import pandas as pd

import numpy as np

import sklearn

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

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

from sklearn import tree

import matplotlib.pyplot as plt

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

print(HD\_Data.shape)#Shape before droping anyting from database

print(HD\_Data.head(3))

HD\_Data.drop(HD\_Data.columns[HD\_Data.columns.str.contains('Unnamed',case = False)],axis = 1, inplace = True)

print(HD\_Data.shape)#Shape after droping column

print(HD\_Data.head(3))

print(HD\_Data.isnull().sum()) #To view among all columns which of the columns contains null value

print("Number of rows with null values in trestbps :", HD\_Data['trestbps'].isnull().sum())

print("Number of rows with null values in chol :", HD\_Data['chol'].isnull().sum())

print("Number of rows with null values in thalach :", HD\_Data['thalach'].isnull().sum())

print("Shape of dataframe before dropping:", HD\_Data.shape)

HD\_Data = HD\_Data.dropna(axis = 0, subset = ['trestbps', 'chol', 'thalach'])#Droping null value row from column

print("Shape after dropping:", HD\_Data.shape)

HD\_Data['sex'] = HD\_Data['sex'].map({'male':1, 'female':2})#Maping categorical feature (Male=1, Female=2)

HD\_Data['target'] = HD\_Data['target'].map({'yes':1, 'no':0})#Maping categorical feature (Yes=1, No=0)

print(HD\_Data[['sex', 'target']])#Printing after encoding categorical feature

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, random\_state=0)#Spliting data into train and test set

print(X\_train.shape)

print(X\_test.shape)

print(Y\_train.shape)

print(Y\_test.shape)

knn=KNeighborsClassifier()

knn.fit(X\_train, Y\_train)

m=(knn.score(X\_test, Y\_test))

print("Test set accuracy: {:.2f}".format(knn.score(X\_test, Y\_test)))

x = HD\_Data.iloc[:,0:13]

y = HD\_Data.iloc[:,13]

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.2,random\_state=1)

clf = DecisionTreeClassifier(criterion='entropy',random\_state=1)

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

y\_pred = clf.predict(x\_test)

print("Decision Tree accuracy: {:.2f}".format(accuracy\_score(y\_pred,y\_test)))#accuracy by decision tree

n=accuracy\_score(y\_pred,y\_test)

fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (4,4), dpi=500)

tree.plot\_tree(clf,feature\_names = x.columns, class\_names=['yes','no'],filled = True);

#showing accuracy using bar chart

p=["Normal Test accuracy","Decision Tree accuracy"]

q=[m,n]

plt.bar(p,q)

plt.xlabel("processs")

plt.ylabel("accuracy")

plt.show()