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"