UNIVERSITI TUNKU ABDUL RAHMAN

Wholly owned by UTAR Education Foundation
(Co. No. 578227-M)
DU012(A)

Faculty of Information and Communication Technology (FICT)

Bachelor of Computer Science (Hons)

UCCD1033 Fundamental of Data Mining & Machine Learning

June Trimester 2025

Group Assignment

Title: Diabetes Prediction Using Logistic Regression

Group No: 2

No.	Name	Student ID	Programme
1	Loh Chia Heung	2301684	CS
2	Tong Yu Shan	2301157	CS
3	Low Jia Hao	2302161	CS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

LR	Logistic Regression
SVM	Support Vector Machine
RF	Random Forest
A1C	Glycated Hemoglobin
OGTT	Oral Glucose Tolerance Test
FPG	Fasting Plasma Glucose
ML	Machine Learning
BMI	Body Mass Index

Chapter 1

Introduction

Written By: Low Jia Hao, 2302161

The worldwide burden of type 2 diabetes, a chronic metabolic disease, is rising quickly. A lot of people go undiagnosed until major problems arise. Early detection is costly and time-consuming due to the traditional screening method's reliance on lab testing and clinic visits. For risk assessment, machine learning provides a quick, data-driven substitute.

This project addresses the need for reliable, low-cost diabetes prediction tools. We build and compare three supervised classifiers using the Pima Indian diabetes dataset. By automating risk stratification, we aim to flag high-risk patients for follow-up testing and timely intervention.

The scope of this research covers binary classification of type 2 diabetes in adults using structured tabular data. We will clean and preprocess the dataset—handling missing values, scaling features, and correcting for class imbalance—before training logistic regression, random forest, and support vector machine models. The performance of each model will be measured using accuracy, precision, recall, F1 score, ROC-AUC, and confusion matrix analysis. Feature importance and interpretability techniques will also be applied to uncover key health predictors. This work does not include longitudinal prediction, causal inference, deep learning architecture, or real-world deployment.

By comparing models based on accuracy, robustness, and transparency, we aim to provide a transparent, reproducible process to support early screening and clinical decision-making in public health programs. Clinicians can use these models to allocate resources more effectively. Public health departments can integrate them into prevention strategies. Data scientists gain methodological insights into processing medical datasets and balancing interpretability and predictive power.

Problem Statement and Motivation

Diabetes is a long-term metabolic condition. It is also one of the biggest health hazards in the world. The International Diabetes Federation predicts that over 589 million adults worldwide will have diabetes by 2024. Additionally, they estimate that by 2050, there will be 852 million patients. Of the various forms of diabetes, type 2 diabetes is responsible for over 90% of cases that are reported [1][2]. Prior to their condition worsening and leading to major complications like blindness or kidney failure, 43% of patients do not receive a diagnosis [3]. This circumstance suggests that diabetes risk assessment and early detection techniques need to be improved.

The A1C, OGTT, and FPG were previously used to diagnose diabetes. These tests are expensive, time-consuming, and difficult to obtain in resource-constrained settings, despite their clinical efficacy [4][5]. Moreover, these methods may fail to detect early diabetes or the complex daily blood glucose fluctuations before severe hyperglycemia occurs [6][7].

With the growth of medical data and the improvement of computer computing power, ML has become a powerful tool for disease risk prediction and medical diagnosis [8][9]. Among the many ML methods, logistic regression, random forest, and SVM are the most common supervised learning algorithms. They are able to identify implicit patterns in large multidimensional data sets, thereby supporting the development of predictive models for disease diagnosis and prognosis [10][11]. These models can use data such as patient age, BMI, blood glucose level, and family history to predict the likelihood of diabetes even before clinical symptoms appear, thereby significantly improving early detection rates. Therefore, the original intention of this project is to use machine learning to address the growing burden of diabetes by creating robust, interpretable, and easy-to-understand prediction models. By comparing logistic regression, random forest, and support vector machine on the Pima Indian diabetes dataset, this project aims to demonstrate how data-driven algorithms can accurately stratify diabetes risk, enhance preventive healthcare, and reduce diagnostic delays, ultimately improving patient treatment outcomes and reducing healthcare costs.

Project Scope

This project investigates three supervised learning algorithms for diabetes prediction. It uses logistic regression, random forest, and SVM. The data source is the Pima Indian Diabetes Dataset. The workflow covers data acquisition, data cleaning, preprocessing, model training, evaluation, and comparison.

Data prepressing addresses missing and implausible values. Numerical features are scaled to ensure fair comparisons. Class imbalance is addressed to prevent biased results. Model training constructs three binary classifiers. Logistic regression provides interpretable probabilities. Random forest captures nonlinear patterns and ranks features. Support vector machines define clear class boundaries in high dimensions. Evaluation metrics include accuracy, precision, recall, F1 score, and ROC-AUC. A confusion matrix reveals the trade-off between positive and negative samples.

A comparative study highlights the strengths and weaknesses of each model. Interpretability and clinical relevance guide the analysis. The scope of this study is limited to binary classification of tabular data. Deep learning and real-world applications are beyond the scope of this study. The process is transparent and fully reproducible. Clinicians and researchers will gain actionable insights. The final deliverable includes documented code and results.

Project Objectives

The project objectives of this project is to develop a predictive model that classifies patients as either diabetic or non-diabetic using individual health and demographic data from the Pima Indian Diabetes Dataset. To this end, we will train three supervised machine learning algorithms: logistic regression, random forest, and support vector machine. We will compare their performance on the same binary classification task. Before building the model, we will perform thorough preprocessing to clean the dataset, handle missing and implausible values, and scale numerical features for consistency. We will also address class imbalance in the dataset to prevent biased predictions. Each trained model will be evaluated using multiple metrics: accuracy, precision, recall, F1 score, ROC-AUC, and confusion matrix analysis. These evaluation criteria will enable multi-dimensional comparisons and reveal the strengths and weaknesses of each algorithm. Furthermore, we will apply feature importance and interpretability techniques to pinpoint the most influential health parameters driving diabetes risk. This research focuses on predicting type 2 diabetes in adults using structured tabular data and does not cover longitudinal prediction, causal inference, or medical intervention simulation. By achieving these goals, we aim to provide a transparent and reproducible framework to support the integration of machine learning-based risk assessment into primary care and public health strategies. Ultimately, this work will lay a solid foundation for future research in automated disease prediction.

Chapter 2

Literature Review

Written By: Tong Yu Shan, 2301157

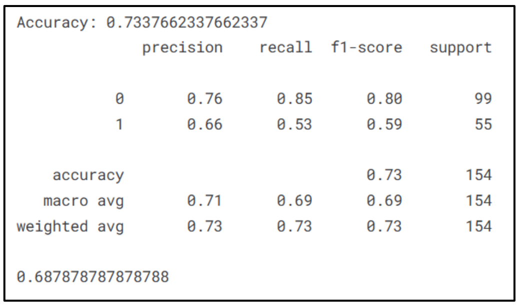
2.1 Code Review 1: KNN_Classifer vs Logistic Regression

Link: https://www.kaggle.com/code/mohanedmashaly/knn-classifier-vs-logistic-regression

In the notebook written by Mashaly [12], the author predicted diabetes based on the Pima Indians dataset by using K-Nearest Neighbors (KNN) and Logistic Regression (LR) algorithms to compare the performance between them. During preprocessing, SimpleImputer was applied to handle missing values by filling in the means, and then MinMaxScaler was used for features scaling. The author finally reported test accuracy of 0.7338 and 0.7532 respectively for KNN and Logistic Regression, with corresponding ROC-AUC scores of 0.6879 and 0.7232. These results are shown in Figure 2.1 [12] and Figure 2.2 [12]. Furthermore, the seaborn library was used to visualize relationships in the dataset, as presented in Figure 2.3 [12].

The findings indicate that Logistic Regression model performed slightly better than the KNN model. However, the analysis has several limitations. For instance, no data exploration, analysis and visualization were carried out to understand the relationships between features and outcomes. Besides, the models were trained without hyperparameters tuning, such as the choice of k in KNN or the regularization strength in Logistic Regression. As a result, the reported results may not fully reflect the models' potential performance.

In conclusion, this review highlights the importance of performing data analysis, exploration and visualization to understand datasets more deeply before training models. It also represents metrics as an essential element in model training to understand and compare the result, such as precision, recall, and F1-score. For future work to enhance performance, hyperparameter tuning is suggested to be applied to ensure that model prediction is robust and accurate, particularly in medical prediction tasks.



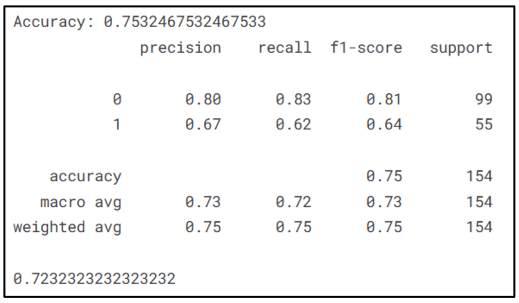
```
Accuracy: 0.7337662337662337
              precision    recall  f1-score   support

           0       0.76      0.85      0.80        99
           1       0.66      0.53      0.59        55

    accuracy                           0.73       154
   macro avg       0.71      0.69      0.69       154
weighted avg       0.73      0.73      0.73       154

0.687878787878788
```

Figure 2.1 Classification Report of the Model Trained Using KNN Classifier [12]



```
Accuracy: 0.7532467532467533
              precision    recall  f1-score   support

           0       0.80      0.83      0.81        99
           1       0.67      0.62      0.64        55

    accuracy                           0.75       154
   macro avg       0.73      0.72      0.73       154
weighted avg       0.75      0.75      0.75       154

0.7232323232323232
```

Figure 2.2 Classification Report of the Model Trained Using Logistic Regression [12]

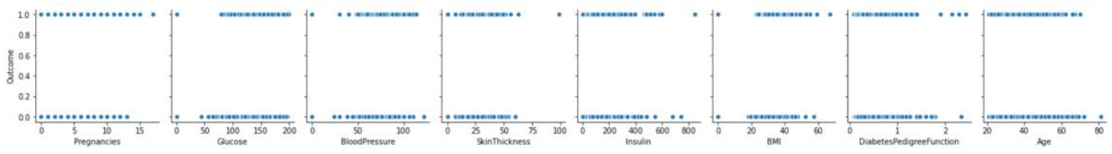


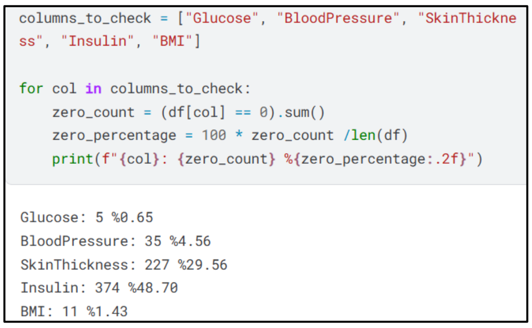
Figure 2.3 Data Visualization with the Seaborn Library [12]

2.2 Code Review 2: Pima Indians Diabetes Database with Random Forest

Link: https://www.kaggle.com/code/alperaydoan/pima-indians-diabetes-database-with-randomforest

In Kaggle, Alper Aydoğan has shared his work on predicting diabetes by using Random Forest (RF) algorithms. The value in each attribute is checked in preprocessing stage, there is no null in all attributes in this dataset fortunately, however, many features have unreasonable value. As the author said, it is possible for pregnancies to have value of zero, but others such as glucose, blood pressure, skin thickness, insulin, diabetes predigree function and age, are impossible to have value of zero in norma. [13] To solve this problem, median values are replaced in those records instead of discarding them directly. Besides, StandardScaler is utilized for feature scaling and GridSearchCV is also used to find the best combination of parameters to improve performance. The author reported that test accuracy of 0.7273, which is comparable to the KNN and Logistic Regression model reviewed previously.

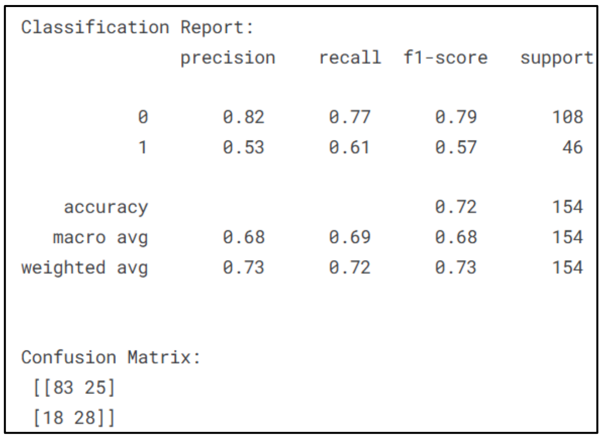
This report emphasizes using all features provided in dataset to train models, thus preprocessing is a crucial step to clean and transform data to enhance performance. To avoid removing some important features intentionally, the author has suggested transforming them by replacing them with medians, but it may cause misleading to wrong outcomes. It is because there are nearly half of the insulin records (48.70%) that have value of zero which are regarded as missing value as shown in Figure 2.4. While changing them with means, consequently, the model trained may be less effective in predicting this outcome correctly. The result has also revealed that this consequence is because it has lower precision, recall and F1-score as shown in Figure 2.5.


```
columns_to_check = ["Glucose", "BloodPressure", "SkinThickne
ss", "Insulin", "BMI"]

for col in columns_to_check:
    zero_count = (df[col] == 0).sum()
    zero_percentage = 100 * zero_count /len(df)
    print(f"{col}: {zero_count} %{zero_percentage:.2f}")
```

```
Glucose: 5 %0.65
BloodPressure: 35 %4.56
SkinThickness: 227 %29.56
Insulin: 374 %48.70
BMI: 11 %1.43
```

Figure 2.4 Data Exploration for Invalid Values [13]



```
Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.77      0.79       108
           1       0.53      0.61      0.57        46

    accuracy                           0.72       154
   macro avg       0.68      0.69      0.68       154
weighted avg       0.73      0.72      0.73       154


Confusion Matrix:
 [[83 25]
 [18 28]]
```

Figure 2.5 Classification Report of the Model Trained Using Random Forest Classifier [13]

Furthermore, another limitation is the limited data visualization of only one feature, named 'Predigree Function Distribution', which is represented in colorful charts in Figure 2.6. This indicates we have less understanding of other features, which will lead to poor data preprocessing and affect performance of model trained.

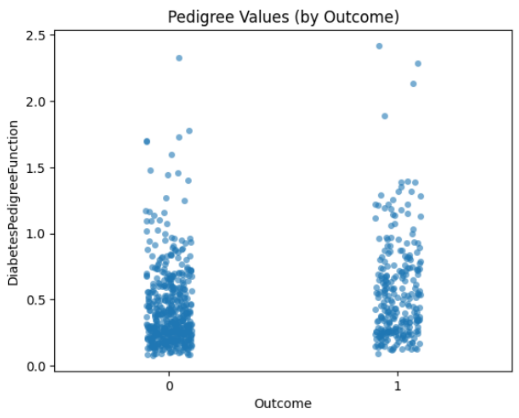
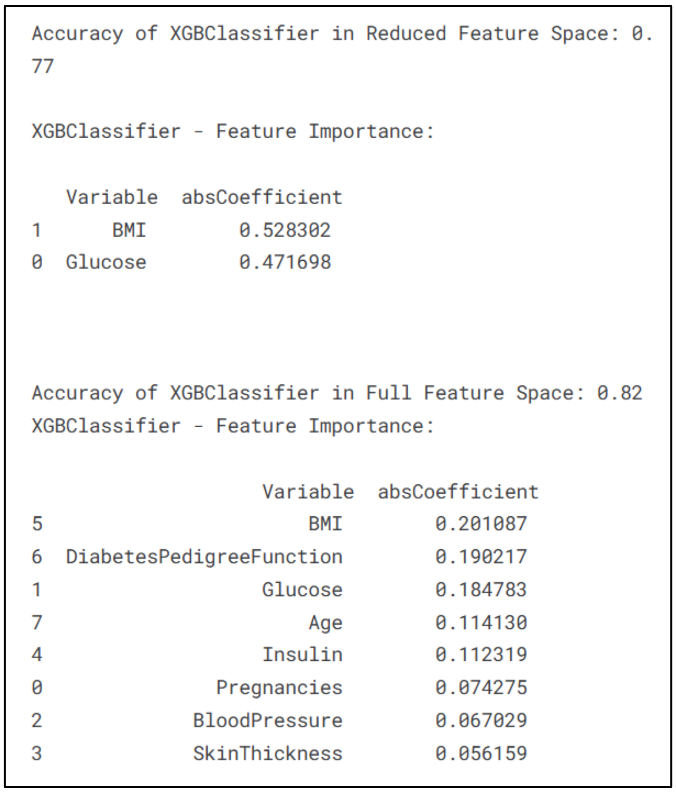


Figure 2.6 Data Visualization of 'Predigree Function Distribution'[13]

At the same time, Paul Mooney has compared the performance of using reduced and full features in her notebook. As Figure 2.7 shown below, by using same classifier, the author obtains accuracy of 0.77 and 0.82 respectively for using reduced and full features to train models. [14] Higher accuracy is obtained when using full features, we believe that we can minimize the chance of discarding important features intentionally by using most reasonable records if possible.



```
Accuracy of XGBClassifier in Reduced Feature Space: 0.
77

XGBClassifier - Feature Importance:

   Variable  absCoefficient
1      BMI        0.528302
0  Glucose        0.471698



Accuracy of XGBClassifier in Full Feature Space: 0.82
XGBClassifier - Feature Importance:

                    Variable  absCoefficient
5                        BMI        0.201087
6   DiabetesPedigreeFunction        0.190217
1                    Glucose        0.184783
7                        Age        0.114130
4                    Insulin        0.112319
0                Pregnancies        0.074275
2              BloodPressure        0.067029
3              SkinThickness        0.056159
```

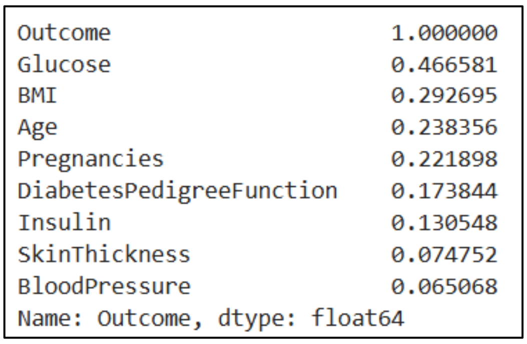
Figure 2.7 Comparison of Using Reduced or Full Features to Train Model [14]

In a nutshell, other than using preprocessing tools or parameter tuning tools to enhance performance and accuracy, we can conclude that this review highlights the importance of careful data preprocessing, especially when dealing with missing values in datasets and filtering crucial features. Additionally, sufficient data visualization also plays a key role in having better understanding of data and improving the preprocessing steps, which finally enhances the overall performance of the model trained.

2.3 Code Review 3: Pima Indians Diabetes (Simple Logistic Regression)

Link: https://www.kaggle.com/code/mshirlaw/pima-indians-diabetes-simple-logistic-regression

In the notebook done by Mshirlaw, Logistic Regression (LR) is used to predict diabetes. During data preprocessing, correlations between features are checked first to identify which one is more strongly related to the target outcome as shown in Figure 2.8. The author then drops the record having null value in the higher correlated features to minimize the difference between dataset and real situation. After that, the two higher correlated features are only used to train model after being scaled by StandardScalar. Finally, test accuracy of 0.8013 is obtained in this notebook, which has better accuracy than in the code reviewed previously.



```
Outcome                     1.000000
Glucose                     0.466581
BMI                         0.292695
Age                         0.238356
Pregnancies                 0.221898
DiabetesPedigreeFunction    0.173844
Insulin                     0.130548
SkinThickness               0.074752
BloodPressure               0.065068
Name: Outcome, dtype: float64
```

Figure 2.8 Correlations between features and outcomes [15]

By comparing the previous model trained by the same algorithm, which is Linear Regression, they both didn’t use any hyperparameter tool, but the current model got higher accuracy. It may be due to the different ways to handle missing values and choose features to train; thus, this review highlights the importance of analyzing the dataset wisely and selecting important features, which can literally improve the accuracy and performance.

Despite it having better accuracy, there are some limitations found when using two features out of eight of them. It may oversimplify the relationship between diabetes and features since diseases like diabetes are related to multiple factors, so it is not suitable for applying less features to explain complex relationships. Furthermore, the more predictive features in combination are not considered while using correlation

level to choose features. To avoid losing any vital information, we should use multiple approaches to analyze data and perform wise decision making.
Thus, the result given by the previous model may be more reliable than the current model even if it has lower accuracy, since it has taken account of every feature provided in dataset.

Table 2.1: Comparison of Code Reviews

Code Review	**1**	**2**	**3**
Data set used	Pima Indians Diabetes Database [16]		
Handling missing values	Replacing by means using SimpleImputer	Replacing by medians	Dropping records having null value (Glucose, BMI)
Features used to train model	All features (Glucose, BMI, Age, Pregnancies, DiabetesPredigree Function, Insulin, SkinThickness, BloodPressure)	All features (Glucose, BMI, Age, Pregnancies, DiabetesPredigree Function, Insulin, SkinThickness, BloodPressure)	Two highest correlated features to outcomes (Glucose, BMI)
Preprocessing tool	MinMaxScaler	StandardScaler	StandardScaler
Model used	KNN Classifier	Random Forest Classifier	Logistic Regression
	Logistic Regression		
Hyperparameter tuning tool	-	GridSearchCV	-
Accuracy on test set	0.7338	0.7273	0.8013
	0.7532		
Roc_auc_score	0.6879	0.6808	0.7508
	0.7232		

Chapter 3
Machine Learning Model Development Process

Written By: Loh Chia Heung, 2301684

3.1 Machine Learning Model Development Process Overview

Figure 3.1 presents the overall development process carried out in this project. The process begins with acquiring the dataset, followed by describing and understanding its characteristics through data exploration and visualization. Then, data preprocessing steps are applied to prepare the data before splitting it into training and testing sets. Next, three machine learning models are selected and trained using K-Fold cross-validation combined with hyperparameter tuning to ensure fair and optimized performance. The models are then subsequently evaluated, and the best-performing model is chosen. Finally, the selected model is analyzed and interpreted in the result and analysis stage. Further details of each stage are elaborated in the subsequent sections of this chapter.

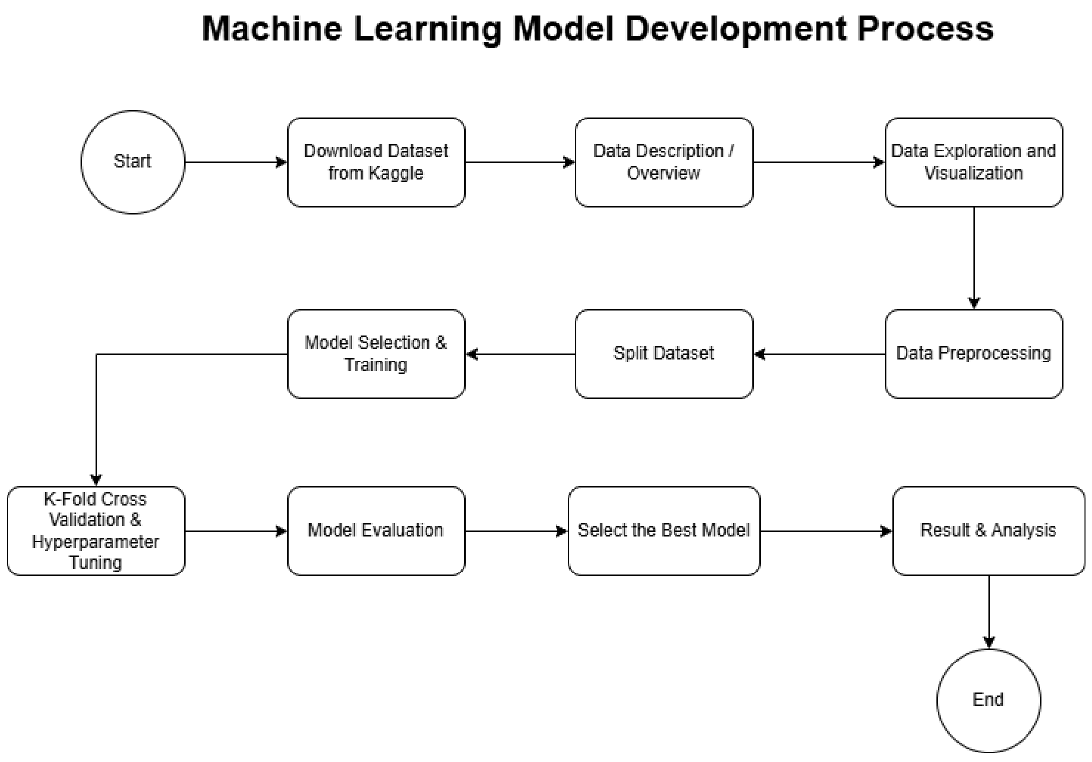

Machine Learning Model Development Process
Start
Download Dataset from Kaggle
Data Description / Overview
Data Exploration and Visualization
Data Preprocessing
Split Dataset
Model Selection & Training
K-Fold Cross Validation & Hyperparameter Tuning
Model Evaluation
Select the Best Model
Result & Analysis
End

Figure 3.1 Machine Learning Model Development Process Overview

3.2 Data Preparation and Understanding

3.2.1 Dataset Acquisition

The dataset used in this project is the **Pima Indians Diabetes Dataset [16].** It is obtained from the Kaggle repository. It consists of **768 records** of female patients of Pima Indian heritage, with **8 numerical medical attributes** and **1 binary target variable (Outcome)**, indicating that whether the patient is diagnosed with diabetes (Outcome = 1) or not (Outcome = 0).

The dataset was selected because it is widely used in machine learning research for benchmarking classification algorithms in the medical domain. It provides a balanced challenge with moderately imbalanced classes and includes features that are relevant for diabetes risk assessment, such as glucose level, blood pressure, body mass index (BMI), insulin, and skin thickness.

The dataset was downloaded in **CSV format (diabetes.csv)** and analyzed using **Python** within a **Jupyter Notebook environment**, which provides an interactive platform for data preprocessing, visualization, and model development. Since it is a public resource, the dataset also enhances reproducibility, allowing other researchers to obtain the same dataset and extend this work by experimenting with different methods or improvements.

3.2.2 Data Description and Understanding

This section provides an overview of the data used in this project. The Pima Indians Diabetes dataset contains 768 records and 9 variables in total, which consists of **8 input features (x)** and **1 target variables (y)**. All features are numerical and represent clinical measurements that are commonly used in diabetes risk assessment.

The 9 variables are described in Table 3.1.

No.	Feature	Description	Datatype	Missing Values	Duplicated Record
1.	Pregnancies	Number of times the patient has been pregnant [16].	Integer	No	No
2.	Glucose	Plasma glucose concentration after a 2-hour oral glucose tolerance test (mg/dL) [16].	Integer	No	No
3.	Blood Pressure	Diastolic blood pressure (mm Hg) [16].	Integer	No	No
4.	Skin Thickness	Triceps skinfold thickness (mm) [16].	Integer	No	No
5.	Insulin	2-hour serum insulin (µU/mL) [16].	Integer	No	No
6.	Body Mass Index (BMI)	Body mass index (weight in kg / (height in m^2)) [16].	Float	No	No
7.	Diabetes Pedigree Function	A function that scores the likelihood of diabetes based on family history	Float	No	No
8.	Age	Patient's age (years)	Integer	No	No
9.	Outcome	Class label (1 = diabetic, 0 = non-diabetic)	Binary	No	No

Table 3.1 Summary of 9 Variables Information

At this initial stage, the dataset was inspected to verify its basic structure and quality. The following summary was obtained:

- Total records: 768
- Attributes/features: 9 (8 features + 1 outcome)
- Data types: All numeric (7 integers, 2 floats)
- Missing values: None detected
- Duplicated records: None found

This descriptive understanding provides an overview for subsequent preprocessing steps, where additional data handling procedures will be applied to enhance the quality of inputs for model training.

3.2.3 Data Exploration and Visualization

After describing the dataset, exploration data analysis (EDA) was performed to further understand the distribution and relationships of the features. Visualization techniques were applied to identify potential data quality issues, detect imbalances, and reveal insights about the feature behavior. The following key aspects are examined:

1. Feature Distributions

- Histograms were generated for each numerical feature to observe their spread and distribution.
- As shown in Figure 3.2, some features such as Insulin and Skin Thickness are highly skewed towards lower values, in the other hands, Age and BMI demonstrate more normal-like distributions.
- Recognizing skewness is important because it can affect algorithms that are sensitive to feature scaling.

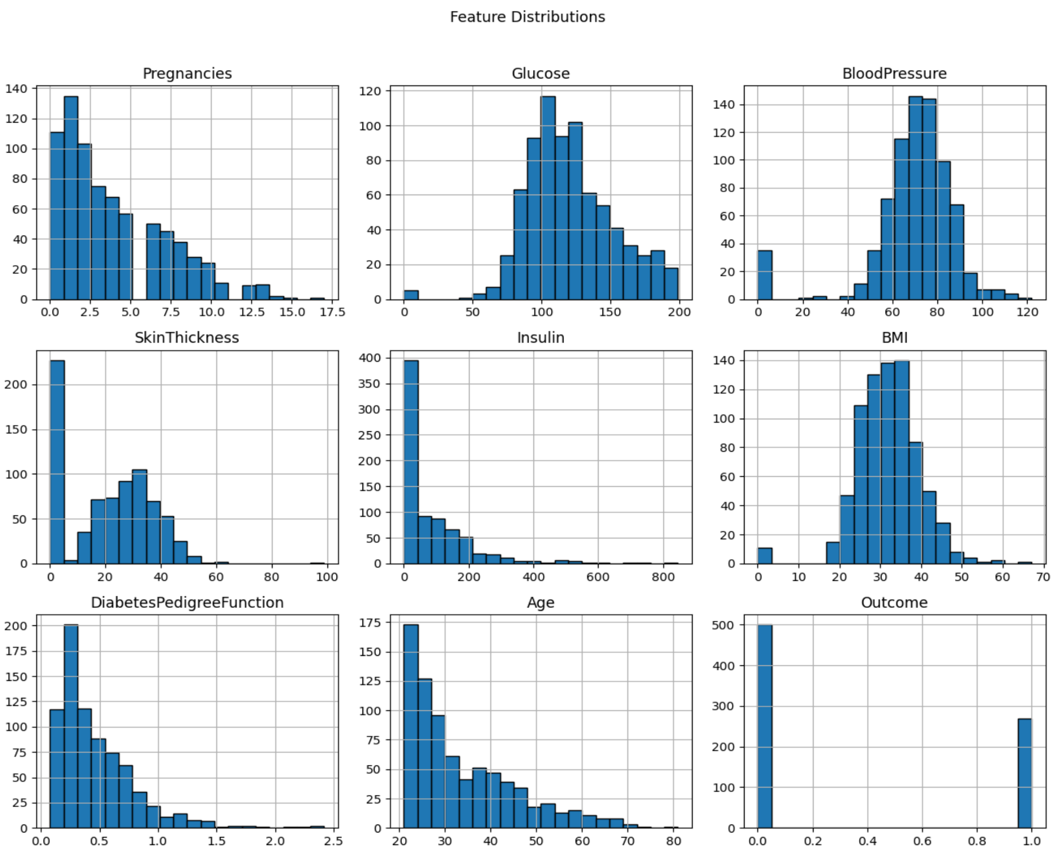


Figure 3.2 Feature Distributions Histogram

2. Correlation Analysis

- A correlation heatmap was plotted to evaluate the strength of relationships between features.
- As presented in Figure 3.3, Glucose shows a strong positive correlation with the target variable (Outcome), while BMI and Age also demonstrate moderate positive relationships. In contrast, other features, such as Skin Thickness and Insulin, appear to have weaker direct correlations.

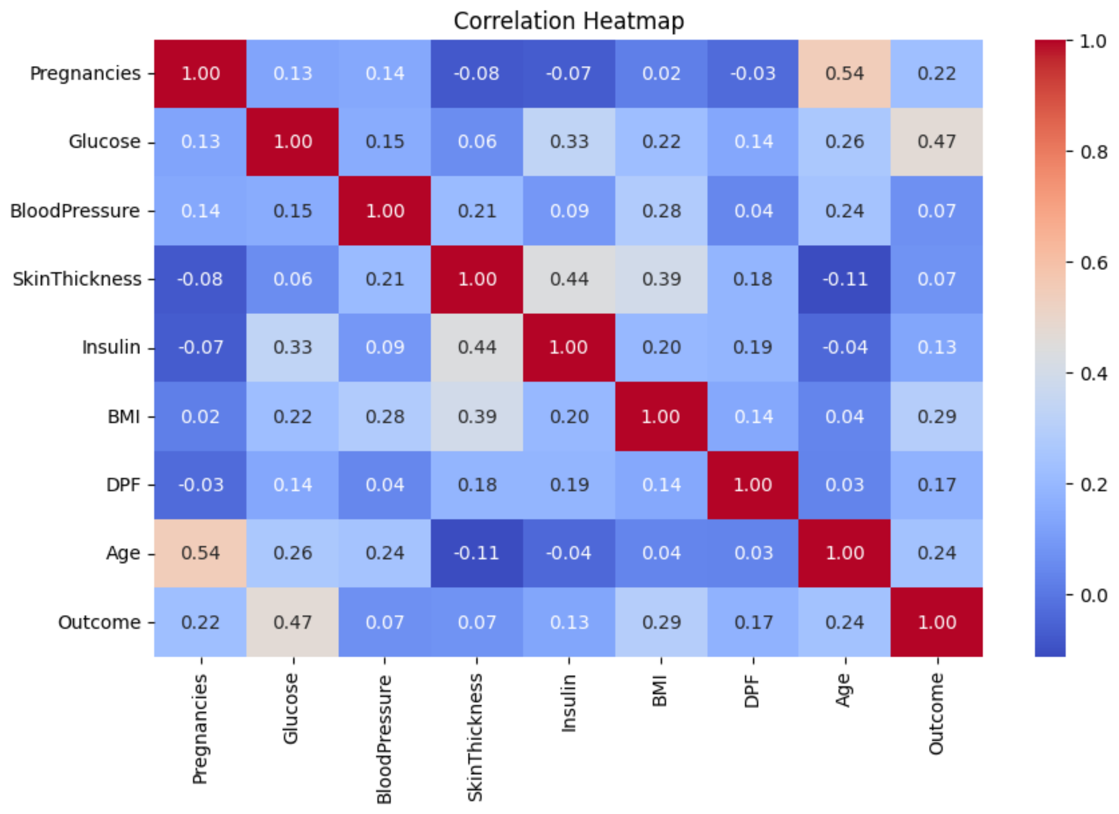


Figure 3.3 Correlation Heatmap

3. Class Distribution

- The distribution of the target variable (Outcome) was analysed to check for balance between diabetic and non-diabetic cases.
- As illustrated in Figure 3.4, the dataset contains 500 non-diabetic records (Outcome = 0) and 268 diabetic records (Outcome = 1). This reveals a moderately imbalanced dataset, with non-diabetic cases dominating for approximately 65% of the data.
- This imbalance suggests that accuracy alone may not provide a reliable measure of model performance, and additional metrics such as the ROC–AUC, Precision–Recall, are needed for proper evaluation.

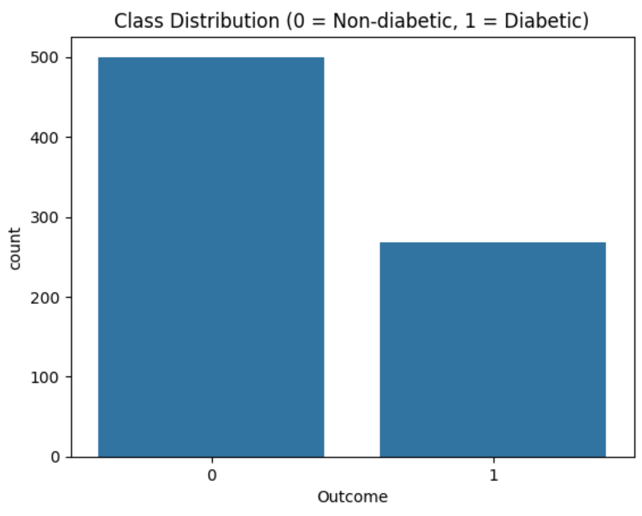


Figure 3.4 Class Distribution

4. Additional Visual Inspection

- In addition to the above, boxplots and scatterplots were used to check for outliers, particularly in features such as Insulin and Skin Thickness, where extreme values were observed.
- Zero-value checks were also carried out for clinical features such as Glucose, Blood Pressure, BMI, and Insulin. The results are showed in Figure 3.5. Although no missing values were reported in the dataset, these zero values may require further handling during the data preprocessing stage.

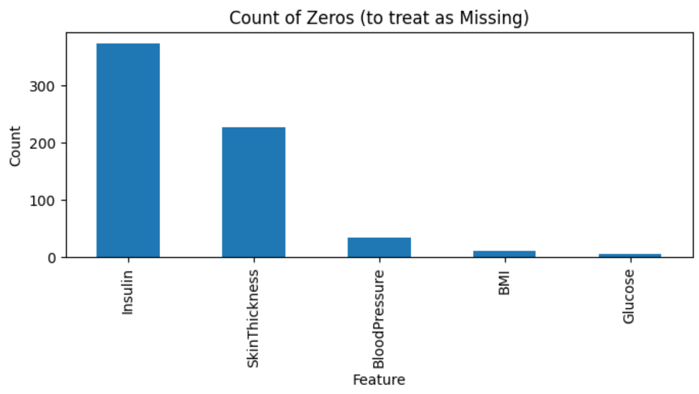


Figure 3.5 Count of Zeros Value

In summary, the exploratory and visualization stage highlighted the key characteristics of the dataset: varying feature distributions, meaningful correlations with the target variable, moderate class imbalance, and the presence of potential outliers and zero-value anomalies. These findings guided the preprocessing strategies and informed the model evaluation approaches described in the subsequent sections.

3.2.4 Data Preprocessing

Upon analyzing the data, data preprocessing was an essential step to ensure that the dataset was suitable for model training and evaluation. The raw dataset was examined for inconsistencies and adjusted through a sequence of cleaning and transformation procedures.

Although no missing values were reported, several clinical features (Glucose, Blood Pressure, Skin Thickness, Insulin, and BMI) contained **zero entries**. These values are biologically implausible, therefore it may affect the analysis. The preprocessing initially focused on Glucose and BMI, as they showed the strongest correlation with the outcome variable and are clinically important indicators in diabetes. In this early stage, zeros in these two features were replaced with NaN and the corresponding rows were dropped entirely from the dataset. While this simpler method, it can effectively remove invalid entries, and it reduced the dataset size and risked losing potentially valuable information. Recognizing this limitation, the approach was subsequently extended to all five features with zero values replaced by NaN, preparing the dataset for next preprocessing step.

The dataset was then split into **training (80%)** and **testing (20%)** subsets using a stratified split to preserve the class distribution. Splitting before imputation was crucial to avoid data leakage, as computing median values using the full dataset would allow the test data to indirectly influence the training process. A SimpleImputer with the median strategy was fitted on the training set to replace the missing values. The median was chosen because it is more robust to outliers and skewed feature distributions. The same fitted imputer was subsequently applied to the testing set, ensuring that no information leaked from the test data into the training process.

For algorithms sensitive to feature magnitude, such as Logistic Regression and Support Vector Machine, standardization was performed using StandardScaler. The scaler was fitted only on the training data and applied to the test data to maintain the data integrity. Tree-based models such as Random Forest were excluded from scaling, as they are not influenced by differences in feature scale.

To ensure the consistency and reproducibility, all preprocessing steps, including imputation and scaling were embedded within scikit-learn Pipelines. This integration guaranteed that transformations were applied correctly during cross-validation, with parameters derived solely from the training folds, before being applied to the held-out test set.

In summary, the preprocessing stage cleaned and standardized the dataset by addressing implausible zero values, imputing missing-like entries, applying scaling where appropriate, and splitting the data into training and testing subsets. These steps established a reliable foundation for the subsequent model training and evaluation stages.

3.3 Model Selection

To evaluate the performance of different supervised learning algorithms on the Pima Indians Diabetes dataset, 3 classification models were selected. They are **Logistic Regression (LR), Support Vector Machine (SVM)**, and **Random Forest (RF).** These models were chosen because they represent diverse learning paradigms, which are linear, kernel-based, and ensemble tree-based approaches, which allows for a fair comparison of their strengths and weaknesses in medical classification tasks.

3.3.1 Logistic Regression (LR)

Logistic Regression was selected as a baseline linear classifier. It is widely used in medical applications due to its interpretability and ability to estimate probabilities for binary outcomes. To handle class imbalance, the parameter class_weight="balanced" was applied, and the maximum number of iterations was increased (max_iter=1000) to ensure convergence. The model was also initialized with a fixed random_state=42 to ensure reproducibility.

3.3.2 Support Vector Machine (SVM)

The Support Vector Machine with a radial basis function (RBF) kernel was chosen to capture the potential non-linear relationships between features and the target variable. The parameter probability=True was enabled to allow probability estimates for evaluation metrics such as ROC–AUC. The model was also initialized with a fixed random_state=42 to ensure reproducibility.

3.3.3 Random Forest (RF)

Random Forest was selected as a representative ensemble method that can capture complex feature interactions without requiring feature scaling. It is robust to noise, less prone to overfitting compared to individual decision trees, and provides useful feature importance measures. To handle the class imbalance, the parameter class_weight="balanced" was included, along with random_state=42 for ensuring the reproducibility.

All models were trained using the scikit-learn Pipeline framework, which embedded preprocessing steps (imputation and scaling where applicable) to prevent data leakage. Training was conducted on the stratified training split, while evaluation was reserved for the test set.

To ensure fairness in comparison, the same training protocol was applied to all models. Each model was subjected to **5-fold stratified cross-validation with hyperparameter tuning.** This allowed the models to be optimized under consistent conditions before proceeding to final evaluation on unseen data.

3.4 Hyperparameter Tuning

After the initial baseline evaluation, hyperparameter tuning was performed to optimize model performance and ensure that each classifier was fairly compared under consistent conditions. Instead of relying on default parameters, a systematic search was conducted to identify the best configurations for each model.

3.4.1 Cross-Validation Setup

A **GridSearchCV** approach was employed with **5-fold stratified cross-validation (CV)**. Stratification was applied to maintain the proportion of diabetic and non-diabetic cases across folds, reducing bias from class imbalance. For reproducibility, the CV procedure was executed using a fixed random_state=42.

3.4.2 Scoring Metric

The primary scoring metric used during tuning was **ROC–AUC (Receiver Operating Characteristic – Area Under Curve)**. This metric was selected because it provides a more reliable assessment than accuracy in imbalanced datasets. In addition to ROC–AUC, secondary metrics such as precision, recall, and F1-score were also considered during evaluation.

3.4.3 Parameter Grids

The following hyperparameter grids were defined for each model:

- **Logistic Regression (LR)**
    - C: {0.01, 0.1, 1, 10, 100}
    - penalty: {l1, l2}
    - solver: {liblinear, saga}
- **Support Vector Machine (SVM, RBF kernel)**
    - C: {0.1, 1, 10, 100}
    - kernel: {linear, rbf}
    - gamma: {scale, auto}
- **Random Forest (RF)**
    - n_estimators: {100, 300, 500}
    - max_depth: {None, 5, 10, 15}
    - min_samples_split: {2, 5, 10}
    - min_samples_leaf: {1, 2, 4}
    - max_features: {sqrt, log2, None}

3.4.4 Best Estimator Selection

For each algorithm, GridSearchCV identified the **best estimator** based on the highest mean ROC–AUC score across the training folds. These best-tuned models were then retrained on the entire training set and evaluated on the unseen test set, as discussed in Section 3.5.

In summary, hyperparameter tuning ensured that each model was evaluated under optimized conditions. This process provided a fair comparison across classifiers and helped reduce the risk of underfitting or overfitting.

3.5 Final Training and Model Evaluation

After hyperparameter tuning, the best estimators identified by **GridSearchCV** were selected for final evaluation. Each model was retrained on the entire **training set (80%)** using the optimal parameter configurations obtained during cross-validation. The models were then evaluated on the **test set (20%)**, which had not been used in training or tuning, to provide an unbiased assessment of generalization performance.

3.5.1 Evaluation Metrics

To ensure a comprehensive evaluation, multiple performance metrics were considered:

- **Accuracy**: It shows the overall proportion of correct predictions. It is simple to understand, but it can be misleading when one class dominates the dataset.
- **Precision**: Out of all the cases predicted as "diabetic," precision shows that how many were actually diabetic. High precision means fewer false alarms.
- **Recall (Sensitivity)**: Out of all actual diabetic cases, recall shows that how many were correctly detected. High recall is important in healthcare, as missing a diabetic case can be critical.
- **F1-Score**: A balance between precision and recall. It is useful when both false positives and false negatives carry significant consequences.
- **Confusion Matrix**: A table that breaks down correct and incorrect predictions into true positives, false positives, true negatives, and false negatives. It summarizes the types of errors each model makes.
- **ROC–AUC (Receiver Operating Characteristic – Area Under Curve)**: It indicates how well the model distinguishes between diabetic and non-diabetic cases. A value closer to 1 represents stronger performance.
- **Precision–Recall Curve and Average Precision (AP)**: It focuses on how well the model performs on the minority class (diabetic patients), making it more informative in imbalanced datasets.

3.5.2 Final Training Protocol Summary

1. Retrain each best-tuned model on the training set.
2. Test the models on the unseen test set.
3. Compare their results using the evaluation metrics above.
4. Identify which model performs the best overall and interpret the results by considering the trade-offs between precision, recall, and other measures

Chapter 4

Results, Analysis, Findings & Conclusions

4.1 Logistic Regression (LR) Results

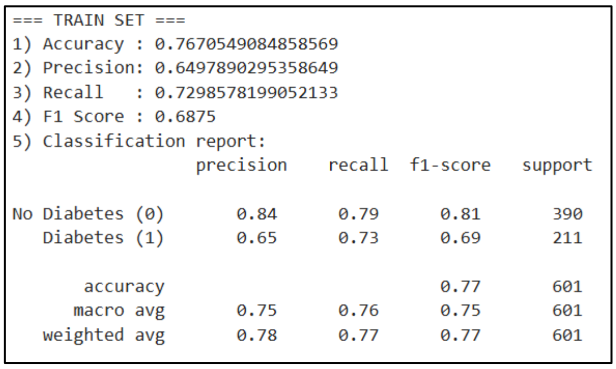
Written By: Tong Yu Shan, 2301157

After model training, we have shown the results in terms of accuracy, precision, recall, F1 score, classification report, confusion matrix, average precision, precision-recall curve, ROC-AUC score and ROC curve. They are common and great metric to describe the result, thus each of them will be compared to the results obtained in each stage, then they are elaborated and analyzed in the following subtopics.

4.1.1 Result on Baseline Training (LR)

After Trained a Logistic Regression model with StandardScaler and class_weight="balanced" to handle class imbalance. On the train set, the model achieved Accuracy = 76.7%, Precision = 65.0%, Recall = 73.0%, F1 Score = 68.6%, and ROC-AUC = 0.841, as shown in Figure 4.1.1. For overall, this model has moderate performance on the training set for diabetes classification but better at identifying non-diabetes cases accurately.

By comparing the test set, the model achieved Accuracy = 76.2%, Precision = 68.9%, Recall = 58.5%, F1 Score = 63.3%, and ROC-AUC = 0.851, as shown in Figure 4.1.2.

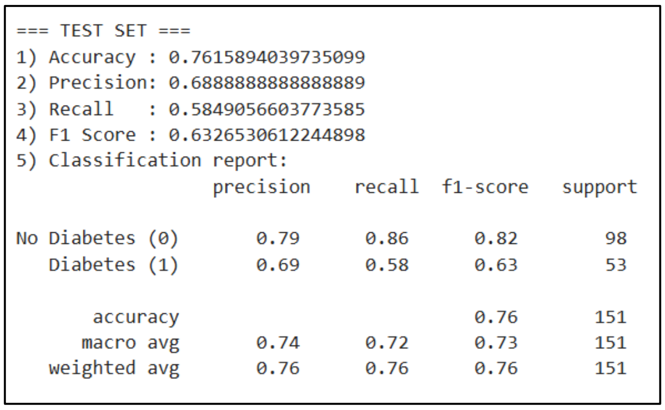


```
=== TRAIN SET ===
1) Accuracy : 0.7670549084858569
2) Precision: 0.6497890295358649
3) Recall   : 0.7298578199052133
4) F1 Score : 0.6875
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.84      0.79      0.81       390
   Diabetes (1)       0.65      0.73      0.69       211

       accuracy                           0.77       601
      macro avg       0.75      0.76      0.75       601
   weighted avg       0.78      0.77      0.77       601
```

Figure 4.1.1 Result of Baseline Training on Train Set (LR)



```
=== TEST SET ===
1) Accuracy : 0.7615894039735099
2) Precision: 0.6888888888888889
3) Recall   : 0.5849056603773585
4) F1 Score : 0.6326530612244898
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.79      0.86      0.82        98
   Diabetes (1)       0.69      0.58      0.63        53

       accuracy                           0.76       151
      macro avg       0.74      0.72      0.73       151
   weighted avg       0.76      0.76      0.76       151
```

Figure 4.1.2 Result of Baseline Training on Test Set (LR)

As Table 4.1.1 shown below, both train and test set have similar accuracy which proves that the model is constructed well without serious overfitting. Besides, they have consistent high scores in ROC-AUC which are greater than 0.84 (>0.84), as shown in Figure 4.1.3. For the increasing of precision in test set, it represents the model is more accurate to provide positive diabetes predictions. However, the decline of Recall and

F1 Score indicates that the model cannot give correct diagnosis among diabetic patients, it will be a critical problem in healthcare system.

Table 4.1.1 Comparison of Result of Train Set and Test Set in Baseline Training (LR)

Metrics	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Train Set	76.7%	65.0%	73.0%	68.8%	0.841
Test Set	76.2%	68.9%	58.5%	63.3%	0.851

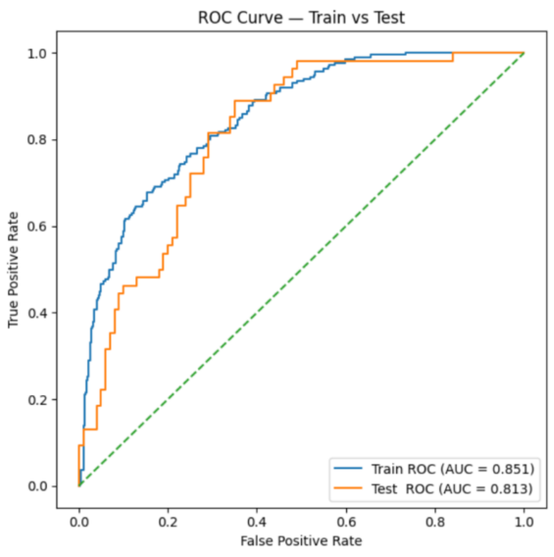
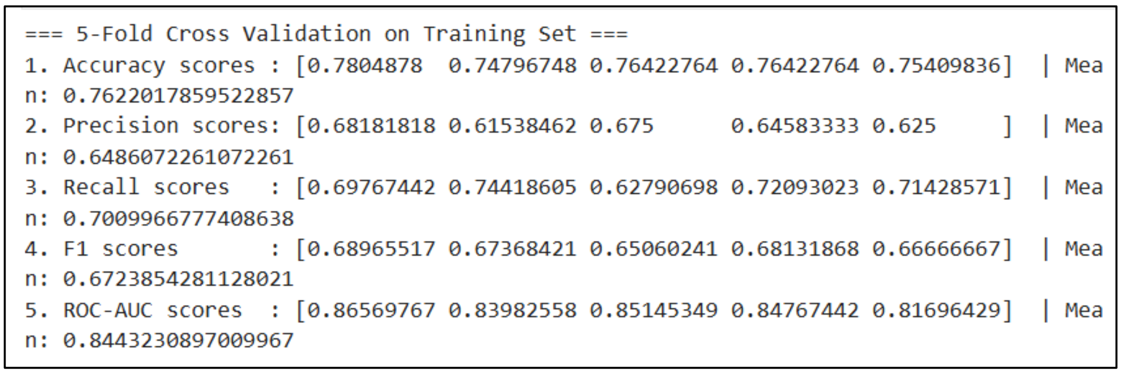


Figure 4.1.3 ROC Curve of Train and Test Set in Baseline Training (LR)

4.1.2 Result on Cross-Validation (LR)

To enhance the performance of metric and model, 5-fold stratified cross-validation is performed before training data by using LR algorithm. The average results across folds were Accuracy = 75.1%, Precision = 64.1%, Recall = 72.6%, F1 = 67.5%, and ROC-AUC = 0.834, as shown in Figure 4.1.4.

By comparing the results after cross-validation and in baseline training in Table 4.1.2, we can observe that the average results are slightly higher or lower than the result of train or test set. Cross-validation provides a more robust estimate and reliable metric since average results are obtained from multiple training, thus, the small difference between the two results indicates that the model is able to perform reliable and stable performance without overfitting.



```
=== 5-Fold Cross Validation on Training Set ===
1. Accuracy scores : [0.7804878  0.74796748 0.76422764 0.76422764 0.75409836]  | Mea
n: 0.7622017859522857
2. Precision scores: [0.68181818 0.61538462 0.675      0.64583333 0.625     ]  | Mea
n: 0.6486072261072261
3. Recall scores   : [0.69767442 0.74418605 0.62790698 0.72093023 0.71428571]  | Mea
n: 0.7009966777408638
4. F1 scores       : [0.68965517 0.67368421 0.65060241 0.68131868 0.66666667]  | Mea
n: 0.6723854281128021
5. ROC-AUC scores  : [0.86569767 0.83982558 0.85145349 0.84767442 0.81696429]  | Mea
n: 0.8443230897009967
```

Figure 4.1.4 Average Result After Cross-Validation (LR)

Table 4.1.2 Comparison of Result in Baseline Training and After Cross-Validation (LR)

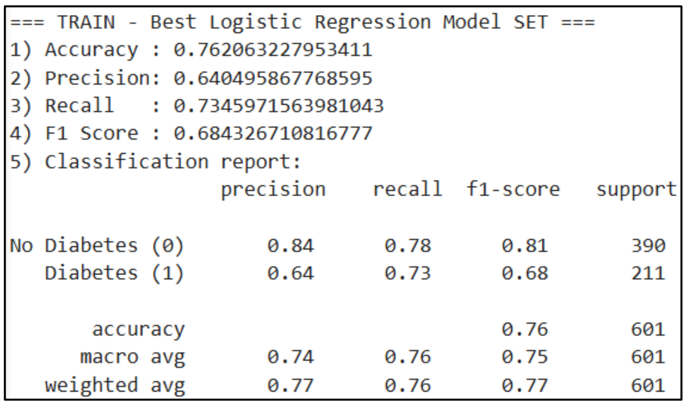
Metrics	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC-AUC
Train Set in Baseline Training	76.7	65.0	73.0	68.8	0.841
Test Set in Baseline Training	76.2	68.9	58.5	63.3	0.851
Average Result After Cross-Validation	75.1	64.1	72.6	67.5	0.833

4.1.3 Result on Hyperparameter Tuning

Before obtaining final results, further improvements are made by adjusting the parameters and finding the best combination of them. Once the hyperparameter tuning is done by using GridSearchCV, the best combination of parameters suggested is stated below.

- **Best Parameters**
    - C: {0.1}
    - penalty: {l2}
    - solver: {liblinear}

Finally, the model achieved Accuracy = 76.5%, Precision = 64.6%, Recall = 72.4%, F1 Score = 68.3%, and ROC-AUC = 0.841, on train set as shown in Figure 4.1.5. By comparing the test set, the model achieved Accuracy = 75.5%, Precision = 66.7%, Recall = 60.4%, F1 Score = 63.4%, and ROC-AUC = 0.852, as shown in Figure 4.1.6.

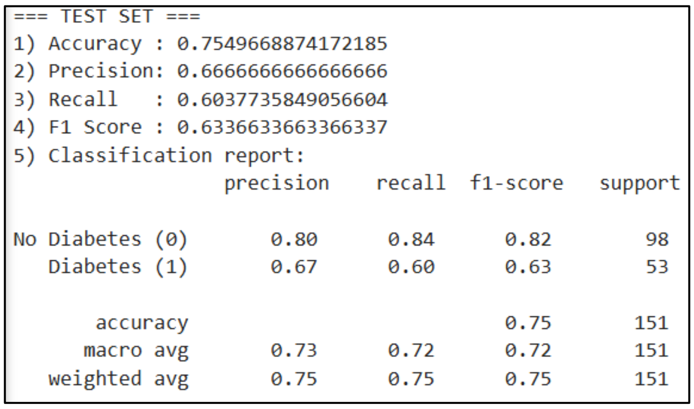


```
=== TRAIN - Best Logistic Regression Model SET ===
1) Accuracy : 0.762063227953411
2) Precision: 0.640495867768595
3) Recall   : 0.7345971563981043
4) F1 Score : 0.684326710816777
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.84      0.78      0.81       390
   Diabetes (1)       0.64      0.73      0.68       211

       accuracy                           0.76       601
      macro avg       0.74      0.76      0.75       601
   weighted avg       0.77      0.76      0.77       601
```

Figure 4.1.5 Result After Hyperparameter Tuning on Train Set (LR)



```
=== TEST SET ===
1) Accuracy : 0.7549668874172185
2) Precision: 0.6666666666666666
3) Recall   : 0.6037735849056604
4) F1 Score : 0.6336633663366337
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.80      0.84      0.82        98
   Diabetes (1)       0.67      0.60      0.63        53

       accuracy                           0.75       151
      macro avg       0.73      0.72      0.72       151
   weighted avg       0.75      0.75      0.75       151
```

Figure 4.1.6 Result After Hyperparameter Tuning on Test Set (LR)

As Table 4.1.3 shown below, there is small difference between the results in test set and in train set which indicates that the model provides great generalization ability without overfitting.

Table 4.1.3 Comparison of Result of Train Set and Test Set (LR) After Hyperparameter Tuning

Metrics	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Train Set	76.2%	64.0%	73.5%	68.4%	0.841
Test Set	75.5%	66.7%	60.4%	63.4%	0.852

On the other hand, the test set performs lower accuracy and recall but higher precision and ROC-AUC, it showed the model performs fewer true positive diabetes cases but fewer false positive diabetes predictions on unpredictable data, the confusion matrix has proved that as in Figure 4.1.7 and 4.1.8. Besides, 0.852 of ROC-AUC indicates that the model can classify diabetic and non-diabetic patients efficiently on unseen data.

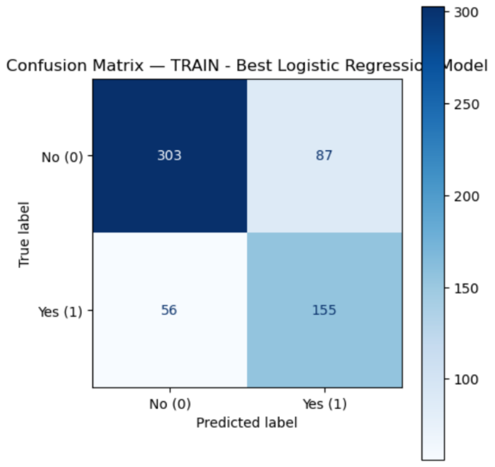


Figure 4.1.7 Confusion Matrix After Hyperparameter Tuning on Train Set (LR)

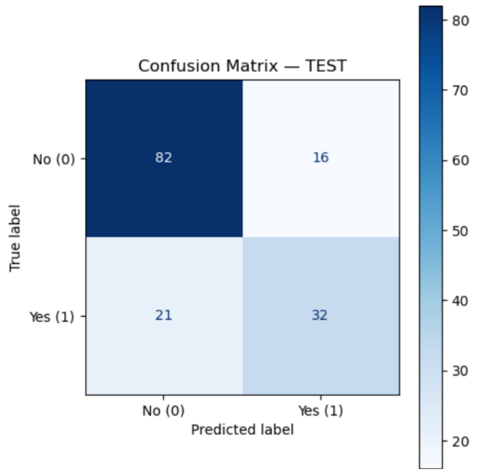


Figure 4.1.8 Confusion Matrix After Hyperparameter Tuning on Test Set (LR)

By using the best Linear Regression model, we have also visualized the result by illustrating Precision-Recall Curve and ROC Curve on both train set and test set. As the Figure shown below, we can conclude that there is no sign of overfitting because result in test set is slightly better than in train set.

```
7) Average Precision (AP): 0.7154585749169756
```

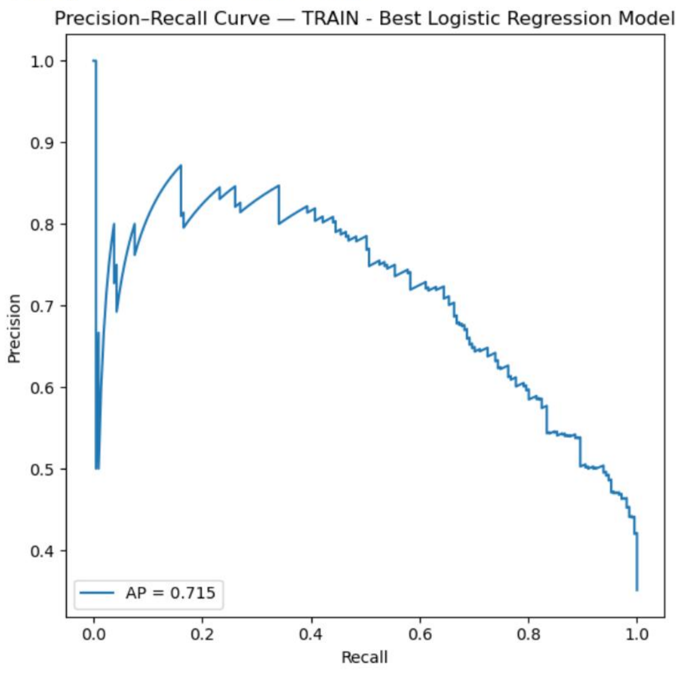


Figure 4.1.9 Precision-Recall Curve on Train Set (LR)

```
7) Average Precision (AP): 0.7630912842757527
```

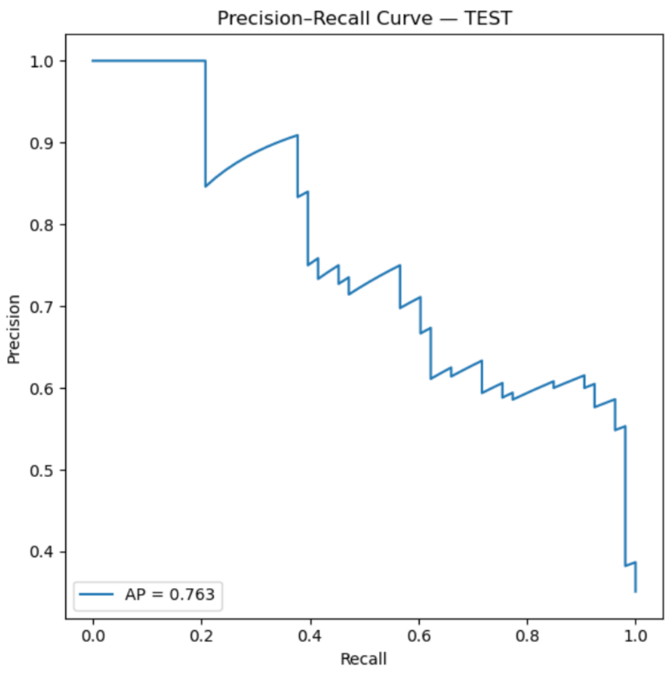
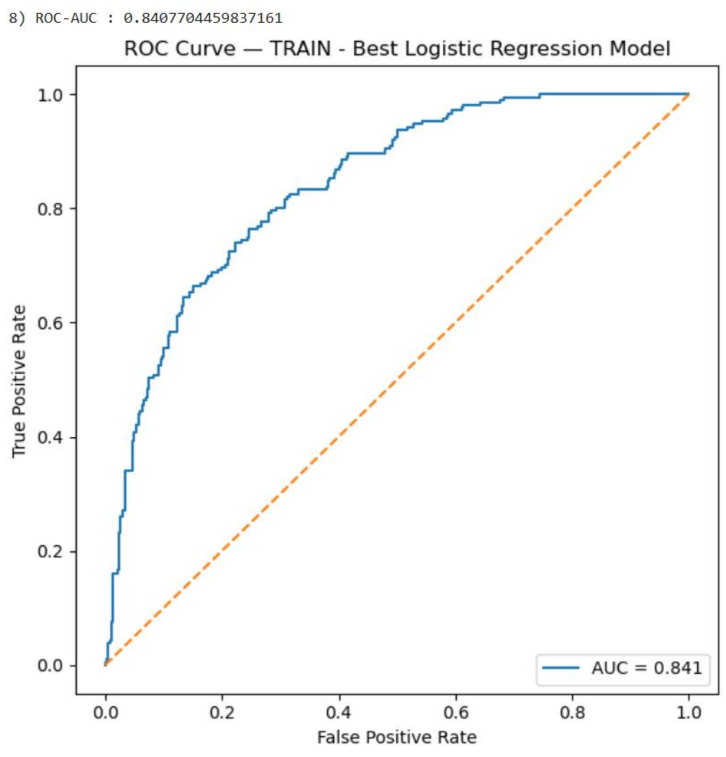
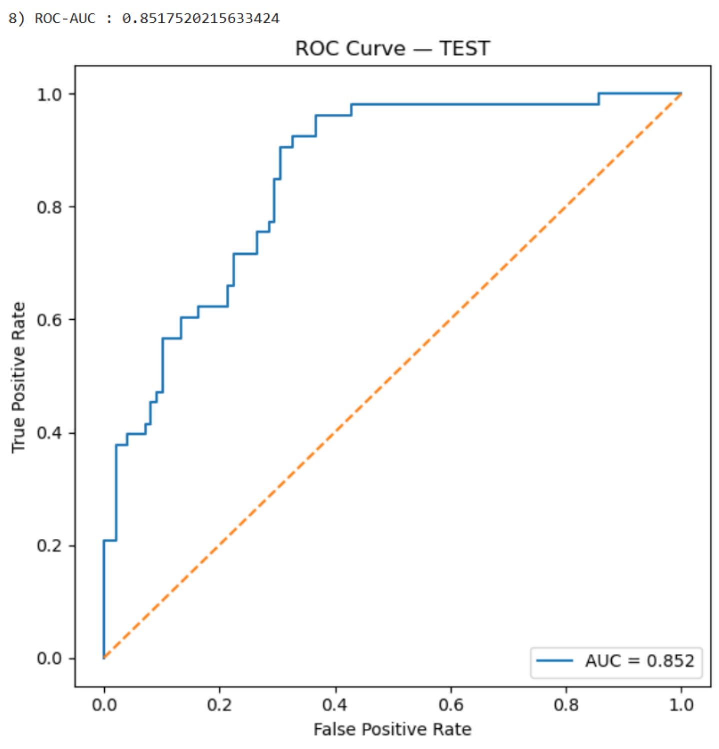


Figure 4.1.10 Precision-Recall Curve on Test Set (LR)



```
8) ROC-AUC : 0.8407704459837161
```

Figure 4.1.11 ROC Curve on Train Set (LR)



```
8) ROC-AUC : 0.8517520215633424
```

Figure 4.1.12 ROC Curve on Test Set (LR)

While we compare the results after hyperparameter tuning and in baseline training, we found that their performances are similar as Table 4.1.4 shown below. Thus, we can summarize that the initial model was already well-calibrated, but tuning confirms robustness.

Table 4.1.4 Comparison of All Result in Each Stage (LR)

Metrics	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC-AUC
Train Set in Baseline Training	76.7	65.0	73.0	68.8	0.841
Test Set in Baseline Training	76.2	68.9	58.5	63.3	0.851
Average Result After Cross-Validation	75.1	64.1	72.6	67.5	0.833
Train Set After Hyperparameter Tuning	76.2	64.0	73.5	68.4	0.841
Test Set After Hyperparameter Tuning	75.5	66.7	60.4	63.4	0.852

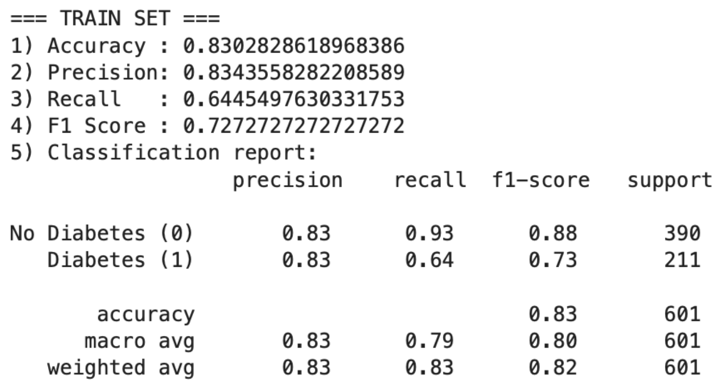
4.2 Support Vector Machine (SVM) Results

Written By: Low Jia Hao, 2302161

4.2.1 Result on Baseline Training (SVM)

Using the training data, a support vector machine (SVM) model trained with an RBF kernel and using the StandardScaler pipeline and median imputation performs exceptionally well. On the training set, the model achieves 83.0% accuracy, 83.4% precision, 64.5% recall, 72.7% F1 score, and 0.893 ROC-AUC, as illustrated in Figure 4.2.1. This illustrates how well the model can identify patterns in the training set of data.

However, the model's performance drastically deteriorates when tested on the test set. As illustrated in Figure 4.2.2, the ROC-AUC is 0.853, the accuracy is 75.5%, the precision is 78.6%, the recall is 41.5%, and the F1 score is 54.3%.

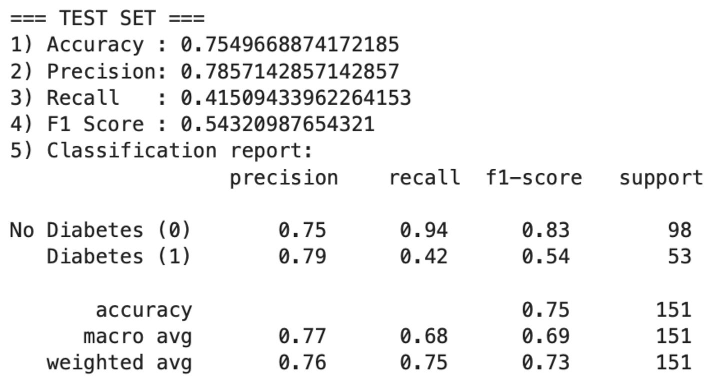


```
=== TRAIN SET ===
1) Accuracy : 0.8302828618968386
2) Precision: 0.8343558282208589
3) Recall   : 0.6445497630331753
4) F1 Score : 0.7272727272727272
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.83      0.93      0.88       390
   Diabetes (1)       0.83      0.64      0.73       211

       accuracy                           0.83       601
      macro avg       0.83      0.79      0.80       601
   weighted avg       0.83      0.83      0.82       601
```

Figure 4.2.1 Result of Baseline Training on Train Set (SVM)



```
=== TEST SET ===
1) Accuracy : 0.7549668874172185
2) Precision: 0.7857142857142857
3) Recall   : 0.41509433962264153
4) F1 Score : 0.54320987654321
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.75      0.94      0.83        98
   Diabetes (1)       0.79      0.42      0.54        53

       accuracy                           0.75       151
      macro avg       0.77      0.68      0.69       151
   weighted avg       0.76      0.75      0.73       151
```

Figure 4.2.2 Result of Baseline Training on Test Set (SVM)

As shown in Table 4.2.1, performance between the training and test sets drops significantly. In particular, recall drops sharply from 64.5% to 41.5%, demonstrating that the baseline model overfits the training data. While precision on the test set is high, the extremely low recall indicates that the model fails to identify over half of the actual diabetic patients, a serious flaw in the healthcare field. The ROC curve in Figure 4.2.3 visually demonstrates the performance gap between the training and test environments.

Table 4.2.1 Comparison of Result of Train Set and Test Set in Baseline Training (SVM)

Metrics	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Train Set	83.0%	83.4%	64.5%	72.7%	0.893
Test Set	75.5%	78.6%	41.5%	54.3%	0.853

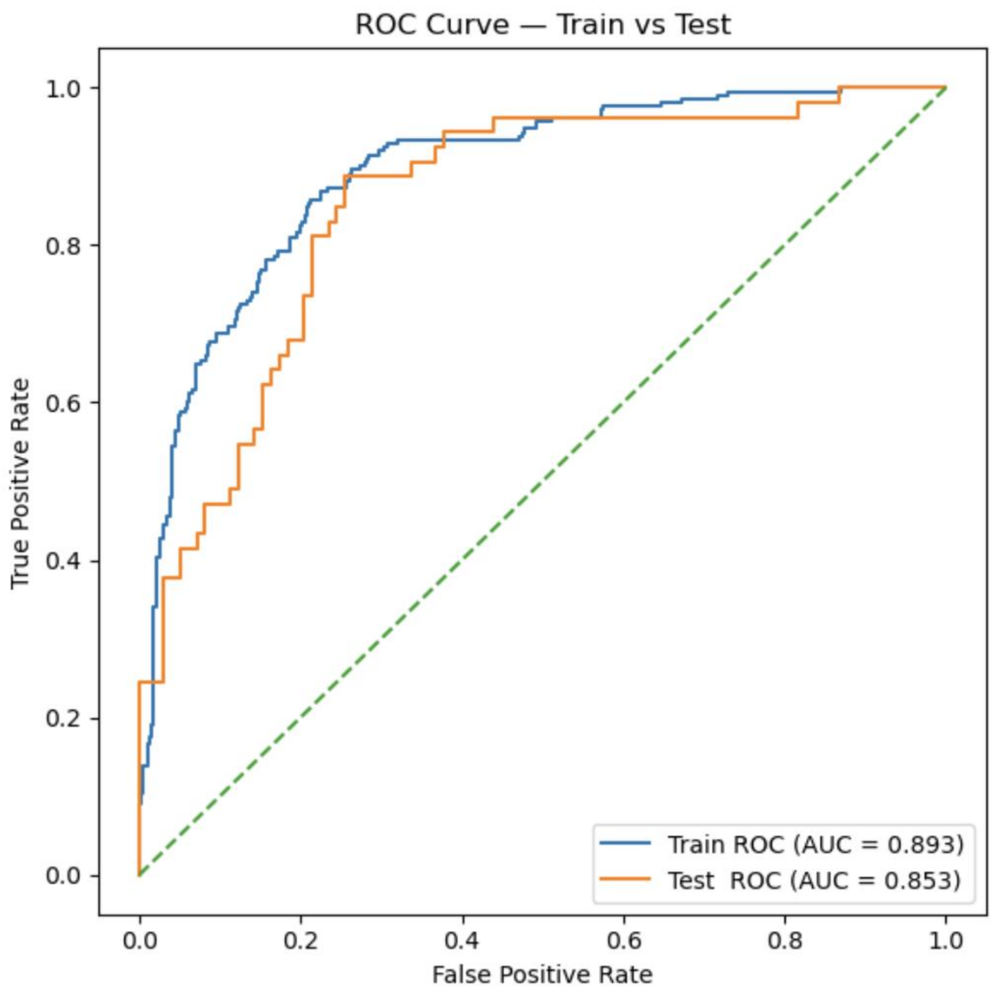
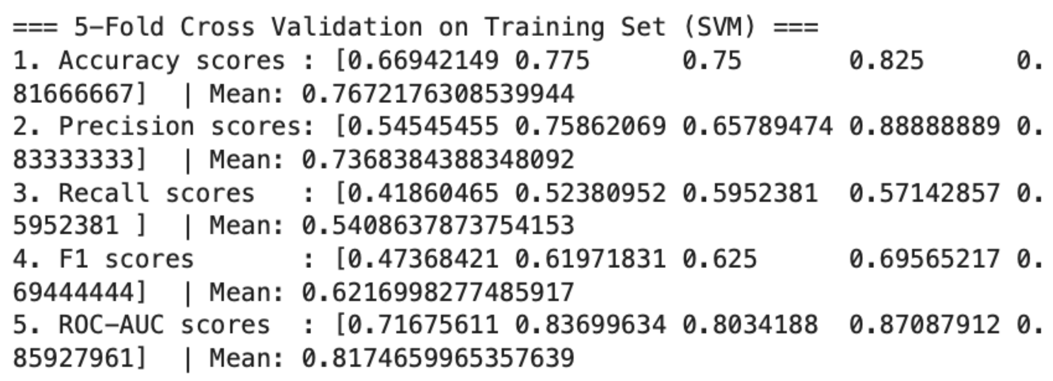


Figure 4.2.3 ROC Curve of Train and Test Set in Baseline Training (SVM)

4.2.2 Result on Cross-Validation (SVM)

To mitigate the observed overfitting and obtain more stable estimates of model performance, a 5-fold stratified cross-validation was performed on the training data. The average results across the five folds were: accuracy = 76.7%, precision = 73.7%, recall = 54.1%, F1 score = 62.2%, and ROC-AUC = 0.817, as shown in Figure 4.2.4.



```
=== 5-Fold Cross Validation on Training Set (SVM) ===
1. Accuracy scores : [0.66942149 0.775      0.75       0.825      0.
81666667]  | Mean: 0.7672176308539944
2. Precision scores: [0.54545455 0.75862069 0.65789474 0.88888889 0.
83333333]  | Mean: 0.7368384388348092
3. Recall scores   : [0.41860465 0.52380952 0.5952381  0.57142857 0.
5952381 ]  | Mean: 0.5408637873754153
4. F1 scores       : [0.47368421 0.61971831 0.625      0.69565217 0.
69444444]  | Mean: 0.6216998277485917
5. ROC-AUC scores  : [0.71675611 0.83699634 0.8034188  0.87087912 0.
85927961]  | Mean: 0.8174659965357639
```

Figure 4.2.4 Average Result After Cross-Validation (SVM)

Comparing the results in Table 4.2.2, we can see that cross-validation provides a more realistic and robust assessment of the model's true performance. These average metrics are significantly lower than the initially inflated training scores, but more reliably reflect the model's expected performance on unseen data. This process confirms the limitations of the baseline model, particularly its mediocre recall, and provides a more realistic benchmark before hyperparameter tuning.

Table 4.2.2 Comparison of Result in Baseline Training and After Cross-Validation (SVM)

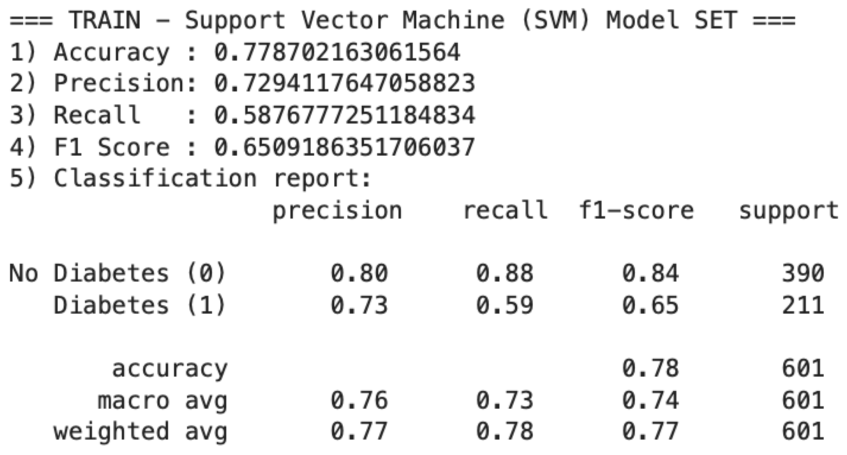
Metrics	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Train Set in Baseline Training	83.0%	83.4%	64.5%	72.7%	0.893
Test Set in Baseline Training	75.5%	78.6%	41.5%	54.3%	0.853
Average Result After Cross-Validation	76.7%	73.7%	54.1%	62.2%	0.817

4.2.3 Result on Hyperparameter Tuning (SVM)

We used GridSearchCV to conduct an exhaustive search in order to identify the best configuration for the SVM model. The following is the combination of parameters that was found to be optimal:

Best Parameters
C: {1}
kernel: {'linear'}
gamma: {'scale'}

These settings were used to retrain the model. The model's performance on the training set was as follows: 77.9% accuracy, 72.9% precision, 58.8% recall, 65.1% F1 score, and ROC-AUC of 0.840 (Figure 4.2.5). As illustrated in Figure 4.2.6, the final adjusted model reached 74.8% accuracy, 75.9% precision, 41.5% recall, 53.7% F1 score, and ROC-AUC of 0.856 on the test set.

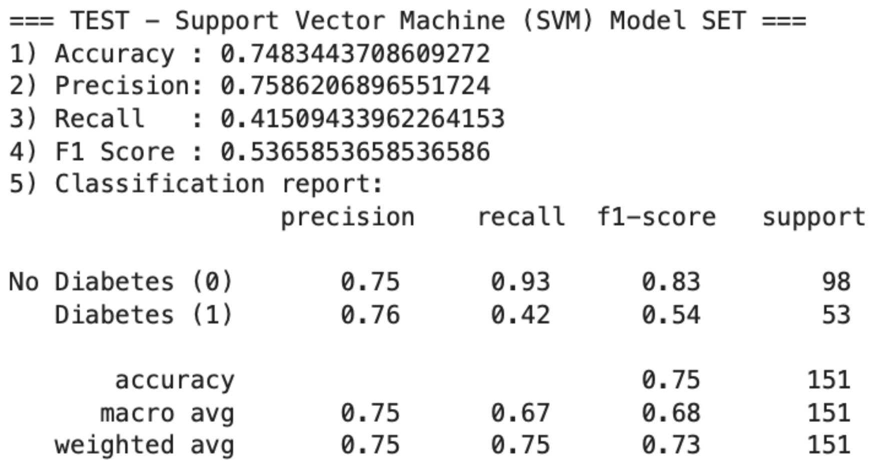


```
=== TRAIN - Support Vector Machine (SVM) Model SET ===
1) Accuracy : 0.778702163061564
2) Precision: 0.7294117647058823
3) Recall   : 0.5876777251184834
4) F1 Score : 0.6509186351706037
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.80      0.88      0.84       390
   Diabetes (1)       0.73      0.59      0.65       211

       accuracy                           0.78       601
      macro avg       0.76      0.73      0.74       601
   weighted avg       0.77      0.78      0.77       601
```

Figure 4.2.5 Result After Hyperparameter Tuning on Train Set (SVM)


```
=== TEST - Support Vector Machine (SVM) Model SET ===
1) Accuracy : 0.7483443708609272
2) Precision: 0.7586206896551724
3) Recall   : 0.41509433962264153
4) F1 Score : 0.5365853658536586
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.75      0.93      0.83        98
   Diabetes (1)       0.76      0.42      0.54        53

       accuracy                           0.75       151
      macro avg       0.75      0.67      0.68       151
   weighted avg       0.75      0.75      0.73       151
```

Figure 4.2.6 Result After Hyperparameter Tuning on Test Set (SVM)

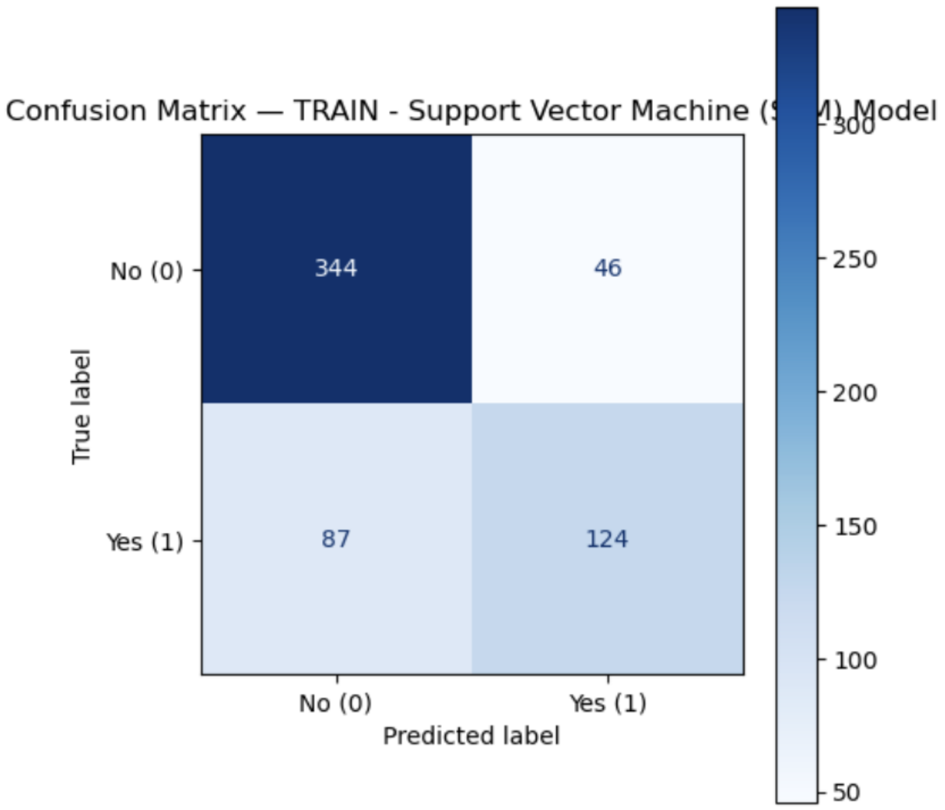


Figure 4.2.7 Confusion Matrix After Hyperparameter Tuning on Train Set (SVM)

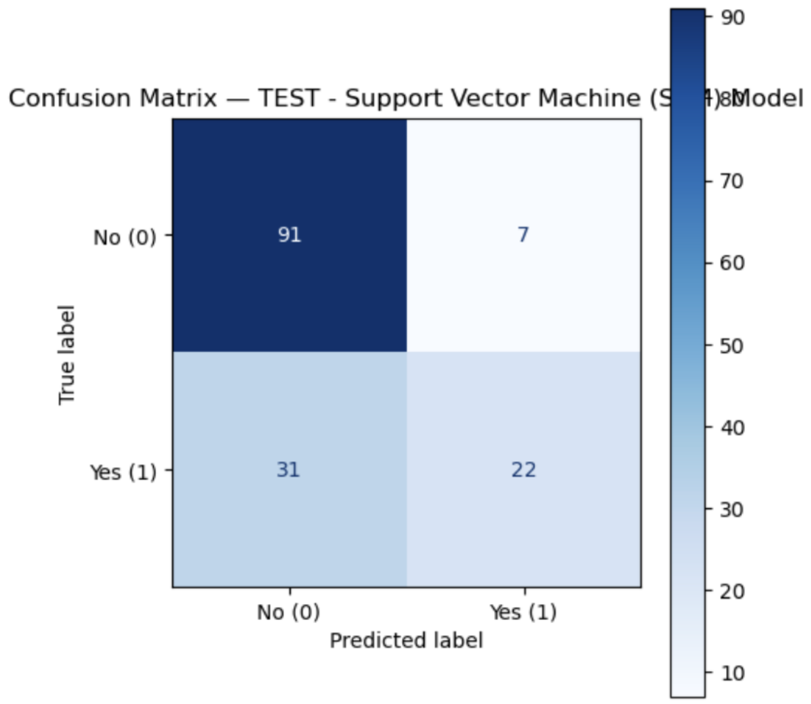


Figure 4.2.8 Confusion Matrix After Hyperparameter Tuning on Test Set (SVM)

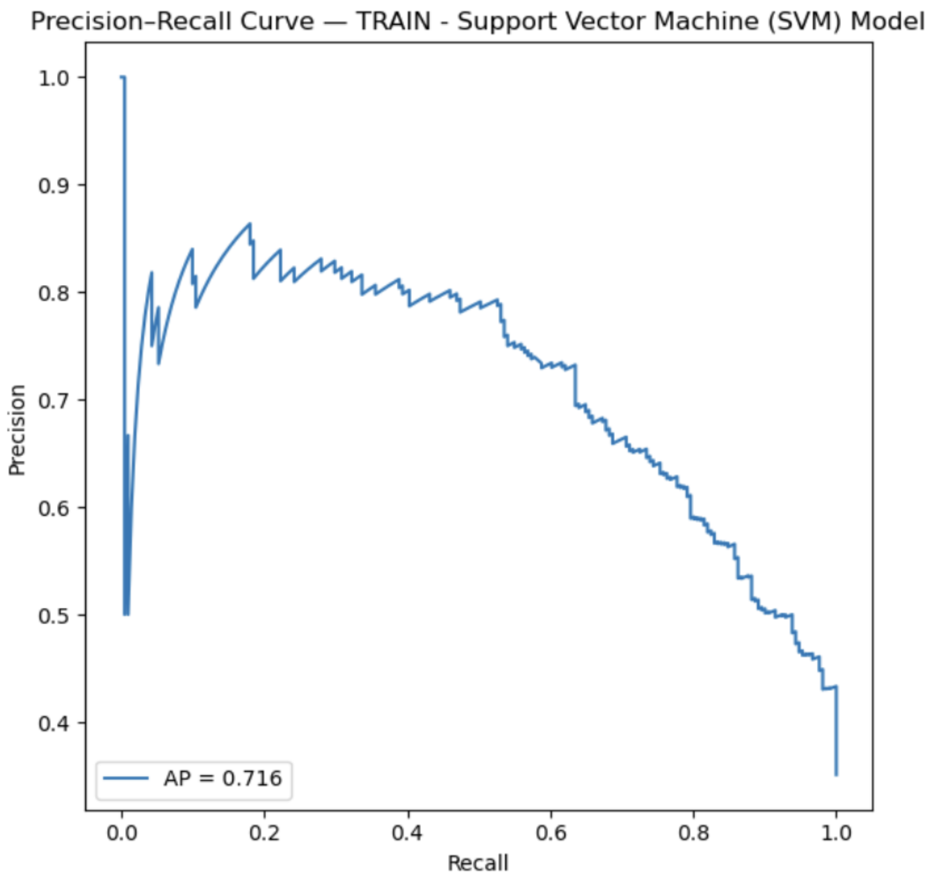


Figure 4.2.9 Precision-Recall Curve on Train Set (SVM)

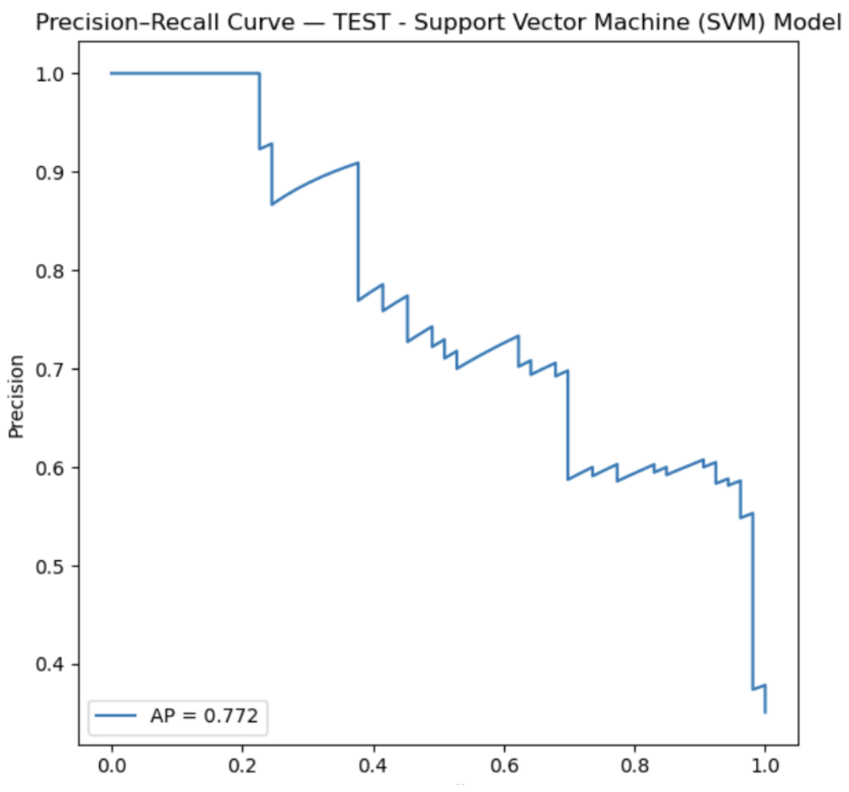


Figure 4.2.10 Precision-Recall Curve on Test Set (SVM)

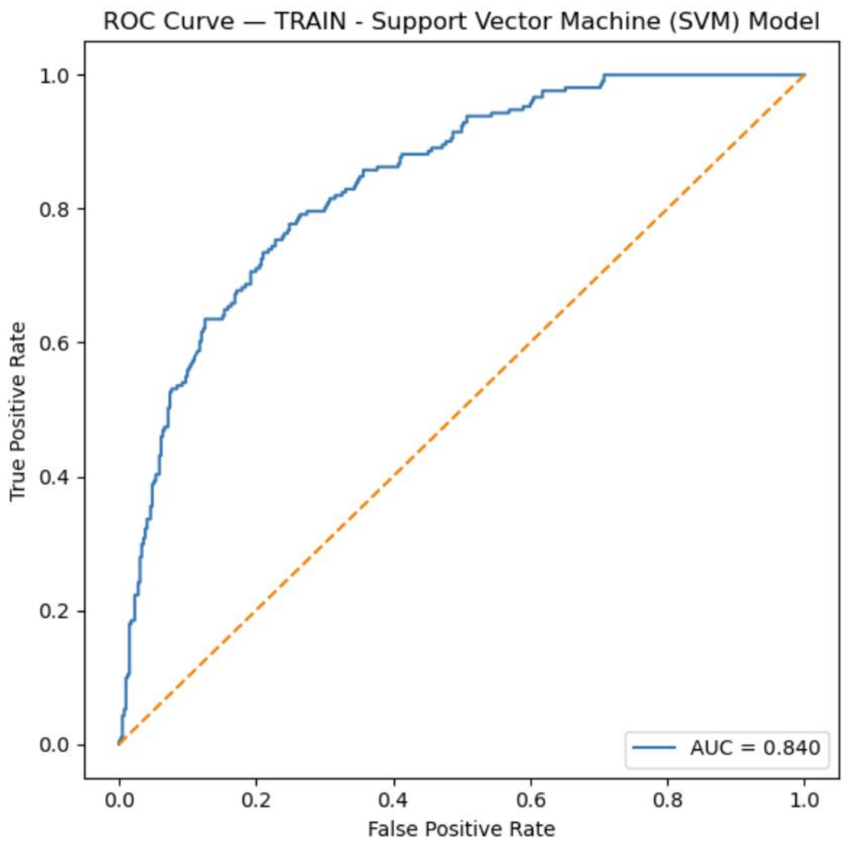


Figure 4.2.11 ROC Curve on Train Set (SVM)

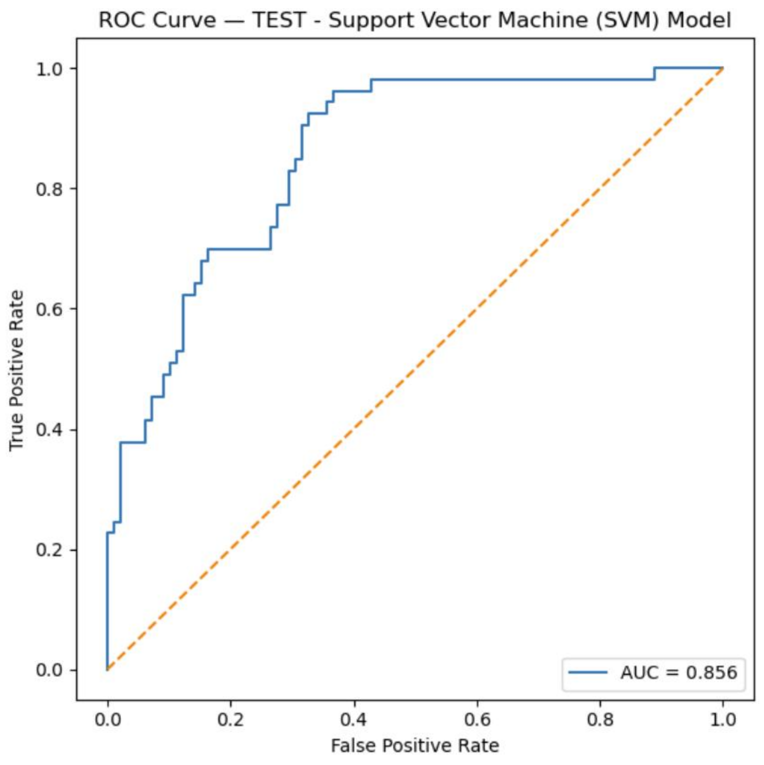


Figure 4.2.12 ROC Curve on Test Set (SVM)

The performance difference between the training and test sets has decreased, as indicated by Table 4.2.3, suggesting that overfitting has been successfully decreased by the tuning. Although the model achieves high precision on the test set, its recall is still very low, missing 31 cases of diabetes, as shown by the confusion matrix (Figures 4.2.7 and 4.2.8). The model is a good classifier overall, but its low sensitivity (recall) is still its biggest flaw, according to the test set's ROC curve (Figure 4.2.12), which displays a high AUC of 0.856.

Table 4.2.3 Comparison of Result of Train Set and Test Set (SVM) After Hyperparameter Tuning

Metrics	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Train Set	77.9%	72.9%	58.8%	65.1%	0.840
Test Set	74.8%	75.9%	41.5%	53.7%	0.856

It is evident that hyperparameter tuning produced a more stable and broadly applicable model when comparing the final tuned results with the baseline results in Table 4.2.4. It did not, however, address the root cause of poor recall. This implies that even after optimization, the SVM model is unable to reliably detect positive diabetes cases for this specific dataset, which restricts its clinical use.

Table 4.2.4 Comparison of All Result in Each Stage (SVM)

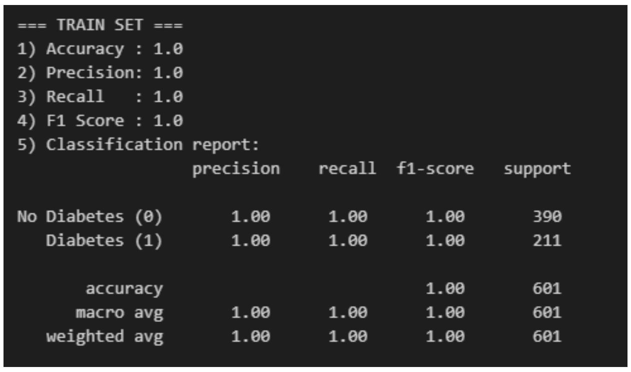
Metrics	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC-AUC
Train Set in Baseline Training	83.0%	83.4%	64.5%	72.7%	0.893
Test Set in Baseline Training	75.5%	78.6%	41.5%	54.3%	0.853
Average Result After Cross-Validation	76.7%	73.7%	54.1%	62.2%	0.817
Train Set After Hyperparameter Tuning	77.9%	72.9%	58.8%	65.1%	0.840
Test Set After Hyperparameter Tuning	74.8%	75.9%	41.5%	53.7%	0.856

4.3 Random Forest (RF) Results

Written By: Loh Chia Heung, 2301684

4.3.1 Result on Baseline Training (RF)

After training a Random Forest model with default hyperparameters, the model achieved perfect performance on the training set. As shown in Figure 4.3.1, the train set reached **Accuracy = 100%, Precision = 100%, Recall = 100%, F1 Score = 100%, and ROC-AUC = 1.000**. The classification report confirmed that both diabetic and non-diabetic cases were classified without error, which is also reflected in the confusion matrix where no misclassifications occurred. The Precision-Recall and ROC curves further illustrate this perfect performance on the training data.



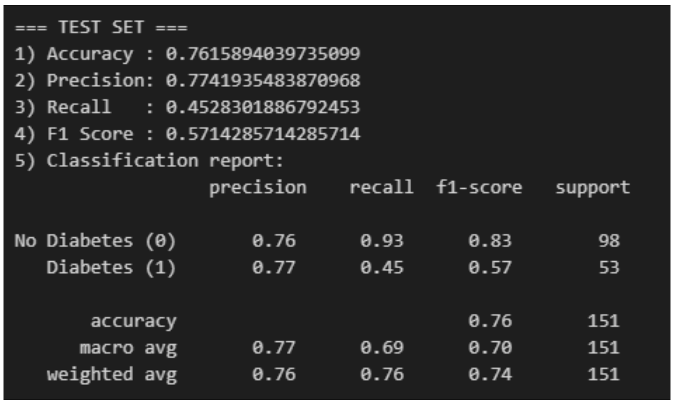
```
=== TRAIN SET ===
1) Accuracy : 1.0
2) Precision: 1.0
3) Recall   : 1.0
4) F1 Score : 1.0
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       1.00      1.00      1.00       390
   Diabetes (1)       1.00      1.00      1.00       211

       accuracy                           1.00       601
      macro avg       1.00      1.00      1.00       601
   weighted avg       1.00      1.00      1.00       601
```

Figure 4.3.1 Result of Baseline Training on Train Set (RF)

However, when evaluated on the test set, the model performance dropped significantly, as shown in Figure 4.3.2. **The test set achieved Accuracy = 76.2%, Precision = 77.4%, Recall = 45.3%, F1 Score = 57.1%, and ROC-AUC = 0.856**. The confusion matrix indicates that while the model identified non-diabetic cases with high accuracy (91 correctly classified out of 98), it struggled to correctly identify diabetic cases, with only 24 out of 53 correctly classified. This imbalance highlights a trade-off between precision and recall, where the model is more confident in predicting positives but misses a substantial number of actual diabetic patients.



```
=== TEST SET ===
1) Accuracy : 0.7615894039735099
2) Precision: 0.7741935483870968
3) Recall   : 0.4528301886792453
4) F1 Score : 0.5714285714285714
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.76      0.93      0.83        98
   Diabetes (1)       0.77      0.45      0.57        53

       accuracy                           0.76       151
      macro avg       0.77      0.69      0.70       151
   weighted avg       0.76      0.76      0.74       151
```

Figure 4.3.2 Result of Baseline Training on Test Set (RF)

As Table 4.3.1 shows, the difference between the training and testing results indicates that the Random Forest model overfitted the training data. Although the ROC-AUC score on the test set remained reasonably strong (0.856) as showed in Figure 4.3.3, the recall score suggests the model may not be reliable in clinical settings where correctly identifying diabetic patients is critical.

Table 4.3.1 Comparison of Result of Train Set and Test Set in Baseline Training (RF)

Metrics	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Train Set	100%	100%	100%	100%	1.000
Test Set	76.2%	77.4%	45.3%	57.1%	0.856

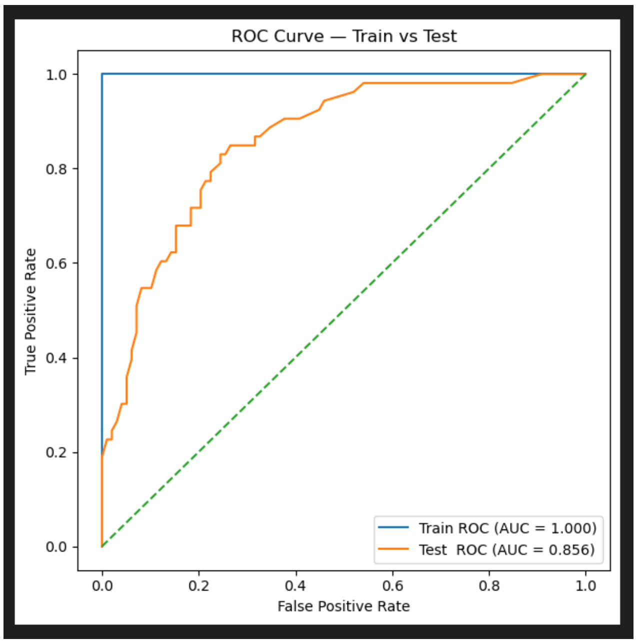
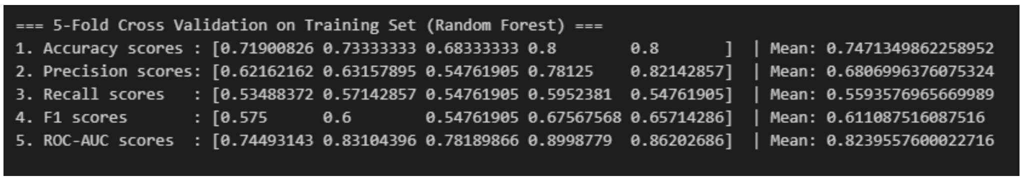


Figure 4.3.3 ROC Curve of Train and Test Set in Baseline Training (RF)

4.3.2 Result on Cross-Validation (RF)

To further validate the stability and generalization of the Random Forest model, a 5-fold stratified cross-validation was conducted on the training set. The average results across the folds were **Accuracy = 74.7%, Precision = 68.1%, Recall = 55.9%, F1 Score = 61.1%, and ROC-AUC = 0.824**, as shown in Figure 4.3.4.



```
=== 5-Fold Cross Validation on Training Set (Random Forest) ===
1. Accuracy scores : [0.71900826 0.73333333 0.68333333 0.8        0.8       ]  | Mean: 0.7471349862258952
2. Precision scores: [0.62162162 0.63157895 0.54761905 0.78125    0.82142857]  | Mean: 0.6806996376075324
3. Recall scores   : [0.53488372 0.57142857 0.54761905 0.5952381  0.54761905]  | Mean: 0.5593576965669989
4. F1 scores       : [0.575      0.6        0.54761905 0.67567568 0.65714286]  | Mean: 0.611087516087516
5. ROC-AUC scores  : [0.74493143 0.83104396 0.78189866 0.8998779  0.86202686]  | Mean: 0.8239557600022716
```

Figure 4.3.4 Average Result After Cross-Validation (RF)

By comparing these values with the baseline training results in Table 4.3.2, we can observe that cross-validation produced lower scores than the perfect training accuracy obtained earlier. This difference highlights that the Random Forest model had **overfitted to the training data**, and the **cross-validation** provides a more **realistic and robust estimate of performance**. Despite this reduction, the ROC-AUC score remains consistently strong above 0.82, confirming that this model retains good discriminative power. The moderate recall score suggests that the model still struggles to identify diabetic patients reliably, which limits its clinical applicability without further tuning.

Table 4.3.2 Comparison of Result in Baseline Training and After Cross-Validation (RF)

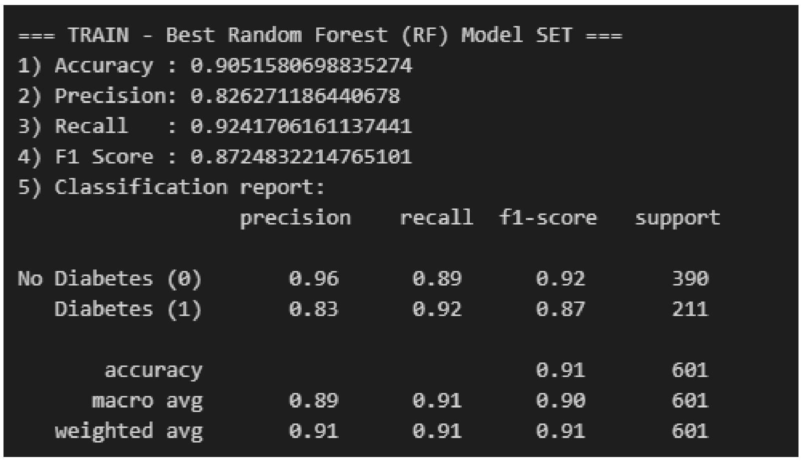
Metrics	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC-AUC
Train Set in Baseline Training	100%	100%	100%	100%	1.000
Test Set in Baseline Training	76.2%	77.4%	45.3%	57.1%	0.856
Average Result After Cross-Validation	74.7	68.1	55.9	61.1	0.824

4.3.3 Result on Hyperparameter Tuning (RF)

To further improve the performance and reduce the overfitting, hyperparameter tuning was performed on the Random Forest model using GridSearchCV with a 5-fold stratified cross-validation. The grid search explored different values of the number of estimators, maximum depth of trees, minimum samples required for split and leaf nodes, and the maximum features considered for splitting. The best combination of parameters suggested is stated below:

- **Best Parameters**
  - n_estimators: 100
  - max_depth: 10
  - min_samples_split: 2
  - min_samples_leaf: 4
  - max_features: "sqrt"

The fine-tuned model showed improved generalization compared to the baseline. On the training set, the best Random Forest achieved **Accuracy = 90.5%, Precision = 82.6%, Recall = 92.4%, F1 Score = 87.2%, and ROC-AUC = 0.975**, as shown in Figure 4.3.5. This indicates that the tuned model still performed strongly on the training data while no longer achieving unrealistic perfect scores, reducing the risk of overfitting.



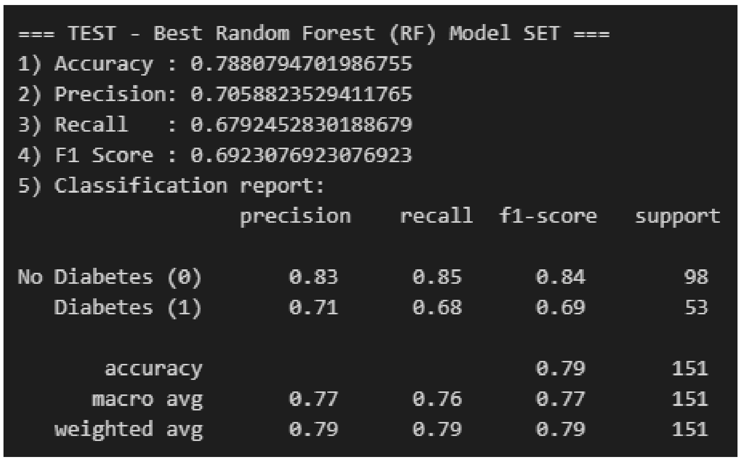
```
=== TRAIN - Best Random Forest (RF) Model SET ===
1) Accuracy : 0.9051580698835274
2) Precision: 0.826271186440678
3) Recall   : 0.9241706161137441
4) F1 Score : 0.8724832214765101
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.96      0.89      0.92       390
   Diabetes (1)       0.83      0.92      0.87       211

       accuracy                           0.91       601
      macro avg       0.89      0.91      0.90       601
   weighted avg       0.91      0.91      0.91       601
```

Figure 4.3.5 Result After Hyperparameter Tuning on Train Set (RF)

On the test set, the tuned model achieved **Accuracy = 78.8%, Precision = 70.6%, Recall = 67.9%, F1 Score = 69.2%, and ROC-AUC = 0.860**, as shown in Figure 4.3.6. Compared with the baseline results, both recall and F1 score improved, which demonstrates that hyperparameter tuning enhanced the model's ability to identify diabetic patients while maintaining balanced overall performance.



```
=== TEST - Best Random Forest (RF) Model SET ===
1) Accuracy : 0.7880794701986755
2) Precision: 0.7058823529411765
3) Recall   : 0.6792452830188679
4) F1 Score : 0.6923076923076923
5) Classification report:
                  precision    recall  f1-score   support

No Diabetes (0)       0.83      0.85      0.84        98
   Diabetes (1)       0.71      0.68      0.69        53

       accuracy                           0.79       151
      macro avg       0.77      0.76      0.77       151
   weighted avg       0.79      0.79      0.79       151
```

Figure 4.3.6 Result After Hyperparameter Tuning on Train Set (RF)

As Table 4.3.3 shows below, the performance gap between training and test sets is now smaller compared to the baseline training, indicating that the tuned Random Forest model provides better generalization and reduces overfitting. The ROC-AUC score of 0.860 on the test set further highlights the model's strong discriminative power in distinguishing diabetic and non-diabetic patients.

Table 4.3.3 Comparison of Result of Train Set and Test Set After Hyperparameter Tuning (RF)

Metrics	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Train Set	90.5%	82.6%	92.4%	87.2%	0.975
Test Set	78.8%	70.6%	67.9%	69.2%	0.860

On the other hand, the confusion matrices (Figures 4.3.7 and Figure 4.3.8) showed that the finetuned model still correctly classified most non-diabetic cases while improving the number of correctly identified diabetic patients compared to the baseline model. Precision-Recall and ROC curves on both the training and test sets (Figures 4.3.9–4.3.12) further confirm that the hyperparameter tuning produced a more balanced classifier without significant overfitting.

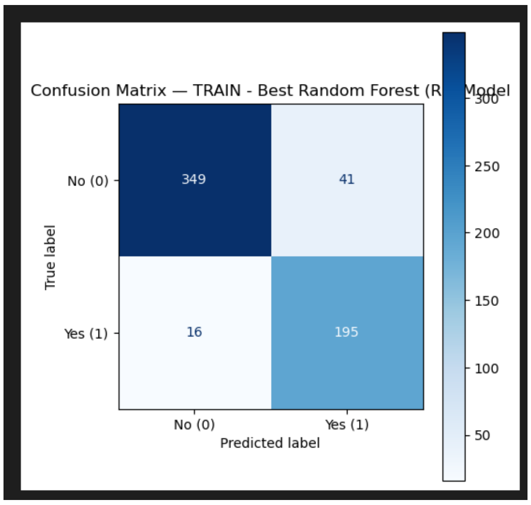


Figure 4.3.7 Confusion Matrix After Hyperparameter Tuning on Train Set (RF)

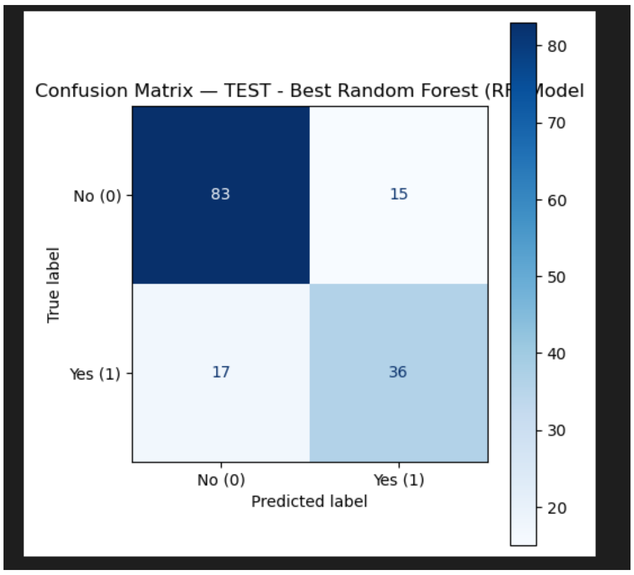


Figure 4.3.8 Confusion Matrix After Hyperparameter Tuning on Test Set (RF)

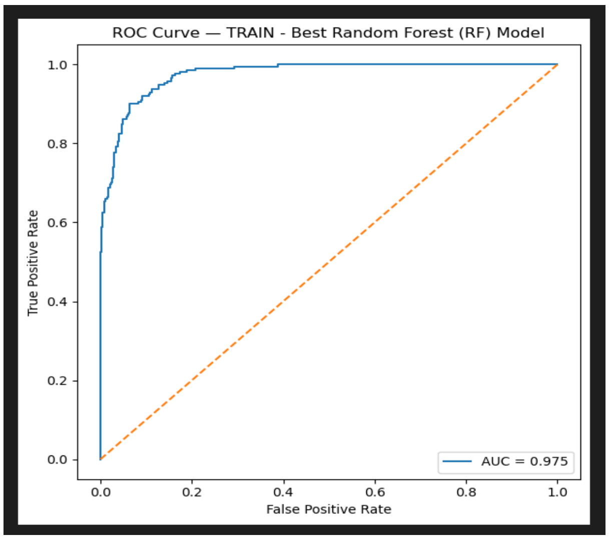


Figure 4.3.9 ROC Curve on Train Set (RF)

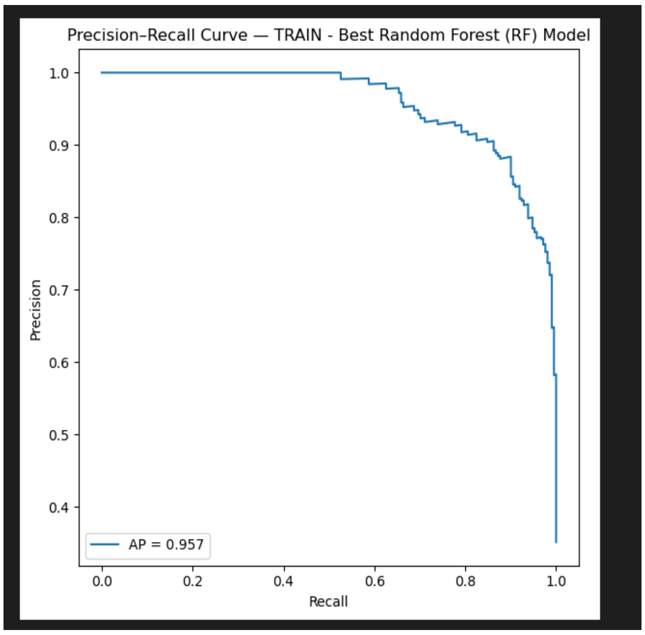


Figure 4.3.10 Precision-Recall Curve on Train Set (RF)

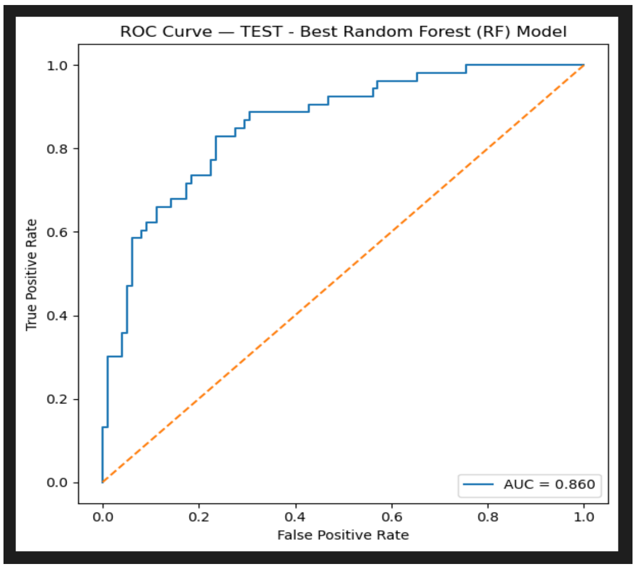


Figure 4.3.11 ROC Curve on Test Set (RF)

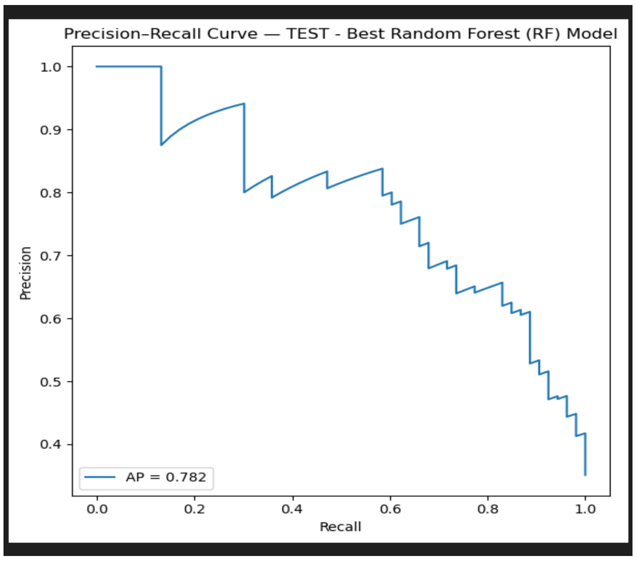


Figure 4.3.12 Precision-Recall Curve on Test Set (RF)

While comparing the results after hyperparameter tuning with the baseline training and cross-validation, it can be observed that tuning improved recall and F1 score on the test set, showing better balance between identifying diabetic patients and avoiding false positives. As Table 4.3.4 shows, the tuned Random Forest model provided the most reliable performance among the three stages, confirming the effectiveness of the tuning process in reducing overfitting and enhancing the generalization.

Table 4.3.4 Comparison of All Result in Each Stage (RF)

Metrics	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC-AUC
Train Set in Baseline Training	100.0	100.0	100.0	100.0	1.000
Test Set in Baseline Training	76.2	77.4	45.3	57.1	0.856
Average Result After Cross-Validation	74.7	68.1	55.9	61.1	0.824
Train Set After Hyperparameter Tuning	90.5	82.6	92.4	87.2	0.975
Test Set After Hyperparameter Tuning	78.8	70.6	67.9	69.2	0.860

In summary, the Random Forest (RF) model initially achieved perfect performance on the training set during the baseline training, but this is considered as the case of overfitting, as reflected by the drop in recall and F1 score on the test set. Cross validation provided a more balanced view, which confirmed that the true performance of the model was lower and less stable than the baseline suggested. After the hyperparameter tuning, the model's performance improved substantially, particularly in the recall and F1 score on the test set, demonstrating a better ability to detect the diabetic patients while maintaining strong discriminative power. Although the tuned Random Forest still showed slightly higher results on the training set, the reduced performance gap and consistent ROC-AUC values indicated that the final model generalizes more effectively and provides a more reliable classifier compared to the baseline.

4.4 Model Comparison & Best Model Selection

Written By: All Members

To determine the best performing model, Linear Regression (LR), Support Vector Machine (SVM), and Random Forest (RF) were compared across multiple metrics on both the training and test sets. The results are summarized in Table 4.4.1.

Table 4.4.1 Comparison of Final Model Performance

Model Used		Linear Regression (LR) / %	Support Vector Machine (SVM) / %	Random Forest (RF) / %
Train Set	Accuracy	76.2	77.9	90.5
	Precision	64.0	72.9	82.6
	Recall	73.5	58.8	92.4
	F1 Score	68.4	65.1	87.2
	ROC-AUC	84.1	84.0	95.5
Test Set	Accuracy	75.5	74.8	78.8
	Precision	66.7	75.9	70.6
	Recall	60.4	41.5	67.9
	F1 Score	63.4	53.7	69.2
	ROC-AUC	85.2	77.2	86.0

Written by: Loh Chia Heung, 2301684

From the comparison, Logistic Regression (LR) performed consistently across training and testing with balanced scores, and SVM achieved the highest test precision (75.9%). It showed that it is effective in avoiding the false positives. However, SVM's recall was significantly lower, which is 41.5%, indicating that it failed to correctly identify many diabetic patients.

Random Forest (RF), on the other hand, delivered the strongest overall performance. On the training set, RF achieved the highest accuracy (90.5%), recall (92.4%), and F1 score (87.2%), demonstrating its ability to capture complex patterns in the dataset. In addition, on the test set, Random Forest maintained superior recall (67.9%), F1 score (69.2%), and ROC-AUC (86.0%) compared to both LR and SVM. This indicates that

RF is better at identifying diabetic patients while preserving strong generalization ability.

Therefore, **Random Forest (RF)** was selected as the **best model** for this study, because it provides the most balanced trade-off between accuracy, precision, recall, and ROC-AUC. Its higher recall and F1 score on the test set are particularly important in a healthcare context, where minimizing false negatives (undiagnosed diabetic patients) is critical.

4.5 Conclusion

Written by: Loh Chia Heung, 2301684

In this chapter, the results of three machine learning models, Logistic Regression (LR), Support Vector Machine (SVM), and Random Forest (RF) were presented and analyzed. The baseline training, cross-validation, and hyperparameter tuning stages provided detailed insights into each model's strengths and weaknesses. Logistic Regression demonstrated stable and balanced performance, while Support Vector Machine achieved high precision but struggled with recall, limiting its effectiveness in identifying diabetic patients. Random Forest initially suffered from overfitting, but after the hyperparameter tuning, achieved the best overall performance with strong recall, F1 score, and ROC-AUC on the test set.

In conclusion, **Random Forest** (RF) was selected as the **most suitable model** for prediction for diabetes prediction in this study, as it offers the most reliable balance between predictive accuracy and clinical relevance.

REFERENCES

[1] “Diabetes Atlas 589 people worldwide have diabetes million.” Available: https://diabetesatlas.org/media/uploads/sites/3/2025/04/IDF_Atlas_11th_Edition_2025-1.pdf

[2] Marfah, “Global Diabetes Statistics: How Common is Diabetes in 2025?,” Staple and Stone, Mar. 13, 2025. https://remedytalks.com/global-diabetes-statistics-how-common-is-diabetes-in-2025/

[3] N. A. ElSayed et al., “2. Diagnosis and Classification of diabetes: Standards of Care in Diabetes—2025,” Diabetes Care, vol. 48, no. Supplement_1, pp. S27–S49, Dec. 2024, doi: 10.2337/dc25-s002.

[4] “Guide to diabetes testing methods and their benefits.” https://ampath.com/blogs/guide-to-diabetes-testing-methods-and-their-benefits

[5] V. Vaishnavi, R. Jha, and S. Mitra, “Comparative Study on Prediction of Diabetes Disease Using Various Machine Learning Models,” Advances in Intelligent Systems Research, pp. 285–300, 2025, doi: https://doi.org/10.2991/978-94-6463-787-8_24.

[6] M. Branca and M. Branca, “CGM-Based method could make early diabetes detection easier and cheaper,” Inside Precision Medicine, Apr. 22, 2025. https://www.insideprecisionmedicine.com/topics/precision-medicine/cgm-based-method-could-make-early-diabetes-detection-easier-and-cheaper/

[7] “The future of diabetes,” Department of MedicineNews. https://med.stanford.edu/medicine/news/current-news/standard-news/how-ai-and-glucose-monitors-could-transform-early-detection.html

[8] A. Sharma, “Machine learning in healthcare: applications, benefits & future trends,” Feb. 19, 2025. https://www.turing.com/resources/machine-learning-for-healthcare

[9] M. Hoche, O. Mineeva, G. Rätsch, E. Vayena, and A. Blasimme, “What makes clinical machine learning fair? A practical ethics framework,” PLOS Digital Health, vol. 4, no. 3, p. e0000728, Mar. 2025, doi: 10.1371/journal.pdig.0000728.

[10] M. Tuhin and M. Tuhin, “How Machine Learning is Transforming Healthcare,” Science News Today, Jul. 13, 2025. https://www.sciencenewstoday.org/how-machine-learning-is-transforming-healthcare

[11] B. D. Shivahare et al., “Delving into Machine Learning’s Influence on Disease Diagnosis and Prediction,” The Open Public Health Journal, vol. 17, no. 1, Apr. 2024, doi: 10.2174/0118749445297804240401061128.

[12] mohanedmashaly, “knn_classifier vs Logistic Regression,” *Kaggle.com*, Sep. 20, 2020. https://www.kaggle.com/code/mohanedmashaly/knn-classifier-vs-logistic-regression (accessed Aug. 31, 2025).

[13] alperaydoan, “Pima Indians Diabetes Database with RandomForest,” *Kaggle.com*, Aug. 27, 2025. https://www.kaggle.com/code/alperaydoan/pima-indians-diabetes-database-with-randomforest (accessed Aug. 31, 2025).

[14] paultimothymooney, “Predict Diabetes From Medical Records,” *Kaggle.com*, May 13, 2018. https://www.kaggle.com/code/paultimothymooney/predict-diabetes-from-medical-records

[15] mshirlaw, “Pima Indians Diabetes (Simple Logistic Regression),” *Kaggle.com*, Dec. 28, 2017. https://www.kaggle.com/code/mshirlaw/pima-indians-diabetes-simple-logistic-regression (accessed Sep. 04, 2025).

[16] UCI Machine Learning, “Pima Indians Diabetes Database,” *Kaggle.com*, 2016. https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database/code?datasetId=228&searchQuery=lo (accessed Aug. 31, 2025).