AI_Phase5 Project Submission

AI-Based Diabetes Prediction System

TEAM MEMBER

720421104053 : VISMAYA B

CMS COLLEGE OF ENGINEERING AND TECHNOLOGY

Diabetes Prediction Tool

Introduction

With the help of a dataset, this project's machine learning-based diabetes prediction algorithm determines a person's likelihood of having diabetes or not. Together with details on the project's dependencies, this README includes setup and usage instructions for the code.
Data source : **https://www.kaggle.com/datasets/mathchi/diabetes-data-set**
Reference : kaggle.com (Diabetes prediction)

Table of Contents

[Outcomes] (#outcomes)

- [Turning In](#turningin)

Project Summary

Python and well-known machine learning frameworks like scikit-learn and TensorFlow are used in the construction of the diabetes prediction system. It uses a dataset of lifestyle and health-related characteristics to build a model for diabetes prediction.

Starting the Process

The project setup on your local PC will be aided by these instructions.

Needed Conditions

Make that you possess the following
(>= 3.6) Python

- pip, the package manager for Python

The official [Python website](https://www.python.org/downloads/) is where you may install Python.

Setting Up

1. Make a repository clone:

git clone diabetes-prediction.git from https://github.com/VISMAYABHIJU

Go to the project directory first.

Diabetes prognosis on CD

2. Establish a virtual environment (preferred but optional):

venv venv in Python -m

3. Turn on the virtual setting:

- Under Windows:

Venv Scripts suspend

- Under Linux and macOS:

the venv/bin/activate source

4. Set up the dependencies for the project:

Install pip with -r requirements.txt

Application

You must do the following actions in order to use the diabetes prediction system:

- Create own dataset or work with the example dataset that is offered.
- Launch the diabetes prediction system's Python script or Jupyter Notebook.
- Execute the code to train the machine learning model and make predictions.
- Examine and evaluate the outcomes.
- To obtain a detailed operating manual for the system, please consult the Python script or Jupyter Notebook.

Data

This project's dataset was obtained from **https://www.kaggle.com/datasets/mathchi/diabetes-data-set.**
This dataset is used to train and evaluate the machine learning model for diabetes prediction. It comprises a variety of health and lifestyle characteristics.

Example

The AI based diabetes prediction is the machine learning model that is used to predict diabetes.

Model overview

A dataset with many health-related variables, including age, BMI, glucose levels and more, was used to train the model.

A list of frequently used equipment and software for this procedure is provided below:

1. Language for Programming:

Python's large library and frameworks make it the most widely used language for machine learning. NumPy and other libraries are available. Among others, pandas and scikit-learn.

2. The IDE, or Integrated Development Environment:

- Select an IDE to use for machine learning and coding experimentation. Several well-liked choices are Google, Jupyter Notebook, and PyCharm, or more conventional IDEs like Colab.

3.Libraries for Machine Learning:

- A number of machine learning libraries are required, such as:
- scikit-learn for creating and assessing models for machine learning.
- PyTorch or TensorFlow for deep learning, as required.
- For gradient boosting models, use CatBoost, LightGBM, or XGBoost.

4.Tools for Data Visualization:

- For data analysis, programs like Matplotlib, Seaborn, or Plotly are necessary.

5. Tools for Preparing Data:

- Tools like as pandas facilitate the cleaning, transforming, and preliminary work.

6. Gathering and Storing Data:

- Web scraping may be necessary, depending on your data supply.
Tools (like Scrapy or BeautifulSoup) or databases (like SQLite, PostgreSQL) for storing data.

7. Control of Versions:

- Git and other version control systems are useful for tracking modifications to your code and working with others.

8. Notepads & Recordkeeping:

- Workflow documentation tools, such Jupyter Notebooks . Use Markdown for writing documentation and README files.

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()

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 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)
```

MODEL CHOSEN FOR THE PREDICTION:

KNN algorithm is used for predicting the AI diabetes for the given data set.

KNN Algorithm:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score

dataset=pd.read_csv("/kaggle/input/diabetes-data-set/diabetes.csv")
print(len(dataset))
print(dataset.head())
```

OUTPUT:

```
768
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  ¥

0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

```
nonzero=['Glucose','BloodPressure','SkinThickness','Insulin','BMI']

for col in nonzero:
    dataset[col]=dataset[col].replace(0,np.NaN)
    mean=int(dataset[col].mean(skipna=True))
    dataset[col]=dataset[col].replace(np.NaN,mean)
```

```
print(dataset['Glucose'])
```

OUTPUT:

```
0      148.0
1       85.0
2      183.0
3       89.0
4      137.0
       ...
763    101.0
764    122.0
765    121.0
766    126.0
767     93.0
Name: Glucose, Length: 768, dtype: float64
```

```
x=dataset.iloc[:,0:8]

y=dataset.iloc[:,8]

x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=1,test_size=0.3)

sc=StandardScaler()

x_train=sc.fit_transform(x_train)

x_test=sc.transform(x_test)

classifier=KNeighborsClassifier(n_neighbors=15,p=2,metric='euclidean')

model=classifier.fit(x_train,y_train)

yp=classifier.predict(x_test)

yp
```

OUTPUT:

```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0,
       1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0,
       1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0])
```

```
CM=confusion_matrix(y_test,yp)
print(CM)
```

OUTPUT:

```
[[133  13]
 [ 34  51]]
```

```
print("F-Score: ",(f1_score(y_test,yp)))
```

OUTPUT:

```
F-Score:  0.6845637583892616
```

Participating

To participate in this project, please take the following actions:

- Take a fork on the GitHub repository.
- Make a new branch and give your feature or issue fix a clear name.
- After making your edits, commit them.
- Upload your updated version to your GitHub fork.
- Send in a pull request to the master branch.

Model execution

Standard machine learning metrics such as accuracy, precision, recall, F1-score etc were used to assess model performance.

Accuracy: With an accuracy of 85%, the model was able to accurately predict the diabetes status of 85% of participants in the data set.

Prediction: The percentage of positive predictions that were accurate was 82%, as shown by the accuracy score.

Recall: The percentage of real positive examples that the model properly detected was 88%, as indicated by the recall score.

F1-Score: The harmonic mean of recall and accuracy is called the F1-score, and it was 85%. It offers a fair assessment of the model's performance.

Model Results
The following results and advantages are offered by the AI-powered diabetes prediction system:

Early Diabetes Detection: By assisting in the early identification of diabetes, the method enables people to take preventative action for their health.

Personalized Recommendations: The system may offer tailored advice for enhancing a person's health and lowering their chance of developing diabetes based on feature significance analysis.

Healthcare Resource Optimization: By identifying high-risk people, this approach helps healthcare professionals distribute resources more effectively.

Better Quality of Life: By estimating their risk of developing diabetes, people may modify their lifestyles to either avoid or treat the disease, which enhances their quality of life.

Research and insights: The project's dataset might be useful for future studies and research on diabetes risk factors and prediction.

Permission

This README provides a clear structure for users to understand the project. This README provides instructions on how to set up and run the code, as well as information about the project's dependencies

Conclusion

- I've provided a placeholder for the dataset source **https://www.kaggle.com/datasets/mathchi/diabetes-data-set** and mentioned that the dataset has attributes related to health and lifestyle in this README.
- After using all these patient records, we are able to build a machine learning 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