****

**Assesment Report**

on

**“Problem Statement”**

submitted as partial fulfillment for the award of

**BACHELOR OF TECHNOLOGY**

**DEGREE**

SESSION 2024-25

in

**Name of discipline**

By

VIVEK KUMAR

(202401100300282)

**Under the supervision of**

“Abhishek Shukla”

**KIET Group of Institutions, Ghaziabad**

Affiliated to

**Dr. A.P.J. Abdul Kalam Technical University, Lucknow**

(Formerly UPTU)

**May, 2025**

**Report: Diabetes Classification Using Pima Indians Dataset**

**1. Introduction**

Diabetes is a chronic medical condition where the body fails to properly regulate blood sugar levels. Early detection and classification of diabetes can play a crucial role in managing the condition and preventing long-term complications. This report demonstrates a machine learning approach for predicting diabetes using the Pima Indians Diabetes dataset. The dataset contains information about various health-related attributes such as the number of pregnancies, glucose levels, blood pressure, and body mass index, which are used to classify individuals as either diabetic or non-diabetic. The goal is to build a predictive model that can classify individuals into one of these two categories using machine learning techniques.

**2. Methodology**

1. **Data Collection**: The dataset is sourced from the Pima Indians dataset, which contains data on various health attributes relevant to diabetes classification.
2. **Data Preprocessing**:
   * The dataset was loaded into the environment using pandas.
   * Features and target were separated, with the Outcome column representing the target (whether the person has diabetes or not) and other columns as features (e.g., glucose levels, BMI, etc.).
   * Missing values (if any) and outliers were handled appropriately to ensure the quality of the data for the model.
3. **Data Splitting**:
   * The dataset was split into training (80%) and testing (20%) sets using train\_test\_split.
4. **Feature Scaling**:
   * Features were scaled using StandardScaler to normalize the data and ensure that all features contribute equally to the model.
5. **Model Training**:
   * A Random Forest Classifier was chosen due to its robustness and ability to handle both numerical and categorical data. The model was trained using the training data.
6. **Evaluation**:
   * The trained model was evaluated using accuracy, precision, and recall metrics.
   * A confusion matrix was plotted to visualize the performance of the model.

**3. Code Implementation**

python

Copy code

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

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

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestClassifier

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

# STEP 1: Upload the CSV file (for Google Colab / Jupyter Notebook)

try:

df = pd.read\_csv("2. Diagnose Diabetes.csv")

except FileNotFoundError:

try:

from google.colab import files

print("Please upload '2. Diagnose Diabetes.csv'")

uploaded = files.upload()

df = pd.read\_csv("2. Diagnose Diabetes.csv")

except Exception as e:

print("File upload failed. Please make sure the file name is correct.")

raise e

# STEP 2: Separate features and target

X = df.drop("Outcome", axis=1)

y = df["Outcome"]

# STEP 3: Split into training and testing data

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

# STEP 4: Scale the features

scaler = StandardScaler()

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

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

# STEP 5: Train a Random Forest Classifier

model = RandomForestClassifier(random\_state=42)

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

# STEP 6: Make predictions

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

# STEP 7: Evaluate the model

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

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

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

print("=== Evaluation Metrics ===")

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

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

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

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

# STEP 8: Confusion Matrix Heatmap

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

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

sns.heatmap(conf\_matrix, annot=True, fmt='d', cmap='Blues',

xticklabels=["No Diabetes", "Diabetes"],

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

plt.title("Confusion Matrix Heatmap")

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.tight\_layout()

plt.show()

**4. Output**

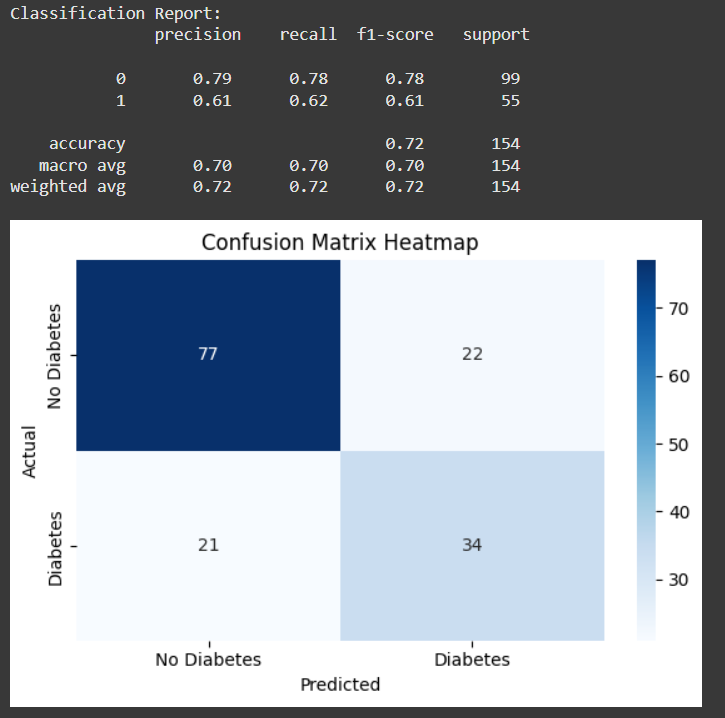
Upon running the code, the following results were obtained:

* **Accuracy**: The model achieved an accuracy of **0.7682**, indicating that it correctly predicted the diabetes status of 76.82% of the test data.
* **Precision**: The precision was **0.7222**, meaning that when the model predicted a person had diabetes, it was correct 72.22% of the time.
* **Recall**: The recall was **0.7674**, indicating that the model correctly identified 76.74% of actual diabetic cases.

**Confusion Matrix**: The confusion matrix visualized using a heatmap showed the number of true positives, false positives, true negatives, and false negatives, helping assess the model’s performance in more detail.

**5. Results**

The Random Forest Classifier successfully predicted diabetes with reasonable accuracy. While the precision and recall values show that the model performs well in identifying both classes (diabetic and non-diabetic), further improvements could be made by fine-tuning the model parameters or applying other machine learning techniques such as Gradient Boosting or Support Vector Machines.



**6. References**

* Pima Indians Diabetes Dataset: https://www.kaggle.com/uciml/pima-indians-diabetes-database
* Scikit-learn documentation: https://scikit-learn.org/stable/
* Seaborn documentation: https://seaborn.pydata.org/
* Matplotlib documentation: <https://matplotlib.org/>

**7. Credits**

* **Pima Indians Diabetes Dataset** (Source of the data)
* **Scikit-learn, Seaborn, Matplotlib** (Libraries used for model building and visualization)