AI-BASED DIABETES PREDICTION SYSTEM

TEAM MEMBER

720421104033 : Mohammed Rishil A

CMS COLLEGE OF ENGINEERING AND TECHNOLOGY

AI_Phase-3 Document Submission

Project : **AI-Based Diabetes Prediction System**

Phase 3 : Development Part 1

Topic : start building the AI-based diabetes prediction system by loading and pre-processing the dataset.

AI-Based Diabetes Prediction System

INTRODUCTION :

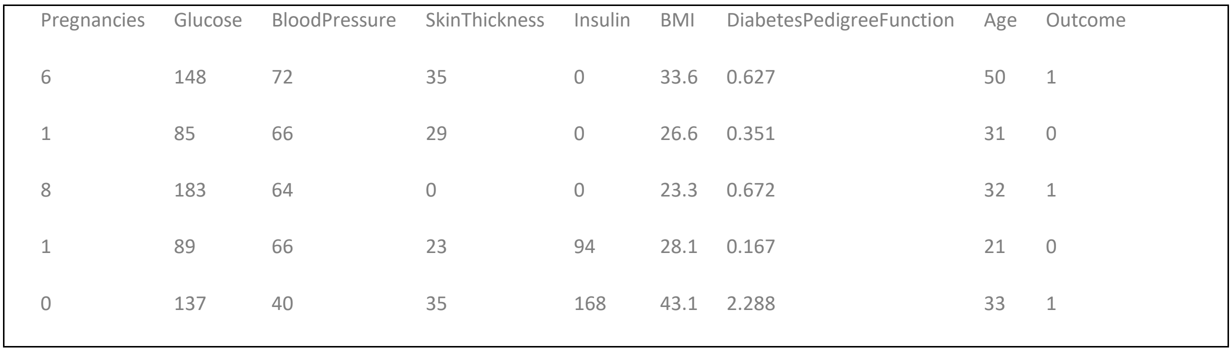
- AI in Diabetes helps to predict or Detect Diabetes. Any neglect in health can have a high cost for the patients and the medical practitioner. It becomes challenging for the patient to trust that this decision is taken by the machine that does not explain how it reaches a particular conclusion. So the use of the Explainable AI is mandatory in predicting disease that will help gain the confidence of an AI system result.

- Explainable AI helps to get the fair and correct output without errors. Generative AI has many potential uses in healthcare, including drug discovery, disease diagnosis, patient care, medical imaging, and medical research.

- This journey begins with the fundamental steps of data loading and pre-processing. We will explore how to import essential libraries, load the diabetes dataset, and perform critical pre-processing steps.

- Data pre-processing is crucial as it helps clean , format and prepare the data for further analysis. This includes handling missing values , encoding categorical variables and ensuring that the data is appropriately scaled.

Given dataset :



Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1

Necessary steps to follow :

1.Import libraries :

Start by importing the necessary libraries:

Program :

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model  import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

2. loading the dataset :

Load your dataset into a pandas data frame. You can typically find diabetes dataset in csv format b ut you can adapt this code to other format as needed.

Program :

```
Import pandas as pd
data = pd.read_csv("/kaggle/input/diabetes-data-set/diabetes.csv")
data.head()
```

3. Exploratory Data Analysis :

Peform EDA to understand the data better.

Program :

```
data.describe()

data.isnull().sum()

data['BMI'] = data['BMI'].replace(0,data['BMI'].mean())
data['BloodPressure'] = data['BloodPressure'].replace(0,data['BloodPressure'].mean())
data['Glucose'] = data['Glucose'].replace(0,data['Glucose'].mean())
data['Insulin'] = data['Insulin'].replace(0,data['Insulin'].mean())
data['SkinThickness'] = data['SkinThickness'].replace(0,data['SkinThickness'].mean())

#import the required libraries

import matplotlib.pyplot as plt
import seaborn as sns
fig, ax = plt.subplots(figsize=(15,10))
sns.boxplot(data=data, width= 0.5,ax=ax,  fliersize=3)
```

4. split the dataset :

Split your data into training and testing sets. This helps to evaluate the model performance better .

```
X = data.drop(columns = ['Outcome'])
y = data['Outcome']
```

5. feature scaling :

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25,random_state=0)
X_train.shape, X_test.shape

import pickle
##standard Scaling- Standardization
def scaler_standard(X_train, X_test):
    #scaling the data
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    #saving the model
    file = open('standardScalar.pkl','wb')
    pickle.dump(scaler,file)
    file.close()

    return X_train_scaled, X_test_scaled

X_train_scaled, X_test_scaled = scaler_standard(X_train, X_test)

X_train_scaled

log_reg = LogisticRegression()

log_reg.fit(X_train_scaled,y_train)
```

Importance of loading and pre-processing dataset :

- One of the main advantages of pre-processing data is that it helps to improve the accuracy of the model. By cleaning and formatting the data, we can ensure that the algorithm is only considering relevant information and that it is not being influenced by any irrelevant or incorrect data. This can lead to a more accurate and robust model.
- Another advantage of pre-processing data is that it can help to reduce the time and resources required to train the model. By removing irrelevant or redundant data, we can reduce the amount of data that the algorithm needs to process, which can greatly reduce the amount of time and resources required to train the model.

1. Loading the dataset :

a. Identify the dataset

b. Load the dataset

c. Pre-process the dataset

Program :

```
import pandas as pd

import numpy as np

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model  import LogisticRegression

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

data = pd.read_csv('/kaggle/input/diabetes-data-set/diabetes.csv')

data.head()
```

output :

```
Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  DiabetesPedigreeFunction  Age  Outcome
          6      148             72             35        0  33.6                     0.627   50        1
          1       85             66             29        0  26.6                     0.351   31        0
          8      183             64              0        0  23.3                     0.672   32        1
          1       89             66             23       94  28.1                     0.167   21        0
          0      137             40             35      168  43.1                     2.288   33        1
```

```
data.describe()
data.isnull().sum()
data['BMI'] = data['BMI'].replace(0,data['BMI'].mean())
data['BloodPressure'] = data['BloodPressure'].replace(0,data['BloodPressure'].mean())
data['Glucose'] = data['Glucose'].replace(0,data['Glucose'].mean())
data['Insulin'] = data['Insulin'].replace(0,data['Insulin'].mean())
data['SkinThickness'] = data['SkinThickness'].replace(0,data['SkinThickness'].mean())
```

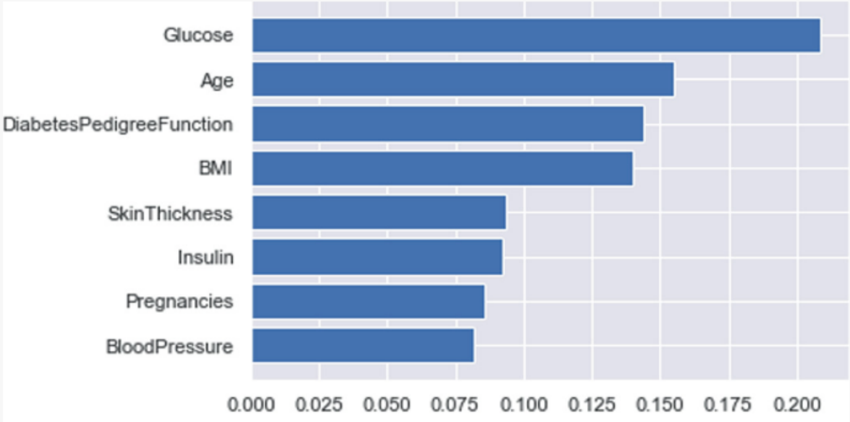
2. Pre-processing the dataset :

```
import matplotlib.pyplot as plt
import seaborn as sns
fig, ax = plt.subplots(figsize=(15,10))
sns.boxplot(data=data, width= 0.5,ax=ax,  fliersize=3)
```

```
X = data.drop(columns = ['Outcome'])
y = data['Outcome']
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25,random_state=0)
X_train.shape, X_test.shape
```

```
import pickle
##standard Scaling- Standardization

def scaler_standard(X_train, X_test):

    #scaling the data
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    #saving the model
    file = open('standardScalar.pkl','wb')
    pickle.dump(scaler,file)
    file.close()

    return X_train_scaled, X_test_scaled

X_train_scaled, X_test_scaled = scaler_standard(X_train, X_test)

X_train_scaled

log_reg = LogisticRegression()

log_reg.fit(X_train_scaled,y_train)
```



conclusion :

- After using all these patient records, we are able to build a model to accurately predict whether or not the patients in the dataset have diabetes or not along with that we were able to draw some insights from the data via data analysis and visualization.
- This study also shows that apart from the choice of algorithms, there are other factors that could improve the accuracy and runtimes of the model, such as: data pre-processing, removal of redundant and null values, normalization, cross-validation, feature selection, and usage of ensemble techniques.