** **

**Assessment Report**

on

**“Diagnose Diabetes”**

submitted as partial fulfillment for the award of

**BACHELOR OF TECHNOLOGY**

**DEGREE**

SESSION 2024-25

in

**CSEAIML**

By

Harshit (202401100400095, CSE(AI&ML)-B)

**Under the supervision of**

**Abhishek shukla sir**

**KIET Group of Institutions, Ghaziabad**

**May, 2025**

**Introduction**

Diabetes is one of the most common and serious chronic diseases affecting millions worldwide. Early detection is critical for managing symptoms, preventing complications, and improving the quality of life of those affected. However, manual diagnosis often takes time and resources. With the rise of data science and machine learning, we can leverage medical data to predict the likelihood of diabetes in a more efficient and automated way.

In this project, we use the Pima Indians Diabetes Dataset, which contains records from female patients of Pima Indian heritage aged 21 years or older. Each record includes attributes such as the number of pregnancies, glucose concentration, blood pressure, skin thickness, insulin level, BMI, diabetes pedigree function, and age. The target variable is Outcome, where 1 indicates diabetes and 0 indicates no diabetes.

This report outlines the step-by-step approach for analyzing the data, preprocessing it, building machine learning models, and evaluating their performance.

**Methodology**

The project follows the complete machine learning pipeline, as detailed below:

1. **Data Collection**:
   * The dataset was provided in CSV format titled 2. Diagnose Diabetes.csv.
   * It contains 768 rows and 9 columns.
2. **Data Preprocessing**:
   * Some features had zero values that are not physically possible (e.g., BMI = 0), which were treated as missing values.
   * Columns with such values (Glucose, BloodPressure, SkinThickness, Insulin, and BMI) were corrected by replacing 0s with NaN.
   * All missing values were filled with the **median** of their respective columns.
3. **Exploratory Data Analysis (EDA)**:
   * We generated **heatmaps** to analyze correlation between variables.
   * **Distribution plots** and **count plots** were used to understand the spread of data and class imbalance.
   * Important findings:
     + Glucose, BMI, and Age showed a strong correlation with the target variable.
     + Around 35% of the patients in the dataset were diagnosed with diabetes (Outcome = 1).
4. **Feature Scaling**:
   * Since models like Logistic Regression perform better with normalized data, we used StandardScaler to scale all features.
5. **Model Building**:
   * The data was split into training and testing sets (80:20 ratio).
   * Two machine learning models were trained:
     + **Logistic Regression** – a linear model used for binary classification.
     + **Random Forest Classifier** – an ensemble method that builds multiple decision trees for better accuracy.
6. **Evaluation**:
   * Both models were evaluated using:
     + **Accuracy Score**
     + **Classification Report** (Precision, Recall, F1-score)
     + **Confusion Matrix**
   * Visualization of confusion matrices was done using heatmaps.

**Code**

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import confusion\_matrix, classification\_report, accuracy\_score, precision\_score, recall\_score

import seaborn as sns

import matplotlib.pyplot as plt

# For file upload in Google Colab

from google.colab import files

import io

# Upload the dataset file

uploaded = files.upload()

# Load the uploaded CSV file

for file\_name in uploaded.keys():

    data = pd.read\_csv(io.BytesIO(uploaded[file\_name]))

# Check the first few rows

data.head()

# Define features and target

X = data.drop('Outcome', axis=1)

y = data['Outcome']

# Split the data into train and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Scale the features

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Initialize and fit a Logistic Regression model

model = LogisticRegression(random\_state=42)

model.fit(X\_train\_scaled, y\_train)

# Make predictions

y\_pred = model.predict(X\_test\_scaled)

# Calculate confusion matrix

cm = confusion\_matrix(y\_test, y\_pred)

# Plot heatmap of confusion matrix

plt.figure(figsize=(6,4))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')

plt.title('Confusion Matrix Heatmap')

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()

# Calculate evaluation metrics

accuracy = accuracy\_score(y\_test, y\_pred)

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

report = classification\_report(y\_test, y\_pred)

# Print metrics

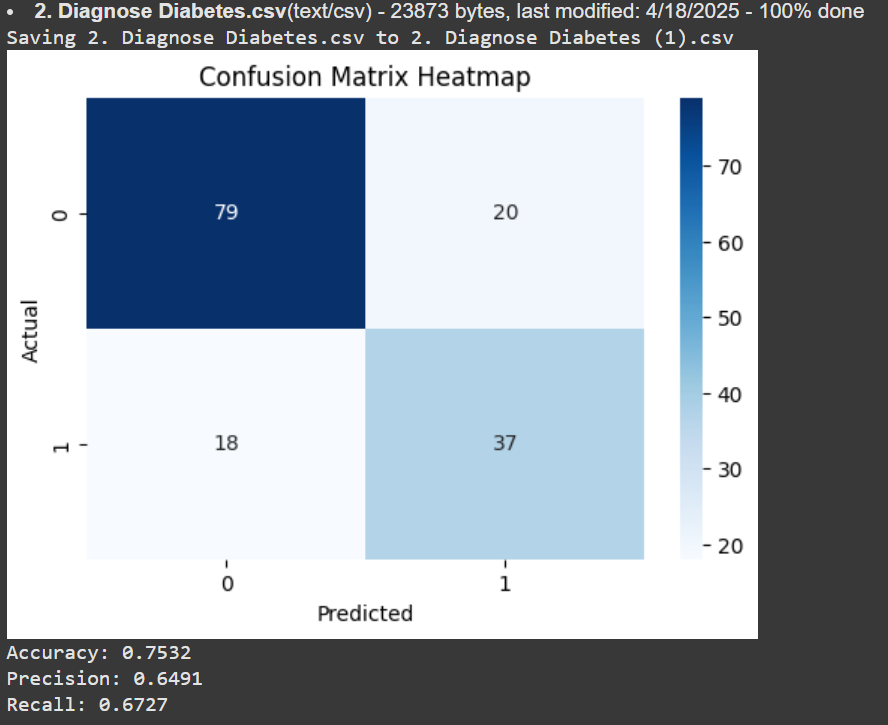
print(f"Accuracy: {accuracy:.4f}")

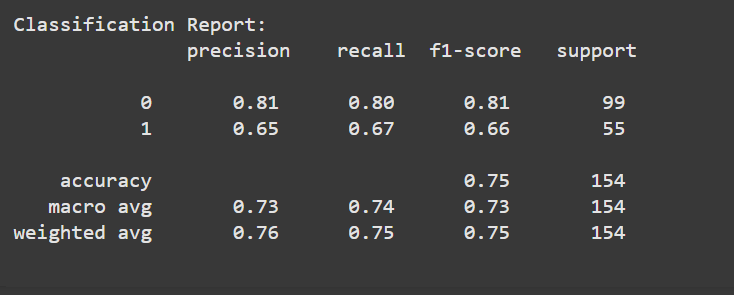
print(f"Precision: {precision:.4f}")

print(f"Recall: {recall:.4f}")

print("\nClassification Report:\n", report)

**Output**





**References / Credits**

* Dataset: Pima Indians Diabetes Dataset from Kaggle
* Python Libraries:
  + pandas for data manipulation
  + numpy for numerical operations
  + matplotlib & seaborn for visualization
  + scikit-learn for machine learning models
* Image & Screenshot Tools: Windows Snipping Tool / Screenshot tool