#Importing necessary libraries

import pandas as pd

import numpy as np

import sklearn

from sklearn.impute import SimpleImputer

from sklearn.preprocessing import LabelEncoder , MinMaxScaler

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

from sklearn.neighbors import KNeighborsClassifier

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import classification\_report

from sklearn.metrics import accuracy\_score

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

from sklearn.tree import DecisionTreeClassifier

#Load the dataset (breast cancer data)

breast\_cancer = pd.read\_csv('/content/breast cancer classification dataset.csv')

#Droping highest null value's and unique values column (here it is-->id,Unnamed: 32)

breast\_cancer = breast\_cancer.drop(['id','Unnamed: 32'],axis=1)

#Imputing Missing values

impute = SimpleImputer(missing\_values=np.nan,strategy='mean')

impute.fit(breast\_cancer[['radius\_mean']])

breast\_cancer['radius\_mean'] = impute.transform(breast\_cancer[['radius\_mean']])

impute.fit(breast\_cancer[['fractal\_dimension\_worst']])

breast\_cancer['fractal\_dimension\_worst'] = impute.transform(breast\_cancer[['fractal\_dimension\_worst']])

#Encode categorical feature

labelling = LabelEncoder()

breast\_cancer['diagnosis'] = labelling.fit\_transform(breast\_cancer['diagnosis'])

#Feature and Label selection

features = breast\_cancer.iloc[:,1:31]

label = breast\_cancer.iloc[:,0]

#Split train-test (8:2)

y = pd.DataFrame(label)

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

#For logistic regression

logisticReg\_model = LogisticRegression()

logisticReg\_model.fit(x\_train, y\_train)

y\_prediction\_logisticReg = logisticReg\_model.predict(x\_test)

accuracy\_logisticReg = accuracy\_score(y\_prediction\_logisticReg,y\_test)

print("Accuracy for Logistic-Regression :",accuracy\_logisticReg)

#For decision tree

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

decisionTree\_model.fit(x\_train,y\_train)

y\_prediction\_decision = decisionTree\_model.predict(x\_test)

accuracy\_decisionTree = accuracy\_score(y\_prediction\_decision,y\_test)

print("Accuracy for Decision-Tree : ",accuracy\_decisionTree)

#Comparing the accuracy

if accuracy\_logisticReg > accuracy\_decisionTree :

   print("Logistic Regression is more accurate than Decision Tree in this case .")

elif accuracy\_logisticReg < accuracy\_decisionTree :

  print("Decision Tree is more accurate than Logistic Regression in this case")

else:

  print("Both Logistic-Regression and Decision-Tree has equal accuracy")

#Bar chart

plt.bar(['Logistic-Regression','Decision-Tree'],[accuracy\_logisticReg,accuracy\_decisionTree],color = 'green')

plt.title('Logistic-Regression Vs Decision-Tree (Accuracy)')

plt.xlabel('Model Name')

plt.ylabel('Accuracy')

plt.show()