CREATE A DIABETES PREDICTION IN PYTHON

Phase 4: Development Part 2

In this Phase, I am building my diabetes prediction by performing some activities like model training, evaluation etc. with the help of machine learning techniques.

Program Code for Model Training

```
      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 import pickle
      diabetes_dataset = pd.read_csv('diabetes.csv')
      diabetes_dataset.head()
      diabetes_dataset.shape
      diabetes_dataset.describe()
      diabetes_dataset['Outcome'].value_counts()
      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')
      classifier.fit(X_train, Y_train)
      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)
```

```
       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)

       input_data_as_numpy_array = np.asarray(input_data)
       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')

       filename = 'trained_model.sav'

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

       loaded_model = pickle.load(open('trained_model.sav', 'rb'))
```

Explanation of Code in Steps

Step 1: Data collection

The very first step is to choose the dataset for our model. We can get a lot of different datasets from Kaggle. You just need to sign in to Kaggle and search for any dataset you need for the project.

Step 2: Exploring the Data

Now we have to set the development environment to build our project. For this project, we are going to build this Diabetes prediction using Machine Learning in Google Colab. You can also use Jupyter Notebook.

Step 3: Splitting the data

The next step in the building of the Machine learning model is splitting the data into training and testing sets. The training and testing data should be split in a ratio of 3:1 for better prediction results.

Step 4: Training the model

The next step is to build and train our model. We are going to use a Support vector classifier algorithm to build our model.

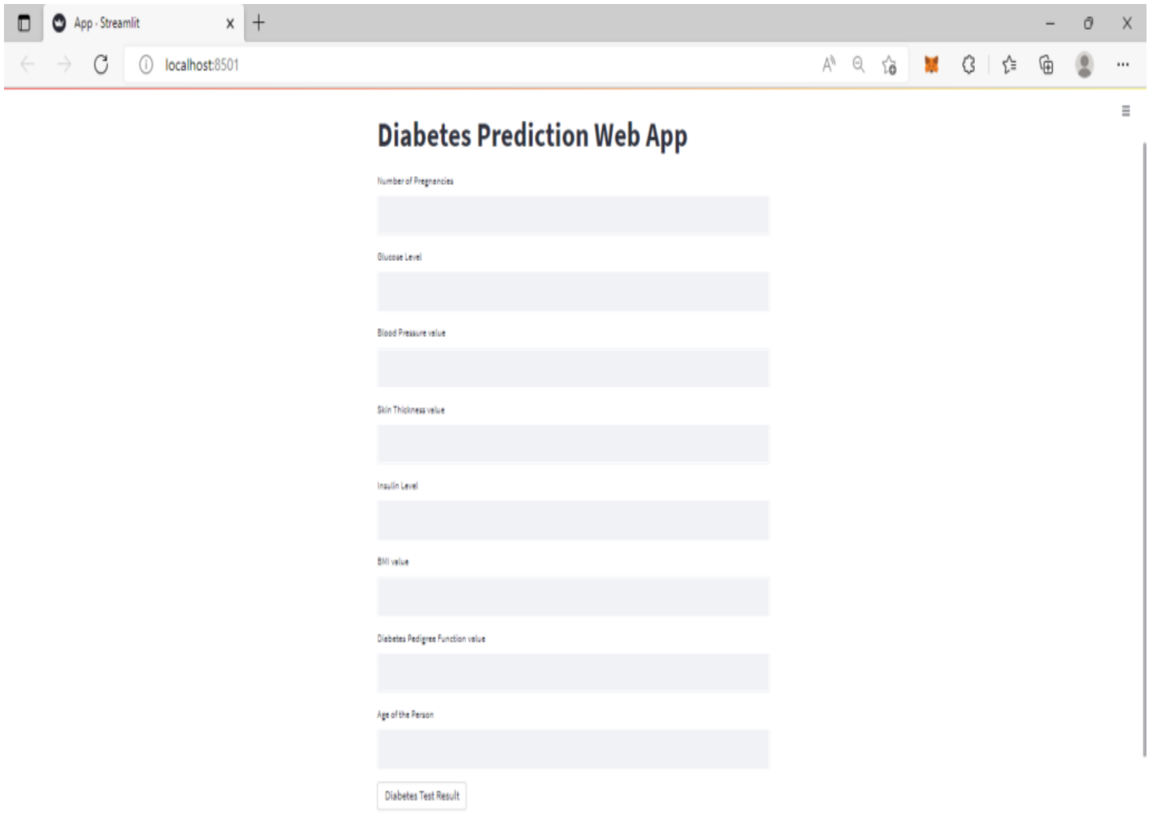
Step 5: Evaluating the model

Once you run this code a new file named trained_model.sav will be saved in the project folder.

Program code for interacting with diabetes prediction

```
import numpy as np
import pickle
import streamlit as st

# Load the saved model
loaded_model = pickle.load(open('C:/Users/ELCOT/Downloads/trained_model.sav', 'rb'))

# Create a function for Prediction
def diabetes_prediction(input_data):

    # Change 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):
      return 'The person is not diabetic'
    else:
      return 'The person is diabetic'

def main():

    # Give a title
    st.title('Diabetes Prediction Web App')

    # To get the input data from the user
    Pregnancies = st.text_input('Number of Pregnancies')
    Glucose = st.text_input('Glucose Level')
    BloodPressure = st.text_input('Blood Pressure value')
    SkinThickness = st.text_input('Skin Thickness value')
    Insulin = st.text_input('Insulin Level')
    BMI = st.text_input('BMI value')
    DiabetesPedigreeFunction = st.text_input('Diabetes Pedigree
Function value')
    Age = st.text_input('Age of the Person')

    # Code for Prediction
    diagnosis = ''

    # Create a button for Prediction

    if st.button('Diabetes Test Result'):
        diagnosis = diabetes_prediction([Pregnancies, Glucose,
BloodPressure, SkinThickness, Insulin, BMI,
DiabetesPedigreeFunction, Age])

    st.success(diagnosis)

if __name__ == '__main__':
    main()
```

Output:



Diabetes Prediction Web App

Number of Pregnancies

Glucose Level

Blood Pressure value

Skin Thickness value

Insulin Level

BMI value

Diabetes Pedigree Function value

Age of the Person

Diabetes Test Result