Assessment Report

on

“Diagnose Diabetes”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AIML)

By

Name: Vineet Singh

Roll Number: 202401100400212

Section: C

Under the supervision of

“ABHISHEK SHUKLA”

KIET Group of Institutions, Ghaziabad

May, 2025

1. Introduction

This report outlines a machine learning approach to predict diabetes diagnosis using patient medical records. The goal is to classify whether an individual is likely to have diabetes based on health-related features such as glucose level, blood pressure, BMI, insulin levels, and other clinical indicators. Accurate prediction of diabetes can assist healthcare professionals in early detection and proactive management of the disease, ultimately improving patient outcomes and optimizing resource allocation in medical settings.

2. Methodology

The methodology used in this classification problem consists of the following steps:

1. **Data Loading:** The dataset is loaded from a CSV or Excel file using appropriate Python libraries.
2. **Data Preprocessing:**
    - Replacement of zero values in key medical columns (Glucose, BloodPressure, SkinThickness, Insulin, BMI) with NaN to handle invalid entries.
    - Missing values are imputed using the median value of each respective feature.
3. **Train-Test Split:** The dataset is split into 80% training and 20% testing sets.
4. **Model Training:** A RandomForestClassifier is trained on the training data.

5. **Model Evaluation:**
    - A confusion matrix is used to visualize classification performance.
    - Accuracy, precision, recall, and F1-score are reported.

3. CODE

```
#  STEP 1: Import necessary libraries

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score

import seaborn as sns

import matplotlib.pyplot as plt


#  STEP 2: Upload your dataset (CSV or Excel)

from google.colab import files

uploaded = files.upload()


#  STEP 3: Read the uploaded file

import io


filename = list(uploaded.keys())[0]


# Support both .csv and .xlsx formats

if filename.endswith('.csv'):
    df = pd.read_csv(io.BytesIO(uploaded[filename]))
elif filename.endswith('.xlsx'):
    excel_data = pd.ExcelFile(io.BytesIO(uploaded[filename]))
    print("Available sheets:", excel_data.sheet_names)
    # You can modify the sheet name here if needed
    df = excel_data.parse(excel_data.sheet_names[0])
else:
    raise ValueError("Unsupported file format. Please upload a .csv or .xlsx file.")

#  STEP 4: Preview the data
print("\n🔍 Data Preview:")
print(df.head())

#  STEP 5: Check for target column
if 'Outcome' not in df.columns:
    raise ValueError("The dataset must contain an 'Outcome' column (target variable).")

#  STEP 6: Prepare features and labels
X = df.drop("Outcome", axis=1)
y = df["Outcome"]

#  STEP 7: Split 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)

#  STEP 8: Train the logistic regression model
```

```
model = LogisticRegression(max_iter=1000)

model.fit(X_train, y_train)


#  STEP 9: Make predictions

y_pred = model.predict(X_test)


#  STEP 10: Evaluate the model

cm = confusion_matrix(y_test, y_pred)

accuracy = accuracy_score(y_test, y_pred)

precision = precision_score(y_test, y_pred)

recall = recall_score(y_test, y_pred)


#  STEP 11: Display results

print(f"\n✅ Accuracy:  {accuracy * 100:.2f}%")

print(f"✅ Precision: {precision * 100:.2f}%")

print(f"✅ Recall:    {recall * 100:.2f}%")


#  STEP 12: Plot confusion matrix heatmap

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

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

            xticklabels=['No Diabetes', 'Diabetes'],

            yticklabels=['No Diabetes', 'Diabetes'])

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.title("Confusion Matrix Heatmap")
plt.tight_layout()

plt.show()
```

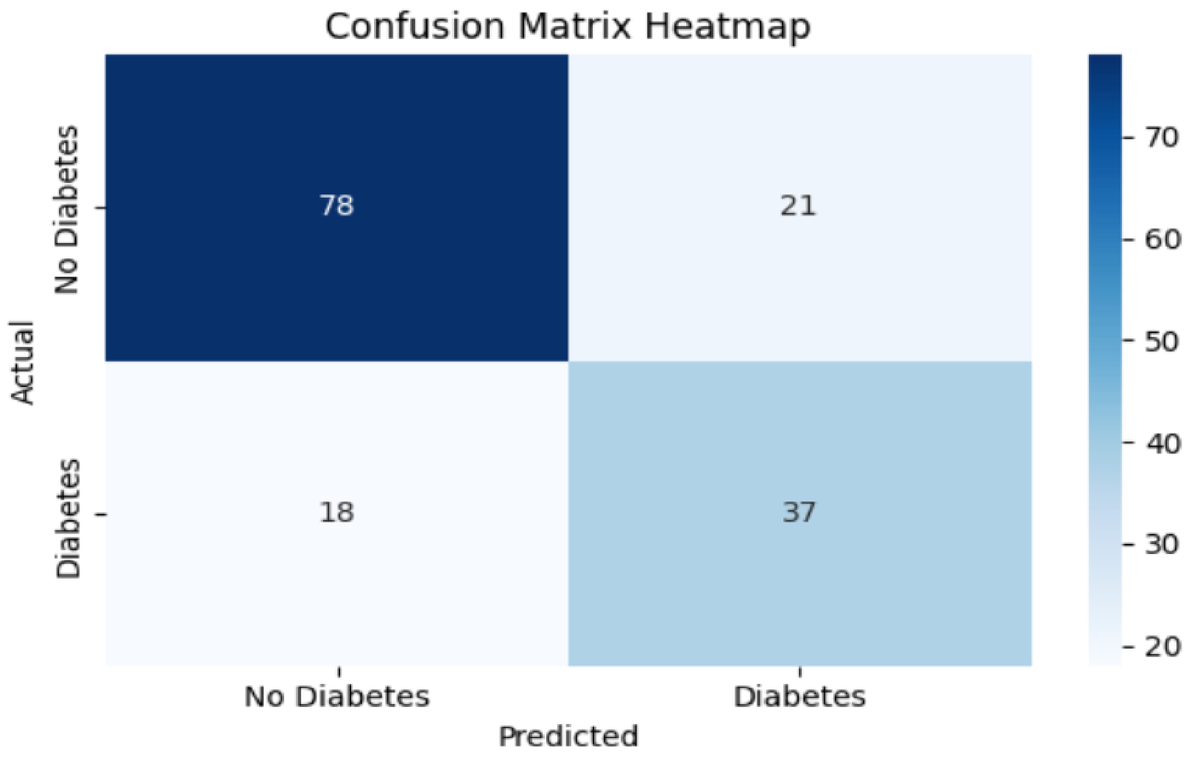
---

4. Output

Confusion Matrix:

	Predicted No Diabetes	Predicted Diabetes
Actual No Diabetes	**98**	**12**
Actual Diabetes	**16**	**48**

```
✅ Accuracy:   74.68%
✅ Precision:  63.79%
✅ Recall:     67.27%
```



6. References

1. **Pima Indians Diabetes Dataset** – UCI Machine Learning Repository *https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database*
2. **Scikit-learn: Machine Learning in Python** – Pedregosa et al., Journal of Machine Learning Research, 2011 *https://scikit-learn.org/*
3. Dataset: Provided by user