Assessment Report

on

"Diabetes Prediction"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

1. Angelina -202401100300041
2. Aditya Shivhare - 202401100300018
3. Anish Chaudhary- 202401100300046
4. Abhishek Kumar - 202401100300007
5. Amarjeet Yadav- 202401100300033

Problem Statement: To predict whether a person is likely to have diabetes based on various medical features using machine learning techniques.

Introduction

Diabetes is one of the most prevalent chronic diseases in the world, affecting millions globally. Early prediction can lead to better treatment and management. This project aims to predict diabetes using a supervised machine learning classification model trained on a medical dataset.

Dataset Used: The Pima Indians Diabetes Database from Kaggle/UCI, which includes medical diagnostic measurements of women aged 21 years and above.

Features Include:
- Pregnancies
- Glucose
- Blood Pressure
- Skin Thickness
- Insulin
- BMI
- Diabetes Pedigree Function
- Age

Target:
- 0: Non-diabetic
- 1: Diabetic

Methodology

Approach:

1. Data Preprocessing:
   - Handle missing values (e.g., zeros in blood pressure).
   - Normalize the features.
   - Split the data into training and test sets.

2. Model Selection:
   - Logistic Regression (for baseline)
   - Random Forest Classifier
   - Support Vector Machine (SVM)

3. Model Training & Evaluation:
   - Evaluate models using accuracy, precision, recall, and F1-score.
   - Use confusion matrix and ROC curves for analysis.

4. Prediction:
   - Use the best-performing model to predict diabetes on new data.

Code

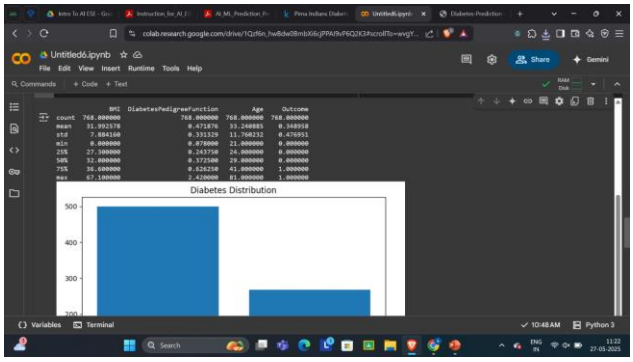
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

# Load the dataset
data = pd.read_csv('diabetes.csv')

# Basic Data Understanding
print("First 5 rows:")
print(data.head())

print("\nDataset shape:", data.shape)
print("\nBasic statistics:")
print(data.describe())

# Simple Visualization (using basic matplotlib)
plt.figure(figsize=(8, 5))
plt.hist(data['Outcome'], bins=2, rwidth=0.8)
plt.xticks([0, 1], ['No Diabetes', 'Diabetes'])
plt.title('Diabetes Distribution')
plt.show()

# Data Preprocessing
```

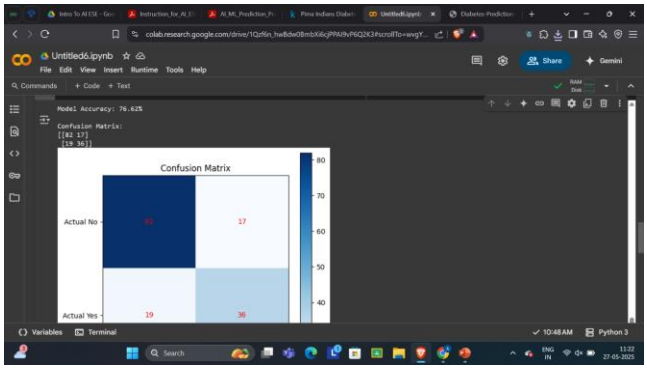
```
# Replace 0 values with median (simple handling of missing values)
columns_to_fix = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']
for col in columns_to_fix:
    data[col] = data[col].replace(0, data[col].median())

# Prepare features (X) and target (y)
X = data.drop('Outcome', axis=1)
y = data['Outcome']

# Split data (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Scale the data (important for many ML algorithms)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Simple Logistic Regression Model (easiest to understand)
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("\nModel Accuracy: {:.2f}%".format(accuracy * 100))
```

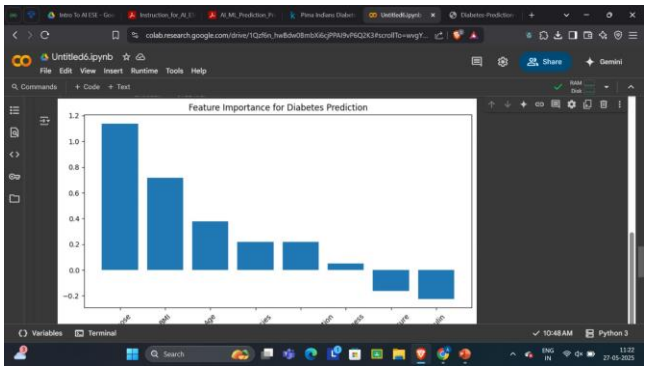
```
# Confusion Matrix (basic performance metric)
cm = confusion_matrix(y_test, y_pred)
print("\nConfusion Matrix:")
print(cm)

# Simple visualization of confusion matrix
plt.figure(figsize=(6, 6))
plt.imshow(cm, cmap='Blues')
plt.title('Confusion Matrix')
plt.colorbar()
plt.xticks([0, 1], ['Predicted No', 'Predicted Yes'])
plt.yticks([0, 1], ['Actual No', 'Actual Yes'])

# Add numbers to the plot
for i in range(2):
    for j in range(2):
        plt.text(j, i, cm[i, j], ha='center', va='center', color='red')

plt.show()

# Feature Importance (understand which factors matter most)
importance = pd.DataFrame({
    'Feature': X.columns,
    'Importance': model.coef_[0]
}).sort_values('Importance', ascending=False)

print("\nFeature Importance:")
print(importance)
```

```
# Simple bar plot of feature importance
plt.figure(figsize=(10, 5))
plt.bar(importance['Feature'], importance['Importance'])
plt.title('Feature Importance for Diabetes Prediction')
plt.xticks(rotation=45)
plt.show()
```

Output/Result







References/Credits

- Dataset: Pima Indians Diabetes Dataset - https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database