**TITLE OF THE PROJECT : Parkinson’s Disease Prediction using ML**

**LANGUAGE : Python**

**AUTHORS**

Dr. D. PRADEEP

ABISHEK R

ASWINKUMAR R

BHARATHIDHASAN M

AJITH A

**PLATFORM : Machine Learning(ML)**

**CODING**

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('/content/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(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 Training

Support Vector Machine Model

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

# training the SVM model with training data

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

Model Evaluation

Accuracy Score

# 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)

Building a Predictive System

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")

Saving the trained model

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)

Visual Studio Code

import pickle

import streamlit as st

from streamlit\_option\_menu import option\_menu

#loading the saved models

parkinsons\_model = pickle.load(open('D:/desktop/Parkinsons disease prediction/saved model/parkinsons\_model.sav','rb'))

#sidebar for navigate

with st.sidebar:

selected = option\_menu('Minor Project -IV',

['Parkinsons Disease Prediction'],

icons = ['person'],

default\_index= 0)

# Parkinson's Prediction Page

if (selected == "Parkinsons Disease Prediction"):

# page title

st.title("Parkinson's Disease Prediction using ML")

col1, col2, col3, col4, col5 = st.columns(5)

with col1:

fo = st.text\_input('MDVP:Fo(Hz)')

with col2:

fhi = st.text\_input('MDVP:Fhi(Hz)')

with col3:

flo = st.text\_input('MDVP:Flo(Hz)')

with col4:

Jitter\_percent = st.text\_input('MDVP:Jitter(%)')

with col5:

Jitter\_Abs = st.text\_input('MDVP:Jitter(Abs)')

with col1:

RAP = st.text\_input('MDVP:RAP')

with col2:

PPQ = st.text\_input('MDVP:PPQ')

with col3:

DDP = st.text\_input('Jitter:DDP')

with col4:

Shimmer = st.text\_input('MDVP:Shimmer')

with col5:

Shimmer\_dB = st.text\_input('MDVP:Shimmer(dB)')

with col1:

APQ3 = st.text\_input('Shimmer:APQ3')

with col2:

APQ5 = st.text\_input('Shimmer:APQ5')

with col3:

APQ = st.text\_input('MDVP:APQ')

with col4:

DDA = st.text\_input('Shimmer:DDA')

with col5:

NHR = st.text\_input('NHR')

with col1:

HNR = st.text\_input('HNR')

with col2:

RPDE = st.text\_input('RPDE')

with col3:

DFA = st.text\_input('DFA')

with col4:

spread1 = st.text\_input('spread1')

with col5:

spread2 = st.text\_input('spread2')

with col1:

D2 = st.text\_input('D2')

with col2:

PPE = st.text\_input('PPE')

# code for Prediction

parkinsons\_diagnosis = ''

# creating a button for Prediction

if st.button("Parkinson's Test Result"):

parkinsons\_prediction = parkinsons\_model.predict([[fo, fhi, flo, Jitter\_percent, Jitter\_Abs, RAP, PPQ,DDP,Shimmer,Shimmer\_dB,APQ3,APQ5,APQ,DDA,NHR,HNR,RPDE,DFA,spread1,spread2,D2,PPE]])

if (parkinsons\_prediction[0] == 1):

parkinsons\_diagnosis = "The person has Parkinson's disease"

else:

parkinsons\_diagnosis = "The person does not have Parkinson's disease"

st.success(parkinsons\_diagnosis)

Streamlit Run Command

streamlit run "D:\desktop\Parkinsons disease prediction\Parkinsons disease pred.py"