Diabetes Prediction Project Report

By

Sripathi.V.R

1. Introduction

Diabetes is a chronic disease that affects millions of people worldwide. Early detection and management are crucial in preventing complications and improving the quality of life for those affected. This project aims to develop a machine learning model that can predict the likelihood of an individual being diabetic based on various health-related attributes. The project involves several stages, including data exploration, preprocessing, model training, evaluation, and deployment through a Streamlit application.

2. Dataset Description

2.1 Source and Content

The dataset used in this project is the "Diabetes Prediction" dataset. It consists of 100,000 records with 9 features, including both numerical and categorical variables. The target variable is diabetes, which indicates whether an individual is diabetic (1) or not (0).

2.2 Features

- gender: Categorical (Male, Female)
- age: Numerical (Years)
- hypertension: Binary (0 = No, 1 = Yes)
- heart_disease: Binary (0 = No, 1 = Yes)
- smoking_history: Categorical (e.g., never, current, formerly, No Info, ever, not current)
- bmi: Numerical (Body Mass Index)
- HbA1c_level: Numerical (Hemoglobin A1c Level)
- blood_glucose_level: Numerical (Blood Glucose Level)
- diabetes: Binary (Target variable, 0 = No, 1 = Yes)

3. Data Exploration and Preprocessing

3.1 Data Exploration

3.1.1 Initial Inspection

The dataset was first inspected to understand its structure, identify the types of variables, and check for missing values. The initial exploration revealed that the dataset is well-structured, with no missing values.

3.1.2 Summary Statistics

Summary statistics were generated to understand the distribution of numerical variables. This included measures such as mean, median, standard deviation, minimum, and maximum values. This step helped identify any potential outliers or anomalies in the data.

3.2 Data Preprocessing

Data preprocessing is a crucial step that involves transforming the raw data into a format suitable for modeling.

3.2.1 Categorical Encoding

Categorical variables, such as gender and smoking_history, were encoded into numerical values using Label Encoding. This was necessary because machine learning algorithms typically require numerical input.

3.2.2 Feature Scaling

Numerical features, such as age, bmi, HbA1c_level, and blood_glucose_level, were scaled using StandardScaler. Feature scaling was essential to ensure that all numerical features contribute equally to the model, particularly for distance-based algorithms.

3.2.3 Saving Preprocessed Data

After preprocessing, the cleaned and transformed dataset was saved as a CSV file for use in model training.

4. Feature Selection

Feature selection is the process of identifying the most important variables that contribute to the predictive power of the model.

4.1 Correlation Analysis

Correlation analysis was used to identify features that have a strong relationship with the target variable (diabetes). Features with a high correlation with the target were considered important.

4.2 Feature Importance from Models

The Random Forest classifier, which was used in this project, provides a feature importance metric that ranks features based on their contribution to the model's predictions. All features were retained in this case, as the model showed good performance with all features included.

4.3 Variance Threshold

A variance threshold method was considered to remove features with very low variance, as these features may not contribute significantly to the model. However, in this dataset, all features were deemed useful, so none were removed.

5. Model Building and Evaluation

5.1 Load and Prepare Data

The first step in the model-building process involves loading the dataset and splitting it into training and testing sets.

- **Dataset**: Diabetes_Prediction
- **Training Set Shape**: (80,000, 8)
- **Test Set Shape**: (20,000, 8)

The dataset was split into 80% training data and 20% testing data, ensuring that the model can be trained on a substantial amount of data while still being evaluated on unseen data to assess its performance.

5.2 Handle Class Imbalance - Oversampling with SMOTE

To address class imbalance, where the number of non-diabetic cases (Class 0) far exceeds the number of diabetic cases (Class 1), SMOTE (Synthetic Minority Over-sampling Technique) was used to oversample the minority class.

- **Original Training Set Size**:
  - Class 0 (Non-Diabetic): 73,200
  - Class 1 (Diabetic): 6,800
- **Resampled Training Set Size**:
  - Class 0 (Non-Diabetic): 73,200
  - Class 1 (Diabetic): 73,200

SMOTE was applied to create synthetic samples for the minority class, thereby balancing the training set.

5.3 Model Training and Hyperparameter Tuning

The Random Forest classifier was chosen for its robustness and effectiveness. Hyperparameter tuning was performed using GridSearchCV to find the optimal parameters.

- **Number of Folds**: 5
- **Number of Candidates**: 20
- **Total Fits**: 100
- **Best Parameters Found**:
  - n_estimators: 150
  - min_samples_split: 2
  - min_samples_leaf: 1
  - max_features: 'sqrt'
  - max_depth: 30
- **Training Time**: 14.58 minutes

GridSearchCV helped in finding the best combination of hyperparameters that maximized the model's performance.

5.4 Model Evaluation

The model's performance was evaluated on the test set using various metrics, including accuracy, precision, recall, F1-score, and a confusion matrix.

- **Test Set Accuracy**: 0.959
- **Classification Report**:

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.98	0.98	0.98	18,300
1	0.77	0.74	0.75	1,700
Accuracy			0.96	20,000
Macro Avg	0.87	0.86	0.87	20,000
Weighted Avg	0.96	0.96	0.96	20,000

- **Confusion Matrix**:

	Predicted 0	Predicted 1
Actual 0	17,917	383
Actual 1	437	1,263

The model performed well, especially in predicting non-diabetic cases. However, there is room for improvement in predicting diabetic cases.

5.5 Adjust Decision Threshold

To improve the model's performance, especially for the minority class (diabetic cases), the decision threshold was adjusted. This involved lowering the threshold for predicting a positive class (Class 1).

- **Optimized Test Set Accuracy**: 0.898
- **Optimized Classification Report**:

Class	Precision	Recall	F1-Score	Support
0	0.99	0.90	0.94	18,300
1	0.45	0.90	0.60	1,700
Accuracy			0.90	20,000
Macro Avg	0.72	0.90	0.77	20,000
Weighted Avg	0.94	0.90	0.91	20,000

Adjusting the decision threshold increased the recall for diabetic cases but at the cost of overall accuracy.

5.6 Trained Machine Learning Model - 'Diabetes Prediction'

The final Random Forest model was trained with the best parameters and saved for deployment.

- **Trained Machine Learning Model**: Diabetes_Prediction.pkl

The model was saved as a .pkl file, which was then used in the Streamlit application for real-time predictions.

6. Model Deployment using Streamlit

The model deployment process involves creating a user-friendly web application using Streamlit, which allows users to interact with the trained machine learning model and make predictions. Below is a detailed explanation of how to deploy the Diabetes_Prediction.pkl model using Streamlit.

6.1 Set Up the Streamlit Application

First, you need to ensure that Streamlit and the required libraries are installed.

6.2 Import Required Libraries

In the Streamlit application, import the necessary libraries to load the model, process the input data, and display the predictions.

6.3 Load the Trained Model

The trained machine learning model, saved as Diabetes_Prediction.pkl, is loaded using joblib.

6.4 Create the Input Features Form

Create a form in the Streamlit app to capture the input features that the model requires to make predictions. These features should match the features used in training the model.

6.5 Data Preprocessing

Before making a prediction, the input data needs to be scaled using the same scaler that was used during model training. This ensures that the data has the same distribution as the data the model was trained on.

6.6 Make Predictions

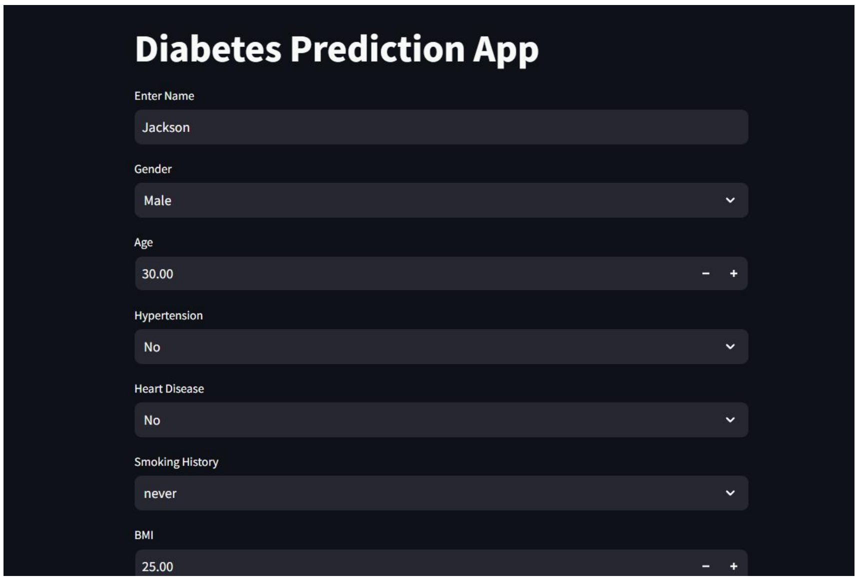
Once the input features are scaled, the model can make predictions. The prediction is displayed in the Streamlit app.

6.7 Create the Main Function

The main() function ties everything together and is called when the Streamlit app is run. It organizes the layout of the application and handles user interactions.

6.8 Run the Streamlit Application

To deploy the Streamlit app, navigate to the directory containing the Diabetes_Prediction.py file and Launch the Streamlit app in your default web browser, where users can input their data and see the predictions made by the trained model.



Diabetes Prediction App

Enter Name: Jackson

Gender: Male

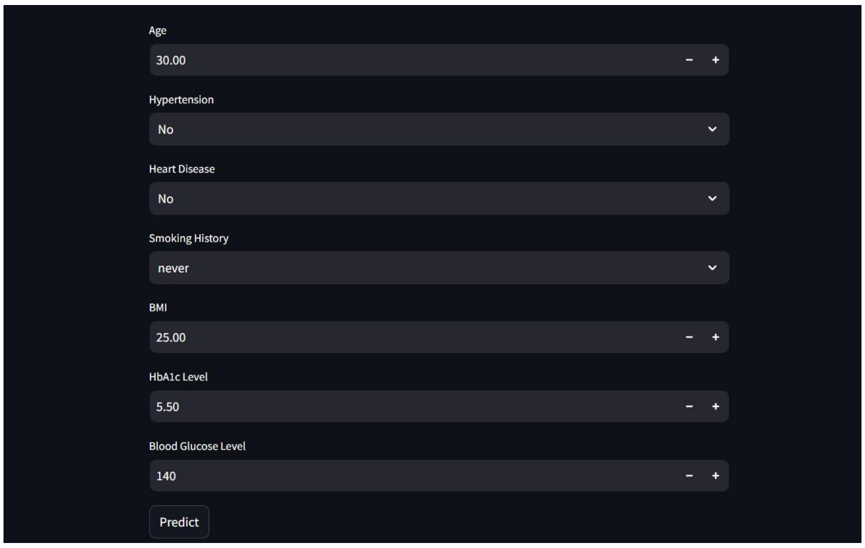
Age: 30.00

Hypertension: No

Heart Disease: No

Smoking History: never

BMI: 25.00



Age: 30.00

Hypertension: No

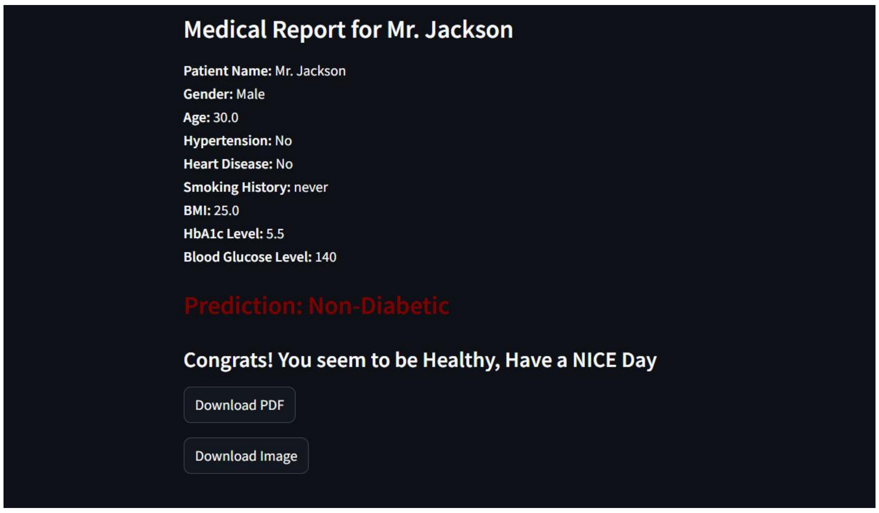
Heart Disease: No

Smoking History: never

BMI: 25.00

HbA1c Level: 5.50

Blood Glucose Level: 140

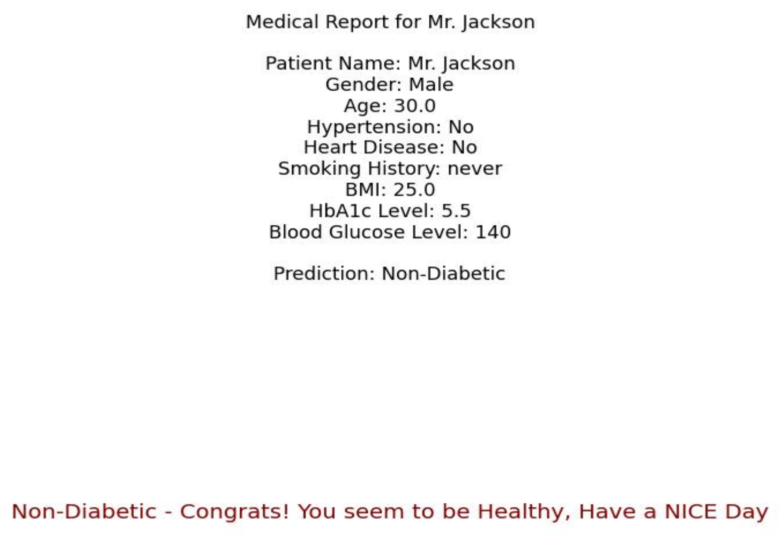


Medical Report for Mr. Jackson

Patient Name: Mr. Jackson
Gender: Male
Age: 30.0
Hypertension: No
Heart Disease: No
Smoking History: never
BMI: 25.0
HbA1c Level: 5.5
Blood Glucose Level: 140

Prediction: Non-Diabetic

Congrats! You seem to be Healthy, Have a NICE Day



```
Medical Report for Mr. Jackson

Patient Name: Mr. Jackson
Gender: Male
Age: 30.0
Hypertension: No
Heart Disease: No
Smoking History: never
BMI: 25.0
HbA1c Level: 5.5
Blood Glucose Level: 140

Prediction: Non-Diabetic

Non-Diabetic - Congrats! You seem to be Healthy, Have a NICE Day
```

7. Conclusion

This project successfully developed and deployed a machine learning model for predicting diabetes. The Random Forest classifier demonstrated high accuracy, especially in identifying non-diabetic cases. The project also involved feature selection, data preprocessing, and the creation of a user-friendly Streamlit application for easy deployment and accessibility. Future improvements could focus on enhancing the model's ability to predict diabetic cases more accurately and deploying the application on a cloud platform for broader reach.