Diabetes Prediction Model

1. Introduction

This project details the development and evaluation of a classification model designed to detect the presence of diabetes in patients based on diagnostic measurements. The model is trained on the PIMA Indians Diabetes Dataset, a widely-used benchmark dataset. The goal is to build a reliable predictive system that can classify individuals as either diabetic or non-diabetic using key health indicators. A Support Vector Machine (SVM) classifier with a linear kernel was selected for this task due to its effectiveness in high-dimensional space and maximizing the margin between classes.

2. Methodology

The project was executed in a Google Colaboratory environment, utilizing Python libraries such as numpy, pandas, and scikit-learn.

2.1. Data Collection and Analysis

The **PIMA Indians Diabetes Dataset** was loaded, containing several diagnostic predictors and a binary outcome variable.

- **Dataset Size:** 768 rows and 9 columns.
- **Target Variable (**Outcome**):**

  0: Non-Diabetic (e.g., 500 instances)

  1: Diabetic (e.g., 268 instances)

- **Features (X) include:** Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, and Age.
- **Data Separation:** The features (X) were separated from the target variable (Y, the 'Outcome').

2.2. Data Standardization

Standardization is a crucial preprocessing step for the SVM algorithm, which is sensitive to the magnitude of features. The features were standardized using the StandardScaler from sklearn. This process transforms the data such that it has a mean of 0 and a standard deviation of 1.

$$z = \frac{x - \mu}{\sigma}$$

where μ is the mean and σ is the standard deviation. This ensures all features contribute equally to the model training.

2.3. Train-Test Split

The standardized data was divided into training and testing sets to evaluate the model's performance on unseen data:

- **Training Data (80%):** Used to train the SVM model.
- **Test Data (20%):** Used to assess the model's accuracy.

The split was performed with test_size = 0.2 and included stratify=Y to ensure both the training and test sets maintain the original proportion of diabetic and non-diabetic cases, preventing sampling bias. A fixed random_state=2 was used for reproducibility.

3. Model Training: Support Vector Machine (SVM)

A Support Vector Machine (SVM) classifier with a **linear kernel** was chosen. The SVM algorithm seeks to find the optimal hyperplane that distinctly classifies the data points into different classes, maximizing the margin between the closest data points of different classes (known as support vectors).

The classifier was trained using the 80% training data (X_train, Y_train).

4. Results and Evaluation

The model's performance was evaluated using the **Accuracy Score**, which measures the proportion of correctly classified instances.

DataSet	Accuracy Score	Interpretation
Training Data	78.66%	The model learned the patterns in the training set well.
Test Data	77.27%	The model generalizes reasonably well to unseen data.

The closeness between the training and testing accuracy (a difference of approximately 1.39%) suggests that the model is **not significantly overfitting** the training data, indicating good generalization capability.

5. Predictive System

The final step involved creating a predictive function to classify a single new data instance.

Example Input Data (standardized and reshaped): (5, 166, 72, 19, 175, 25.8, 0.587, 51)

1. The input is converted to a NumPy array.
2. It is reshaped to indicate a single instance for prediction.
3. The data is standardized using the **same** StandardScaler fitted during preprocessing.
4. The trained classifier predicts the outcome.

Prediction: The model predicted 0 (Non-Diabetic) for the provided sample, demonstrating the pipeline's ability to process new, raw data and provide a diagnosis.

6. Conclusion and Future Work

The SVM classification model successfully achieved an accuracy of approximately **77.3%** on the test dataset for diabetes detection. This model demonstrates a robust pipeline, including essential steps like data standardization and stratified splitting, which are critical for building reliable machine learning systems.