Phase 4: Development Part 2

AI Based Diabetes Prediction System

Introduction:

In this phase of our project, we will continue to develop a diabetes prediction system. The aim is to leverage machine learning techniques to predict the likelihood of an individual having diabetes based on relevant features and data. To accomplish this, we will follow a structured approach that includes the following key steps:

Step 1:Selecting a Machine Learning Algorithm

We will carefully choose a machine learning algorithm that suits our diabetes prediction task. The selected algorithm will play a vital role in the accuracy and reliability of our model. After thorough consideration, we have chosen the Random Forest algorithm for its capability to handle complex relationships and achieve high predictive accuracy.

We have chosen the Random Forest algorithm for several reasons:

- **Complex Data Relationships:** Diabetes prediction often involves complex and non-linear relationships between various factors like age, BMI, glucose levels, and genetic predisposition. Random Forest excels at capturing such intricate patterns in the data.
- **Robustness:** Random Forest is less prone to overfitting compared to individual decision trees, which is essential when working with medical datasets that may contain noise and outliers.
- **Feature Importance:** The algorithm provides a measure of feature importance, allowing us to understand which features are most influential in predicting diabetes.
- **Scalability:** Random Forest can handle datasets with many features, making it suitable for our task.

Step 2:Training the Model

Once we have selected the Random Forest algorithm, we will proceed to train the model. Model training is a critical step where the algorithm learns from historical data. It adapts its internal parameters to make accurate predictions based on the features and outcomes provided in the dataset.

- **Data Preparation:** We will organize and preprocess the dataset, ensuring that it is suitable for model training. This may include handling missing values, scaling features, and encoding categorical data.
- **Splitting the Data:** We will divide the dataset into two parts: a training set and a testing set. The training set will be used to teach the model, while the testing set will serve as a holdout for evaluating the model's performance.
- **Model Training:** The selected machine learning algorithm will be trained on the training dataset. During this process, the model will adjust its internal parameters to minimize prediction errors.

Step 3:Evaluating Model Performance

After the model has been trained, we must assess its performance to determine how well it can predict diabetes. We will use a separate dataset that the model has not seen during training for evaluation. Several key metrics, including accuracy, precision, recall, F1 score, and the ROC AUC score, will be computed.

- **Accuracy:** Accuracy is a commonly used evaluation metric in machine learning, particularly for classification tasks. It measures the overall correctness of a model's predictions by comparing the number of correct predictions (true positives and true negatives) to the total number of predictions made.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

- **Precision:** Precision is a commonly used evaluation metric in machine learning, especially in the context of classification tasks. It measures the accuracy of positive predictions made by a model, focusing on how many of the instances predicted as positive are actually correct.

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP) + False Positives (FP)}}$$

- **Recall:** Recall, also known as sensitivity or true positive rate, is an important evaluation metric in machine learning, especially for classification tasks. Recall measures the ability of a model to correctly identify all the relevant instances from the total number of actual positive instances.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP) + False Negatives (FN)}}$$

➢ **F1 score:** The F1 score is a commonly used evaluation metric in machine learning, especially for classification tasks. It provides a balance between precision and recall, two important metrics for assessing the performance of a model.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

➢ **ROC AUC:** The ROC AUC (Receiver Operating Characteristic Area Under the Curve) is an evaluation metric commonly used in machine learning, particularly for binary classification tasks. It assesses the performance of a model by measuring its ability to distinguish between the two classes (e.g., positive and negative).

➢ **Confusion Matrix:** A confusion matrix is a table used in machine learning for evaluating the performance of a classification model, particularly in binary classification tasks (tasks with two classes, such as positive and negative). The confusion matrix provides a detailed breakdown of the model's predictions and how they align with the actual ground truth.

	Predicted Negative	*Predicted Positive*
Actual Negative	*TN*	*FP*
Actual Positive	*FN*	*TP*

- **True Positives (TP):** The model correctly identified instances that belong to the positive class.
- **True Negatives (TN):** The model correctly identified instances that belong to the negative class.
- **False Positives (FP):** The model incorrectly predicted instances as positive when they actually belong to the negative class.
- **False Negatives (FN):** The model incorrectly predicted instances as negative when they actually belong to the positive class.

Dataset Link:

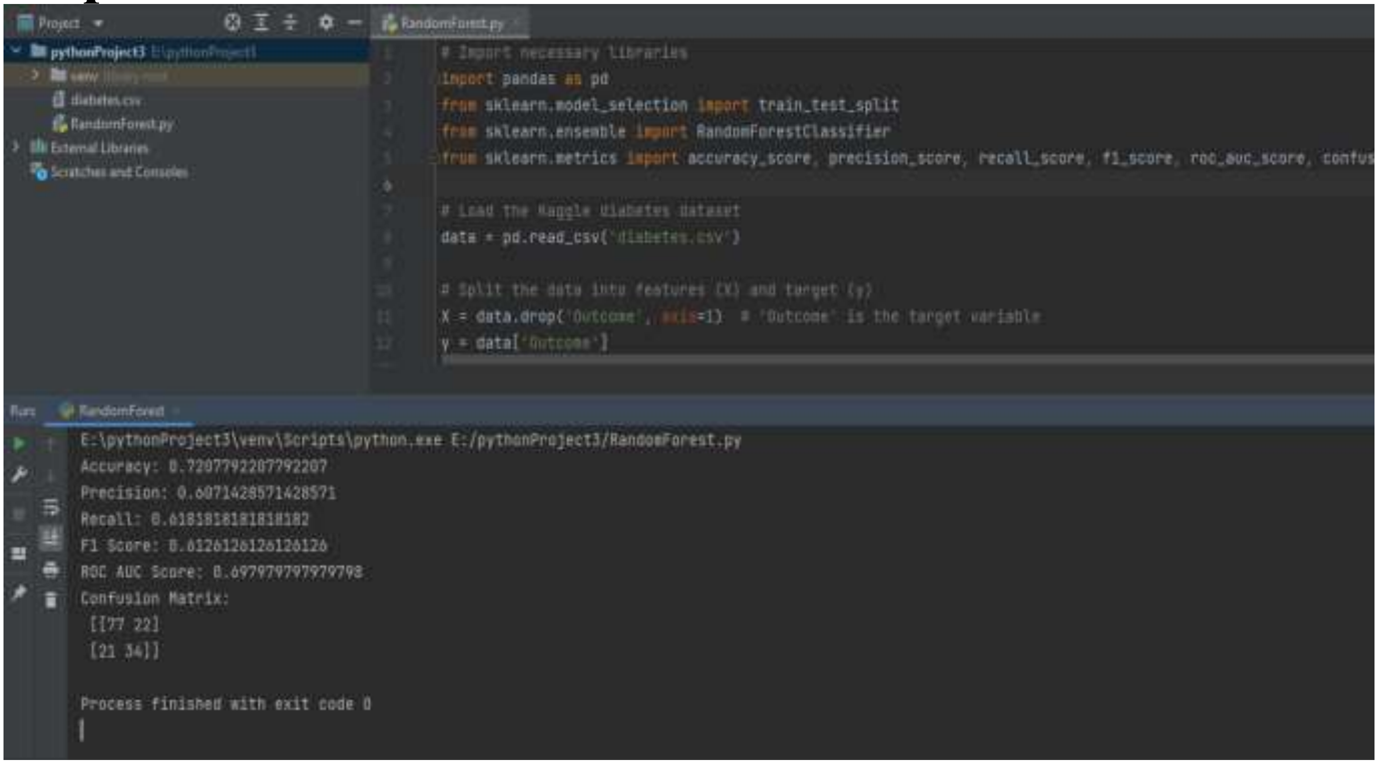
https://www.kaggle.com/datasets/mathchi/diabetes-data-set

Program:

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, roc_auc_score, confusion_matrix
```

```
# Load the Kaggle diabetes dataset
data = pd.read_csv('diabetes.csv')

# Split the data into features (X) and target (y)
X = data.drop('Outcome', axis=1)  # 'Outcome' is the target variable
y = data['Outcome']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Create a RandomForestClassifier
model = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the model on the training data
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Evaluate the model's performance
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)

# Output the results
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
print("ROC AUC Score:", roc_auc)
print("Confusion Matrix:\n", confusion)
```

Output:



```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confus

# Load the Kaggle diabetes dataset
data = pd.read_csv('diabetes.csv')

# Split the data into features (X) and target (y)
X = data.drop('Outcome', axis=1)  # 'Outcome' is the target variable
y = data['Outcome']
```

```
E:\pythonProject3\venv\Scripts\python.exe E:/pythonProject3/RandomForest.py
Accuracy: 0.7207792207792207
Precision: 0.6071428571428571
Recall: 0.6181818181818182
F1 Score: 0.6126126126126126
ROC AUC Score: 0.697979797979798
Confusion Matrix:
 [[77 22]
 [21 34]]

Process finished with exit code 0
```

Conclusion:

In this phase, we are taking significant steps toward building a robust and effective diabetes prediction system. The process involves selecting the right machine learning algorithm, training the model, and rigorously evaluating its performance. The success of our system will ultimately depend on the careful consideration of these factors and the continuous improvement of our model to better predict diabetes in individuals.

As we move forward, we will refine our model, explore additional feature engineering, and fine-tune hyperparameters to enhance the accuracy and reliability of our diabetes prediction system. By addressing the key steps outlined in this document, we aim to develop a valuable tool that can aid in the early diagnosis and management of diabetes