PHASE – 1 DOCUMENT SUBMISSION

Project Name: AI Based Diabetes Prediction System

**Student Name: PREMKUMAR C J**

**Reg No: 312821121023**

AI-Based Diabetes Prediction System

# Abstract:

Diabetes is a chronic medical condition affecting millions of individuals worldwide. Early detection and proactive management of diabetes are crucial for preventing complications and improving patients' quality of life. This abstract presents an AI-based Diabetes Prediction System (DPS) designed to assist healthcare professionals in identifying individuals at risk of developing diabetes. The system employs

advanced machine learning and data analysis techniques to predict the likelihood of diabetes onset using various patient-related parameters.

### Context:

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective is to predict based on diagnostic measurements whether a patient has diabetes.

**Dataset Link: [https://www.kaggle.com/datasets/mathchi/diabetes-data-set](https://www.kaggle.com/datasets/mathchi/diabetes-data-set" \t "_blank)**

# Module 1: Data Collection and Preprocessing

The first module of the DPS focuses on the collection and preprocessing of relevant medical and lifestyle data. Data sources may include electronic health records, wearable devices, and patient surveys. The collected data undergoes rigorous preprocessing, including cleaning, normalization, and feature extraction, to ensure its quality and suitability for subsequent analysis.

# Module 2: Feature Selection and Engineering

Module 2 involves the selection and engineering of predictive features. Advanced feature selection

techniques, such as recursive feature elimination and correlation analysis, are employed to identify the most informative variables from the dataset. Additionally, feature engineering methods are applied to create new relevant features that enhance the model's predictive performance.

# Module 3: Machine Learning Models

This module focuses on the development of machine learning models for diabetes prediction. Various algorithms, including logistic regression, decision trees, random forests, support vector machines, and deep neural networks, are explored and trained on the preprocessed data. Model hyperparameters are fine-tuned using cross-validation to maximize predictive accuracy.

# Module 4: Model Evaluation and Validation

Module 4 assesses the performance of the trained models using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC- ROC). The models are validated on independent datasets to ensure their generalizability and reliability.

# Module 5: Real-time Predictive Interface

The final module of the DPS provides a user-friendly and real-time predictive interface for healthcare professionals. This interface allows inputting patient information and obtaining instant predictions

regarding their risk of developing diabetes. The system also generates interpretable explanations for its predictions to assist healthcare providers in making informed decisions.

# Python Program

import numpy as np *# linear algebra*

import pandas as pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

import matplotlib.pyplot as plt

import seaborn as sns

In [2]:

dataset=pd.read\_csv("/kaggle/input/diabetes-data-set/diabetes.csv")

In [3]:

dataset.head()

Out[3]:

|  | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeFunction | Age | Outcome |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 6 | 148 | 72 | 35 | 0 | 33.6 | 0.627 | 50 | 1 |
| 1 | 1 | 85 | 66 | 29 | 0 | 26.6 | 0.351 | 31 | 0 |
| 2 | 8 | 183 | 64 | 0 | 0 | 23.3 | 0.672 | 32 | 1 |
| 3 | 1 | 89 | 66 | 23 | 94 | 28.1 | 0.167 | 21 | 0 |
| 4 | 0 | 137 | 40 | 35 | 168 | 43.1 | 2.288 | 33 | 1 |

In [4]:

dataset.shape

Out[4]:

(768, 9)

In [5]:

*#check if null value is present*

dataset.isnull().values.any()

Out[5]:

False

In [6]:

dataset.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 768 entries, 0 to 767

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Pregnancies 768 non-null int64

1 Glucose 768 non-null int64

2 BloodPressure 768 non-null int64

3 SkinThickness 768 non-null int64

4 Insulin 768 non-null int64

5 BMI 768 non-null float64

6 DiabetesPedigreeFunction 768 non-null float64

7 Age 768 non-null int64

8 Outcome 768 non-null int64

dtypes: float64(2), int64(7)

memory usage: 54.1 KB

In [7]:

dataset.describe()

Out[7]:

|  | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeFunction | Age | Outcome |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| count | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 |
| mean | 3.845052 | 120.894531 | 69.105469 | 20.536458 | 79.799479 | 31.992578 | 0.471876 | 33.240885 | 0.348958 |
| std | 3.369578 | 31.972618 | 19.355807 | 15.952218 | 115.244002 | 7.884160 | 0.331329 | 11.760232 | 0.476951 |
| min | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.078000 | 21.000000 | 0.000000 |
| 25% | 1.000000 | 99.000000 | 62.000000 | 0.000000 | 0.000000 | 27.300000 | 0.243750 | 24.000000 | 0.000000 |
| 50% | 3.000000 | 117.000000 | 72.000000 | 23.000000 | 30.500000 | 32.000000 | 0.372500 | 29.000000 | 0.000000 |
| 75% | 6.000000 | 140.250000 | 80.000000 | 32.000000 | 127.250000 | 36.600000 | 0.626250 | 41.000000 | 1.000000 |
| max | 17.000000 | 199.000000 | 122.000000 | 99.000000 | 846.000000 | 67.100000 | 2.420000 | 81.000000 | 1.000000 |

In [8]:

dataset.isnull().sum()

Out[8]:

Pregnancies 0

Glucose 0

BloodPressure 0

SkinThickness 0

Insulin 0

BMI 0

DiabetesPedigreeFunction 0

Age 0

Outcome 0

dtype: int64

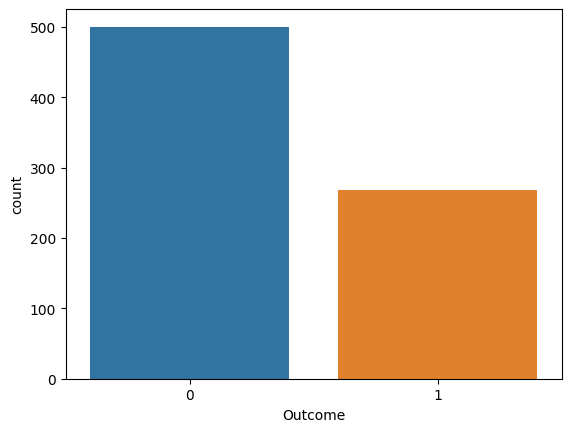
In [9]:

*#data visualization*

sns.countplot(x = 'Outcome',data = dataset)

Out[9]:

<Axes: xlabel='Outcome', ylabel='count'>



In [10]:

*# Pairplot*

sns.pairplot(data = dataset, hue = 'Outcome')

plt.show()

/opt/conda/lib/python3.10/site-packages/seaborn/axisgrid.py:118: UserWarning: The figure layout has changed to tight

self.\_figure.tight\_layout(\*args, \*\*kwargs)

In [11]:

*# Heatmap*

sns.heatmap(dataset.corr(), annot = True)

plt.show()

In [12]:

*# Replacing zero values with NaN*

dataset\_new = dataset

dataset\_new[["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]] = dataset\_new[["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]].replace(0, np.NaN)

In [13]:

*# Count of NaN*

dataset\_new.isnull().sum()

Out[13]:

Pregnancies 0

Glucose 5

BloodPressure 35

SkinThickness 227

Insulin 374

BMI 11

DiabetesPedigreeFunction 0

Age 0

Outcome 0

dtype: int64

In [14]:

*# Replacing NaN with mean values*

dataset\_new["Glucose"].fillna(dataset\_new["Glucose"].mean(), inplace = True)

dataset\_new["BloodPressure"].fillna(dataset\_new["BloodPressure"].mean(), inplace = True)

dataset\_new["SkinThickness"].fillna(dataset\_new["SkinThickness"].mean(), inplace = True)

dataset\_new["Insulin"].fillna(dataset\_new["Insulin"].mean(), inplace = True)

dataset\_new["BMI"].fillna(dataset\_new["BMI"].mean(), inplace = True)

In [15]:

dataset\_new.isnull().sum()

Out[15]:

Pregnancies 0

Glucose 0

BloodPressure 0

SkinThickness 0

Insulin 0

BMI 0

DiabetesPedigreeFunction 0

Age 0

Outcome 0

dtype: int64

In [16]:

*#Logistic regression*

y = dataset\_new['Outcome']

X = dataset\_new.drop('Outcome', axis=1)

In [17]:

*# Splitting X and Y*

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, y, test\_size = 0.20, random\_state = 42, stratify = dataset\_new['Outcome'] )

In [18]:

from sklearn.linear\_model import LogisticRegression

model = LogisticRegression()

model.fit(X\_train, Y\_train)

y\_predict = model.predict(X\_test)

/opt/conda/lib/python3.10/site-packages/sklearn/linear\_model/\_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):

STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:

https://scikit-learn.org/stable/modules/preprocessing.html

Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear\_model.html#logistic-regression

n\_iter\_i = \_check\_optimize\_result(

In [19]:

y\_predict

Out[19]:

array([1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1,

0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0,

0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0,

1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1,

0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1,

0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0])

In [20]:

*# Confusion matrix*

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(Y\_test, y\_predict)

cm

Out[20]:

array([[84, 16],

[25, 29]])

In [21]:

*# Heatmap of Confusion matrix*

sns.heatmap(pd.DataFrame(cm), annot=True)

Out[21]:

<Axes: >

In [22]:

from sklearn.metrics import accuracy\_score

In [23]:

accuracy =accuracy\_score(Y\_test, y\_predict)

accuracy

Out[23]:

0.7337662337662337

In [24]:

*#Example: Let's check whether the person have diabetes or not using some random values*

y\_predict = model.predict([[1,148,72,35,79.799,33.6,0.627,50]])

print(y\_predict)

if y\_predict==1:

print("Diabetic")

else:

print("Non Diabetic")

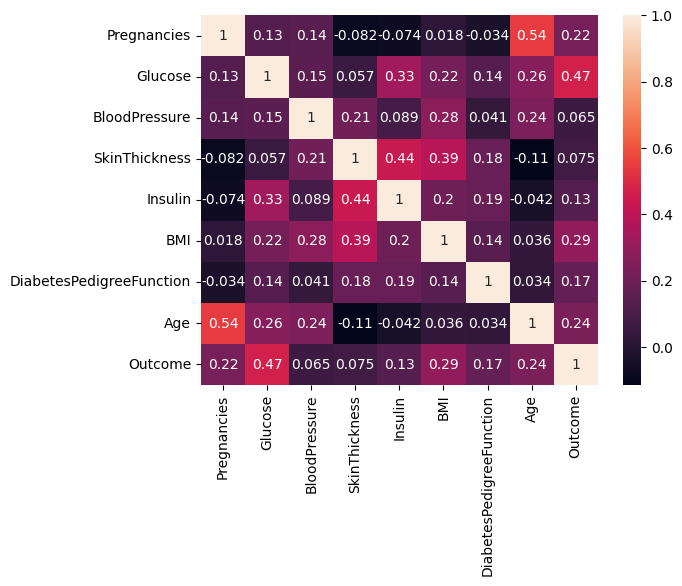
[1]

Diabetic

/opt/conda/lib/python3.10/site-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LogisticRegression was fitted with feature names

warnings.warn(

In [ ]:



# Conclusion

The AI-based Diabetes Prediction System presented in this abstract offers a comprehensive solution for early diabetes risk assessment. By leveraging state-of-the-art machine learning techniques, it empowers healthcare professionals to identify individuals at risk, enabling timely interventions and personalized care plans. This system holds the potential to reduce the burden of diabetes-related complications and

improve the overall health and well-being of affected individuals.