DIABETES:

import numpy as np

import pandas as pd

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

from sklearn import svm

from sklearn.metrics import accuracy\_score

# loading the diabetes dataset to a pandas DataFrame

diabetes\_dataset = pd.read\_csv('E:\Multiple disease prediction system\dataset-20230622T140539Z-001\dataset\diabetes.csv')

# printing the first 5 rows of the dataset

diabetes\_dataset.head()

# number of rows and Columns in this dataset

diabetes\_dataset.shape

# getting the statistical measures of the data

diabetes\_dataset.describe()

diabetes\_dataset['Outcome'].value\_counts()

diabetes\_dataset.groupby('Outcome').mean()

# separating the data and labels

X = diabetes\_dataset.drop(columns = 'Outcome', axis=1)

Y = diabetes\_dataset['Outcome']

print(X)

print(Y)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y, test\_size = 0.2, stratify=Y, random\_state=2)

print(X.shape, X\_train.shape, X\_test.shape)

classifier = svm.SVC(kernel='linear')

#training the support vector Machine Classifier

classifier.fit(X\_train, Y\_train)

# accuracy score on the training data

X\_train\_prediction = classifier.predict(X\_train)

training\_data\_accuracy = accuracy\_score(X\_train\_prediction, Y\_train)

print('Accuracy score of the training data : ', training\_data\_accuracy)

# accuracy score on the test data

X\_test\_prediction = classifier.predict(X\_test)

test\_data\_accuracy = accuracy\_score(X\_test\_prediction, Y\_test)

print('Accuracy score of the test data : ', test\_data\_accuracy)

input\_data = (5,166,72,19,175,25.8,0.587,51)

# changing the input\_data to numpy array

input\_data\_as\_numpy\_array = np.asarray(input\_data)

# reshape the array as we are predicting for one instance

input\_data\_reshaped = input\_data\_as\_numpy\_array.reshape(1,-1)

prediction = classifier.predict(input\_data\_reshaped)

print(prediction)

if (prediction[0] == 0):

print('The person is not diabetic')

else:

print('The person is diabetic')

import pickle

filename = 'diabetes\_model.sav'

pickle.dump(classifier, open(filename, 'wb'))

# loading the saved model

loaded\_model = pickle.load(open('diabetes\_model.sav', 'rb'))

input\_data = (5,166,72,19,175,25.8,0.587,51)

# changing the input\_data to numpy array

input\_data\_as\_numpy\_array = np.asarray(input\_data)

# reshape the array as we are predicting for one instance

input\_data\_reshaped = input\_data\_as\_numpy\_array.reshape(1,-1)

prediction = loaded\_model.predict(input\_data\_reshaped)

print(prediction)

if (prediction[0] == 0):

print('The person is not diabetic')

else:

print('The person is diabetic')

for column in X.columns:

print(column)

HEART:

Importing the Dependencies

import numpy as np

import pandas as pd

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

# loading the csv data to a Pandas DataFrame

heart\_data = pd.read\_csv('E:\Multiple disease prediction system\dataset-20230622T140539Z-001\dataset\heart.csv')

# print first 5 rows of the dataset

heart\_data.head()

# print last 5 rows of the dataset

heart\_data.tail()

# number of rows and columns in the dataset

heart\_data.shape

# getting some info about the data

heart\_data.info()

# checking for missing values

heart\_data.isnull().sum()

# statistical measures about the data

heart\_data.describe()

# checking the distribution of Target Variable

heart\_data['target'].value\_counts()

X = heart\_data.drop(columns='target', axis=1)

Y = heart\_data['target']

print(X)

print(Y)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, stratify=Y, random\_state=2)

print(X.shape, X\_train.shape, X\_test.shape)

model = LogisticRegression()

# training the LogisticRegression model with Training data

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

# accuracy on training data

X\_train\_prediction = model.predict(X\_train)

training\_data\_accuracy = accuracy\_score(X\_train\_prediction, Y\_train)

print('Accuracy on Training data : ', training\_data\_accuracy)

# accuracy on test data

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

test\_data\_accuracy = accuracy\_score(X\_test\_prediction, Y\_test)

print('Accuracy on Test data : ', test\_data\_accuracy)

input\_data = (62,0,0,140,268,0,0,160,0,3.6,0,2,2)

# change the input data to a numpy array

input\_data\_as\_numpy\_array= np.asarray(input\_data)

# reshape the numpy array as we are predicting for only on instance

input\_data\_reshaped = input\_data\_as\_numpy\_array.reshape(1,-1)

prediction = model.predict(input\_data\_reshaped)

print(prediction)

if (prediction[0]== 0):

print('The Person does not have a Heart Disease')

else:

print('The Person has Heart Disease')

import pickle

filename = 'heart\_disease\_model.sav'

pickle.dump(model, open(filename, 'wb'))

# loading the saved model

loaded\_model = pickle.load(open('heart\_disease\_model.sav', 'rb'))

for column in X.columns:

print(column)

Parkinson:

import numpy as np

import pandas as pd

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

from sklearn import svm

from sklearn.metrics import accuracy\_score

# loading the data from csv file to a Pandas DataFrame

parkinsons\_data = pd.read\_csv('E:\Multiple disease prediction system\dataset-20230622T140539Z-001\dataset\parkinsons.csv')

# printing the first 5 rows of the dataframe

parkinsons\_data.head()

# number of rows and columns in the dataframe

parkinsons\_data.shape

# getting more information about the dataset

parkinsons\_data.info()

# checking for missing values in each column

parkinsons\_data.isnull().sum()

# getting some statistical measures about the data

parkinsons\_data.describe()

# distribution of target Variable

parkinsons\_data['status'].value\_counts()

# grouping the data bas3ed on the target variable

parkinsons\_data.groupby('status').mean()

X = parkinsons\_data.drop(columns=['name','status'], axis=1)

Y = parkinsons\_data['status']

print(X)

print(Y)

Splitting the data to training data & Test data

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=2)

print(X.shape, X\_train.shape, X\_test.shape)

model = svm.SVC(kernel='linear')

# training the SVM model with training data

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

# accuracy score on training data

X\_train\_prediction = model.predict(X\_train)

training\_data\_accuracy = accuracy\_score(Y\_train, X\_train\_prediction)

print('Accuracy score of training data : ', training\_data\_accuracy)

# accuracy score on training data

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

test\_data\_accuracy = accuracy\_score(Y\_test, X\_test\_prediction)

print('Accuracy score of test data : ', test\_data\_accuracy)

input\_data = (197.07600,206.89600,192.05500,0.00289,0.00001,0.00166,0.00168,0.00498,0.01098,0.09700,0.00563,0.00680,0.00802,0.01689,0.00339,26.77500,0.422229,0.741367,-7.348300,0.177551,1.743867,0.085569)

# changing input data to a numpy array

input\_data\_as\_numpy\_array = np.asarray(input\_data)

# reshape the numpy array

input\_data\_reshaped = input\_data\_as\_numpy\_array.reshape(1,-1)

prediction = model.predict(input\_data\_reshaped)

print(prediction)

if (prediction[0] == 0):

print("The Person does not have Parkinsons Disease")

else:

print("The Person has Parkinsons")

import pickle

filename = 'parkinsons\_model.sav'

pickle.dump(model, open(filename, 'wb'))

# loading the saved model

loaded\_model = pickle.load(open('parkinsons\_model.sav', 'rb'))

for column in X.columns:

print(column)