import numpy as np

import pandas as pd

import sklearn as sk

import seaborn as sns

b\_c = pd.read\_csv('/content/sample\_data/breast cancer classification dataset.csv')

print(b\_c.columns)

print(b\_c.shape)

b\_c = b\_c.drop(axis = 1, columns = "Unnamed: 32")

b\_c.head()

print(b\_c.isnull().sum())

label = b\_c.diagnosis

f = b\_c.drop(columns = 'diagnosis')

from sklearn.impute import SimpleImputer

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

impute.fit(b\_c[ ['smoothness\_se'] ])

b\_c['smoothness\_se'] = impute.transform(b\_c[ ['smoothness\_se'] ])

impute.fit(b\_c[ ['concavity\_se'] ])

b\_c['concavity\_se'] = impute.transform(b\_c[ ['concavity\_se'] ])

impute.fit(b\_c[ ['symmetry\_se'] ])

b\_c['symmetry\_se'] = impute.transform(b\_c[ ['symmetry\_se'] ])

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

scaler.fit(f)

f\_s = scaler.transform(f)

f\_s\_df = pd.DataFrame(f\_s, columns = list(b\_c.columns)[:-1])

f\_s\_df.replace([np.inf, -np.inf], np.nan, inplace=True)

f\_s\_df.fillna(999, inplace=True)

f\_s\_df.head()

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(f\_s, label.values, test\_size = 0.2,

random\_state = 1)

import matplotlib.pyplot as plt

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

model = LogisticRegression()

model.fit(X\_train, y\_train)

logistic\_prediction = model.predict(X\_test)

print(logistic\_prediction)

print( accuracy\_score(y\_test, logistic\_prediction))

accure = accuracy\_score(y\_test, logistic\_prediction)

from sklearn.tree import DecisionTreeClassifier

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

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

y\_pred = clf.predict(X\_test)

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

print(score)

fig,ax = plt.subplots()

ax.bar(["Decision tree", "Logistic reg"], [score, accure])

ax.set\_title('Method\_accuracy')

ax.set\_xlabel('method\_name')

ax.set\_ylabel('accuracy')

plt.show()