## **Lab Assignment-5**

**Afsarul Islam Meraj**

**18301239**

**Sec-3**

# Importing the libraries

import numpy as np

import pandas as pd

# Loading the dataset as dataframe using pandas

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

heart\_df

# Dropping the unnecessary column

heart\_df.drop("Unnamed: 0", axis=1, inplace=True)

heart\_df.shape

# Handling the missing values

heart\_df.isnull().sum()

from sklearn.impute import SimpleImputer

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

simple\_imputer.fit(heart\_df[['serum\_sodium','time']])

heart\_df[['serum\_sodium','time']]= simple\_imputer.transform(heart\_df[['serum\_sodium','time']])

heart\_df

# Encoding categorical features

heart\_df.info()

heart\_df['sex'].unique()

heart\_df['smoking'].unique()

from sklearn.preprocessing import LabelEncoder

label\_encoder = LabelEncoder()

heart\_df['smoking'] = label\_encoder.fit\_transform(heart\_df['smoking'])

heart\_df['sex'] = label\_encoder.fit\_transform(heart\_df['sex'])

heart\_df.head()

# Scaling all the values between 0-1 using MinMax Scaler

from sklearn.preprocessing import MinMaxScaler

minmax\_scaler = MinMaxScaler()

minmax\_scaler.fit(heart\_df)

heart\_df=pd.DataFrame(minmax\_scaler.transform(heart\_df))

# Splitting the dataset into features and labels

features= heart\_df.iloc[:,:-1]

features.shape

labels= heart\_df.iloc[:,12]

labels.shape

# Training set and testing set splitting (8:2)

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

features\_train, features\_test, labels\_train, labels\_test = train\_test\_split(features, labels, test\_size=0.2, random\_state=1)

# Logistic Regression

from sklearn.linear\_model import LogisticRegression

lr= LogisticRegression()

lr.fit(features\_train, labels\_train)

labels\_predict\_lr= lr.predict(features\_test)

from sklearn.metrics import accuracy\_score

accuracy\_lr= accuracy\_score(labels\_test, labels\_predict\_lr)

accuracy\_lr

# Decision Tree

from sklearn.tree import DecisionTreeClassifier

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

dtc.fit(features\_train, labels\_train)

labels\_predict\_dtc=dtc.predict(features\_test)

from sklearn.metrics import accuracy\_score

accuracy\_dtc= accuracy\_score(labels\_test, labels\_predict\_dtc)

accuracy\_dtc

# Comparing the accuracy using a bar chart

import matplotlib.pyplot as plt

plt.figure(figsize=(10,10))

classifiers = ['Logistic Regression', 'Decision Tree']

accuracy= [accuracy\_lr, accuracy\_dtc]

plt.bar(classifiers, accuracy)

plt.show()