predictions = rfc.predict(x\_test)

scoreRFC = accuracy\_score(y\_test, predictions)

from sklearn.metrics import confusion\_matrix

mat=confusion\_matrix(predictions, y\_test)

print(mat)

PCA

from sklearn.decomposition import PCA

pca = PCA(n\_components=6)

principal\_components= pca.fit\_transform(heart\_dataset)

print(principal\_components)

sum(pca.explained\_variance\_ratio\_)

principal\_df= pd.DataFrame(data=principal\_components, columns = ["PC1", "PC2", "PC3", "PC4", "PC5", "PC6"])

principal\_df.head()

main\_df=pd.concat([principal\_df,heart\_dataset["DEATH\_EVENT"]], axis=1)

main\_df.head()

x2 = main\_df.drop("DEATH\_EVENT" , axis=1)

y2 = main\_df["DEATH\_EVENT"]

x2\_train, x2\_test, y2\_train, y2\_test = train\_test\_split(x2 , y2 , test\_size=0.2, random\_state=0)

#SVM

svc.fit(x2\_train, y2\_train)

print("Training accuracy of the model is {:.2f}".format(svc.score(x2\_train, y2\_train)))

print("Testing accuracy of the model is {:.2f}".format(svc.score(x2\_test, y2\_test)))

predictions = svc.predict(x2\_test)

scoreSVC2 = accuracy\_score(y2\_test, predictions)

#RandomForest

rfc.fit(x2\_train, y2\_train)

print("The Training accuracy of the model is {:.2f}".format(rfc.score(x2\_train, y2\_train)))

print("The Testing accuracy of the model is {:.2f}".format(rfc.score(x2\_test, y2\_test)))

predictions = rfc.predict(x2\_test)

scoreRFC2 = accuracy\_score(y2\_test, predictions)

#MLPClassifier

nnc.fit(x2\_train, y2\_train)

print("The Training accuracy of the model is {:.2f}".format(nnc.score(x2\_train, y2\_train)))

print("The Testing accuracy of the model is {:.2f}".format(nnc.score(x2\_test, y2\_test)))

predictions = nnc.predict(x2\_test)

scoreMLP2 = accuracy\_score(y\_test, predictions)

Comparing Accuracy

plt.bar(['Pre-PCA','Post-PCA'],[scoreSVC, scoreSVC2],align='center',alpha=0.5)

plt.title('Support Vector Machine')

plt.ylabel("Accuracy")

plt.show()

plt.bar(['Pre-PCA','Post-PCA'],[scoreMLP, scoreMLP2],align='center',alpha=0.5)

plt.title('Neural Network MLPClassifier')

plt.ylabel("Accuracy")

plt.show()

plt.bar(['Pre-PCA','Post-PCA'],[scoreRFC, scoreRFC2],align='center',alpha=0.5)

plt.title('Random Forest Classifier')

plt.ylabel("Accuracy")

plt.show()