AI BASED DIABETES PREDICTION SYSTEM

PHASE 2 SUBMISSION

Problem Statement

The main objective is to develop an AI Based Diabetes Prediction System to predict whether a patient has diabetics or not, based on the diagnostic measurements gathered in the database using Machine learning algorithms.

Algorithm used:

Out of the machine learning algorithms used,Random tree algorithm is chosen since it has better accuracy and the following steps are taken;

We'll start with importing Pandas and NumPy into our python environment and loading a .csv dataset into a pandas dataframe named df. To see the first five records from the dataset we use pandas df.head() function. We'll also use seaborn and matplotlib for visualization.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
importing dataset
df = pd.read_csv('../input/pima-indians-diabetes-database/diabetes.csv')
df.head()
```

Before starting to analyze the data and draw any conclusions, it is essential to understand the presence of missing values in any dataset. To do so the simplest way is to use df.info function which will provide us the column names with the number of non null values in each column.

```
df.dtypes
  df.info()
```

Data Visualization

The correlation between each columns are visualized using heatmap. From the output, the lighter colors indicate more correlation. We notice the correlation between pairs of features, like age and pregnancies, or BMI and skin thickness, etc.

sns.heatmap(df.corr())

To plot pairwise relationships in a dataset we us sns.pairplot() function a labelled datapoints based on the target variable classes..

pair plot(df,hue='Outcome')

Classification

We need to separate the dataset into features and target variables. Following the popular convention, we call the dataframe with feature variables as X and the one with target variable as y.

```
X=df.drop('Outcome',axis=1)
y=df['Outcome']
```

Using sklearn's train_test_split, we split the feature (X) and target (y) dataframes into a training set (80%) and testing set (20%). Training set is used for building classification model and testing set is used for evaluating the performance of the model.Before implementing classification algorithm, we scale the feature variables of our dataset using sklearn's StandardScaler() function.This function standardize the features by removing the mean and scaling to unit variance.

Training and Evaluating Model

We'll be using a machine simple learning model called Random Forest Classifier. We train the model with standard parameters using the training dataset. The trained model is saved as "**rcf"**. We evaluate the performance of our model using test dataset.

Plotting decision boundaries

A decision boundaries plot works well only with two features. Our data has eight features, but we still can plot decision boundaries by choosing which features to use. We plot decision boundary for each two possible features and see how well the model classifies the patients. For a detailed evaluation of our model, we look at the confusion matrix.

ROC curve

We will plot a ROC curve to determine the working of our model, If the Area Under the Receiver Operating Characteristic Curve (**ROC AUC**) score is high . This implies that the classification model is good enough to detect the diabetic patient.

CONCLUSION:

Using Random forest algorithm in our prediction system, we can evaluate the performance using the accuracy score, comparing the performance between train and test data and produce accurate prediction values of whether a patient has diabetes or not.