import pandas as pd

import numpy as np

dataset=pd.read\_csv('/content/breast\_cancer\_classification\_dataset.csv')

dataset.head()

dataset.shape

dataset=dataset.drop(['id','Unnamed: 32'],1)

dataset.isnull().sum()

from sklearn.impute import SimpleImputer

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

impute.fit(dataset[['radius\_mean']])

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

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

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

dataset.isnull().sum()

print(dataset['diagnosis'].unique())

from sklearn.preprocessing import LabelEncoder

enc = LabelEncoder()

dataset['diagnosis'] = enc.fit\_transform(dataset['diagnosis'])

dataset.info()

features=['radius\_mean','texture\_mean','perimeter\_mean','area\_mean','smoothness\_mean','compactness\_mean','concavity\_mean','concave points\_mean','symmetry\_mean','fractal\_dimension\_mean','radius\_se','texture\_se','perimeter\_se','area\_se','smoothness\_se','compactness\_se','concavity\_se','concave points\_se','symmetry\_se','fractal\_dimension\_se','radius\_worst','texture\_worst','perimeter\_worst','area\_worst','smoothness\_worst','compactness\_worst','concavity\_worst','concave points\_worst','symmetry\_worst','fractal\_dimension\_worst']

label=['diagnosis']

X=dataset[features]

y=dataset[label]

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

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

print(x\_train.shape)

print(x\_test.shape)

from sklearn.linear\_model import LogisticRegression

#Train the model

model = LogisticRegression()

model.fit(x\_train, y\_train) #Training the model

predictions = model.predict(x\_test)

print("Predictions : ", predictions)# printing predictions

from sklearn.metrics import accuracy\_score

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

print("Accuracy of linear regression:",LinearRegreAccuracy)

from sklearn.tree import DecisionTreeClassifier

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

X = dataset.iloc[:,1:]

y = dataset.iloc[:, 0]

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state=42)

clf = DecisionTreeClassifier(criterion='entropy',random\_state=2)

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

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

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

print("Decision Tree Accuracy: ",decisionTreeAccuracy)

import matplotlib.pyplot as plt

plt.xlabel("Algorithm")

plt.ylabel("Score")

x\_axis=["Logistic Regression","Decision Tree"]

y\_axis=[LinearRegreAccuracy,decisionTreeAccuracy]

plt.bar(x\_axis,y\_axis)

plt.show()