Texas A&M University - Commerce

Department of Computer Science

Performance Analysis of LR and SVM - Supervised Machine Learning Algorithms for Diabetes Prediction

Vyshnavi Sanikommu

Supervisor: Derek Harter, Ph.D.

A report submitted in partial fulfilment of the requirements of
Texas A&M University - Commerce for the degree of
Master of Science in *Computer Science*

March 2, 2024

Declaration

I, Vyshnavi Sanikommu, of the Department of Computer Science, Texas A&M University - Commerce, confirm that this is my own work and figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

I give consent to a copy of my report being shared with future students as an exemplar.

I give consent for my work to be made available more widely to members of TAMUC and public with interest in teaching, learning and research.

Vyshnavi Sanikommu
March 2, 2024

Abstract

Diabetes is one of the most lethal diseases affecting 537 million people worldwide, is projected to rise to 783 million by 2045. Diabetes is a disease caused due to an increase in blood glucose level, causing symptoms like frequent urination, increased hunger, and thirst. Diabetes is a leading cause of blindness, kidney failure, amputations, heart failure and stroke. The body's conversion of food into glucose requires insulin, released by the pancreas, which unlocks cells for glucose entry. This process allows cells to use glucose as an energy source, supporting vital bodily functions. The aim of this project is to develop a system which can perform early prediction of diabetes for a patient, employing supervised machine learning algorithms such as logistic regression and support vector machine. Machine learning techniques provide better results for prediction by constructing models from datasets collected from patients. The project will focus on optimizing performance metrics through a systematic search and tuning of model hyper parameters to maximize overall model effectiveness. The evaluation of model performance will encompass accuracy, precision, recall, f1-score, and confusion matrix. We will compare the performance metrics of the base model with those obtained after tuning using grid search. The results indicate that grid search provides optimal hyper parameters, contributing to the determination of the best-performing model.

Keywords: diabetes, machine learning, logistic regression, support vector machine, performance metrics

Acknowledgements

An acknowledgements section is optional. You may like to acknowledge the support and help of your supervisor(s), friends, or any other person(s), department(s), institute(s), etc. If you have been provided specific facility from department/school acknowledged so.

Contents

List of Figures

List of Tables

List of Abbreviations

SMPCS	School of Mathematical, Physical and Computational Sciences

Chapter 1

Introduction

Diabetes is a fast growing disease among the people even in youngsters nowadays. It is necessary to understand how it develops in our body. Firstly, we need to understand how a body works without diabetes. Sugar comes from the food that we eat, specially carbohydrates. When we eat this food, the body breaks them down to Sugars or Glucose. This glucose moves around the body in the bloodstream. This glucose is needed by body parts like the brain and pancreas to function. The remainder of glucose is taken to the cells of our body and liver which is stored as energy for later use. In order to use glucose for our body, insulin is required which is a hormone generated by pancreas. Insulin is a key to a closed door which helps glucose moves from blood stream. If pancreas is not able to produce enough insulin or if our body cannot use insulin it produces then glucose levels increases in bloodstream which leads to diabetes.

According to World Health Organization, diabetes is major cause of death worldwide. Around 422 million people worldwide have diabetes. Indeed, it caused deaths of 2 million people in 2019.

There are two types of diabetes present as a disease in human beings: **Type 1** diabetes appear most often during childhood and is characterized by the partial functioning of the pancreas. The cells fail to produce sufficient amounts of Insulin. Initially, we do not see any symptoms as the pancreas remains partially functional. There is no proven study and known methods for prevention. **Type 2** diabetes affects how the body uses sugars for energy. The cells produce low quantity of insulin or the body stops using insulin which can lead to high levels of sugar in bloodstream. High levels of glucose in the bloodstream and urine referred as diabetes mellitus. It is most common type of diabetes found in many people. It is caused by genetic factors and the lifestyle. It affects older adults and more obese or overweight people.

Early prediciton of diabetes can help in controlling the disease and potentially save lives. To accomplish this, this work explores prediction of diabetes by taking various attributes related to diabetes disease. For this purpose, we use the diabetes dataset (Johndasilva, 2018) which has 2000 instances with 9 attributes - 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome' for creating classification models using logistic regression and support vector machine algorithms.

1.1 Background

This study is undertaken to address the crucial need for accurate and reliable methods in predicting diabetes using logistic regression and support vector machine algorithms on patient data. The

motivation behind this study stems from the growing significance of leveraging machine learning algorithms to assist medical professionals in diagnosing and managing diabetes. Predictive models based on patient data offer the potential to identify individuals at risk or in the early stages of diabetes, enabling proactive and personalized healthcare strategies. Algorithms - logistic regression and support vector machine are chosen due to their widespread use in medical data analysis and their capability to handle classification tasks. The study outcomes have practical implications for healthcare practitioners and researchers, offering insights into algorithm selection for accurate and interpret diabetes prediction.

1.2 Research question

The research questions are:

1. How do the performance metrics of Logistic Regression (LR) and Support Vector Machines (SVM) models differ in predicting diabetes based on patient's data?
2. What factors contribute to these differences, particularly in relation to the hyper parameters?
3. How do various meta parameter searches contribute to get the best performance from these two models?

The research aims to compare the predictive performance metrics of logistic regression and support vector machine models for diabetes prediction based on patient data. It includes an in-depth analysis of the factors influencing the differences in performance metrics, with a specific focus on hyper parameters. Additionally, the study explores the impact of various meta-parameter searches on optimizing the overall performance of the models, aiming to identify the most effective parameters for diabetes prediction.

1.3 Aims and objectives

The primary aim of this study is to assess and compare the predictive performance metrics of logistic regression and support vector machine algorithms based on diabetic patient's data to decide if a patient is diabetic or not. This study aims to provide valuable insights into the strengths and limitations of these supervised machine learning approaches, contributing to the enhancement of diabetes prediction methodologies.

The main objectives of this study include:

- Obtain the model data, clean and analyze it and prepare it for model training.
- Train a standard logistic regression classification model on the labeled diabetes dataset.
- Train a competing SVM model on the same data.
- Explore model meta parameter and tune models to maximize predictive performance on accuracy, precision, recall, f1-score and confusion matrix.
- Evaluate the results of performance metrics after hyper parameter tuning to determine the best model for diabetes prediction.

1.4 Solution approach

The study follows a systematic approach encompassing model implementation, data preprocessing, splitting of data, hyper parameter tuning and performance analysis. Thorough exploration of various meta parameter search strategies contribute to achieving the aims. I will make sure to provide a clear documentation to make sure results are reliable and can be easily reproducible.

1.4.1 Dataset Description

This Diabetes Dataset Johndasilva (2018) has 2000 instances with 9 attributes - 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'. The 'Outcome' attribute indicates positive or negative for diabetes.

1.4.2 Data Preprocessing

Data preprocessing is most important process. Mostly healthcare related data contains missing values that causes effectiveness of data. This process is essential for accurate result and successful predication using machine learning techniques.

Missing values removal

This process is meant to identify instances with zero value and eliminate all such instances. Through eliminating irrelevant instances we make feature subset and this process is called feature subset selection which reduces the dimensionality of data.

Splitting of data

After cleaning the data, the data is normalized in training and testing the models. After split, we train algorithm with training dataset and keep testing dataset aside. This testing dataset is used to test the trained model.

1.4.3 Model Implementation

After data is ready, we apply machine learning techniques. Implement logistic regression and support vector machine models to predict diabetes on the dataset.

Logistic Regression

Logistic Regression is one of the most common classification models. It is used for classification task where the goal is to predict the probability that an instance belongs to a given class or not. Create a standard logistic regression classification model, train the model with training dataset, and calculate the performance metrics. Similarly calculate the performance metrics for testing dataset. For this classifier, there are multiple hyper parameters such as regularization strength (C), solver, penalty.

Support Vector Machine

Support Vector Machine is a powerful machine learning algorithm used for linear or nonlinear classification, regression, and outlier detection tasks. This classifier aims to establish a hyperplane that can separate the classes by adjusting the distance between data points and the hyperplane.

1.4.4 Hyper Parameter Tuning and Performance Analysis

To determine the best performance model, we should consider the factors like hyper parameters, meta parameter search strategies for logistic regression and support vector machine to achieve more predictive performance metrics.

Grid Search

Grid Search is a method for hyper parameter optimization that involves specifying a list of values for each hyper parameter to optimize. Subsequently, the model is trained for each combination of these values, and the optimal values for the hyper parameters are selected based on the models' performance. For logistic regression classifier, there are multiple hyper parameters such as regularization strength (C), solver, penalty. For support vector machine classifier, there are multiple hyper parameters such as regularization parameter (C), kernel, gamma, degree.

1.5 Summary of contributions and achievements

Describe clearly what you have done/created/achieved and what the major results and their implications are.

Chapter 2

Literature Review

The examination of relevant studies yields findings from various healthcare datasets, where researchers conducted analyses and predictions employing a range of methods and techniques. Numerous prediction models have been devised and applied by various researchers, utilizing different forms of data mining techniques, machine learning algorithms, or a combination of these methodologies.

2.1 Review of state-of-the-art

Mujumdar and Vaidehi (2019) aims to create a system using machine learning algorithms and deep learning techniques to provide accurate results and reduce human efforts. The diabetes dataset contains 800 instances with 10 attributes. This study implemented various machine learning algorithms include Support Vector Classifier, Random Forest Classifier, Decision Tree Classifier, Extra Tree Classifier, Ada Boost algorithm, Perceptron, Linear Discriminant Analysis algorithm, Logistic Regression, K-Nearest Neighbour, Gaussian Naïve Bayes,Bagging algorithm, Gradient Boost Classifier. The study incorporates the concept of pipelining and compares the diabetes dataset with the Pima dataset. Performance analysis includes metrics like classification accuracy, confusion matrix, f1-score, precision, and recall. The findings reveal that Logistic Regression achieves the highest accuracy of 96%, indicating an improvement in accuracy for the diabetes dataset compared to the Pima diabetes dataset. The study concludes that implementing a pipeline model enhances the accuracy of the classification performance, with the Ada Booster classifier identified as the best model, achieving an accuracy of 98.8%.

Soni and Varma (2020) aims to design and implement Diabetes prediction using machine learning methods and performance analysis of that methods for early prediction and to cure diabetes and save humans life. The diabetes dataset is gathered from UCI repository which is named as Pima Indian Diabetes dataset. The dataset have many attributes of 768 patients. The proposed methodology involves the utilization of various classification and ensemble learning methods, including SVM, Logistic Regression, KNN, Rndom Forest, Decision Tree, Gradient Boosting classifiers are used. The findings indicate a 77% accuracy achieved through an 80:20 split. The study concludes that Random Forest classifier exhibits highest accuracy when compared to other machine learning methods.

Swapna et al. (2018) aims to develop a methodology for classification of diabetic and normal Heart rate variability (HRV) signals using advanced deep learning architectures, specifically

employing Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), and combinations. The extracted features are then passed into a Support Vector Machine (SVM) for accurate classification. The study demonstrates performance improvements in CNN and CNN-LSTM architectures compared to earlier work, achieving a high accuracy of 95.7%.

2.2 Critique of the review

The literature review provides insights related to diabetes prediction using machine learning and deep learning techniques. Each study aims to contribute to the development of reliable and accurate models for diabetes diagnosis, with a focus on enhancing classification performance. Mujumdar and Vaidehi (2019) focus on creating an efficient system, achieving a 96% accuracy with Logistic Regression and identifying Ada Booster as the best model. Soni and Varma (2020) design a predictive system, achieving 77% accuracy with Random Forest being the most accurate. Swapna et al. (2018) explore advanced deep learning architectures, obtaining a high accuracy of 95.7% with CNN-LSTM and SVM. The key findings include the effectiveness of pipeling models, ensemble methods and the importance of accurate diabetes prediction for early intervention and patient care. However, the identified studies exhibit some limitations, such as a lack of detailed explanations regarding the selection of specific algorithms in the ensemble and a deficiency in exploring deep learning techniques that could enhance performance. Additionally, there is limited discussion on the generalizability of the proposed methodology to diverse datasets.Future research directions involve anomaly prediction and the utilization of larger datasets.

2.3 Summary

This literature review extensively examines the current research landscape in the realm of diabetes prediction, highlighting numerous technological advancements in this field. The literature review thoroughly investigates three prominent studies in diabetes prediction using machine learning and deep learning. Mujumdar and Vaidehi (2019) employ innovative pipelining techniques for accuracy enhancement but lack in-depth exploration of neural networks. Soni and Varma (2020) focus on diverse classifiers without extensive rationale, neglecting potential gains from deep learning. Swapna et al. (2018) excel in deep learning, emphasizing the need for a unified approach and comprehensive evaluation. The critiques call for collaboration, integration of machine learning strengths, and standardized methodologies for future research in diabetes prediction.

Chapter 3

Methodology

3.1 Dataset Description and Data Exploration

The diabetes data set is originated from kaggle (Johndasilva, 2018). The dataset contains 2000 patients and their corresponding 9 unique attributes. The nine attributes that are used for the prediction of diabetes are 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'. The 'Outcome' attribute is taken as a dependent or target variable, and the remaining eight attributes are taken as independent feature variables. The diabetes attribute 'Outcome' consists of binary value where 0 means non-diabetes, and 1 implies diabetes.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               2000 non-null   int64
 1   Glucose                   2000 non-null   int64
 2   BloodPressure             2000 non-null   int64
 3   SkinThickness             2000 non-null   int64
 4   Insulin                   2000 non-null   int64
 5   BMI                       2000 non-null   float64
 6   DiabetesPedigreeFunction  2000 non-null   float64
 7   Age                       2000 non-null   int64
 8   Outcome                   2000 non-null   int64
dtypes: float64(2), int64(7)
memory usage: 140.8 KB
```

Figure 3.1: Dataset attributes and their data types.

Here, all the attributes in Fig 3.1 are numeric and there are no null values.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	2	138	62	35	0	33.6	0.127	47	1
1	0	84	82	31	125	38.2	0.233	23	0
2	0	145	0	0	0	44.2	0.630	31	1
3	0	135	68	42	250	42.3	0.365	24	1
4	1	139	62	41	480	40.7	0.536	21	0

Figure 3.2: Top 5 patients data.

3.1.1 Correlation Matrix

A correlation matrix is a powerful tool for data analysis. It is a statistical technique used to evaluate the relationship between two variables in a dataset. It provides a correlation coefficient for each cell. The correlation coefficient value remains in the range between -1 and 1.

The correlation coefficient value -1 indicates notable negative linear correlation. The coefficient value 1 indicates notable positive linear correlation. The coefficient value 0 indicates no linear correlation.

A correlation matrix helps summarize data, identify patterns, and make decisions based on relationships between attributes. It helps us to gain insights for building better machine learning models by understanding which attributes are correlated.

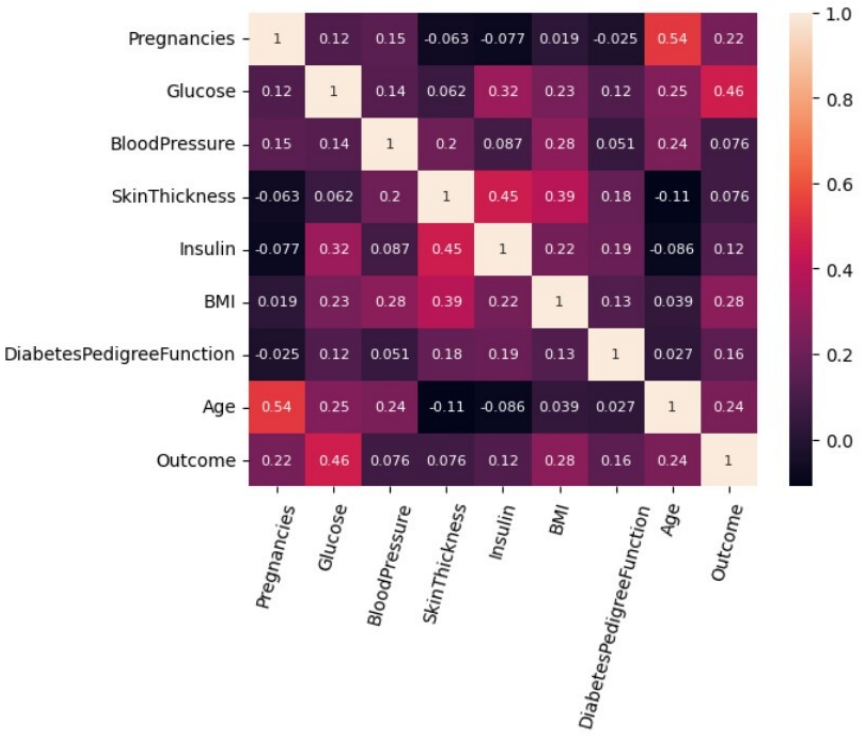

Figure 3.3: Correlation Matrix

3.1.2 Data Distribution using Histograms

Fig 3.4 gives a better feel than the raw numbers and percentiles of the distributions of our numerical attributes. It shows how each feature and label is distributed along different ranges which further confirms the need for scaling. Next, wherever you see discrete bars, it basically means that each of these is actually a categorical variable. We will need to handle these categorical variables before applying Machine Learning. Our outcome labels have two classes, 0 for no disease and 1 for disease.

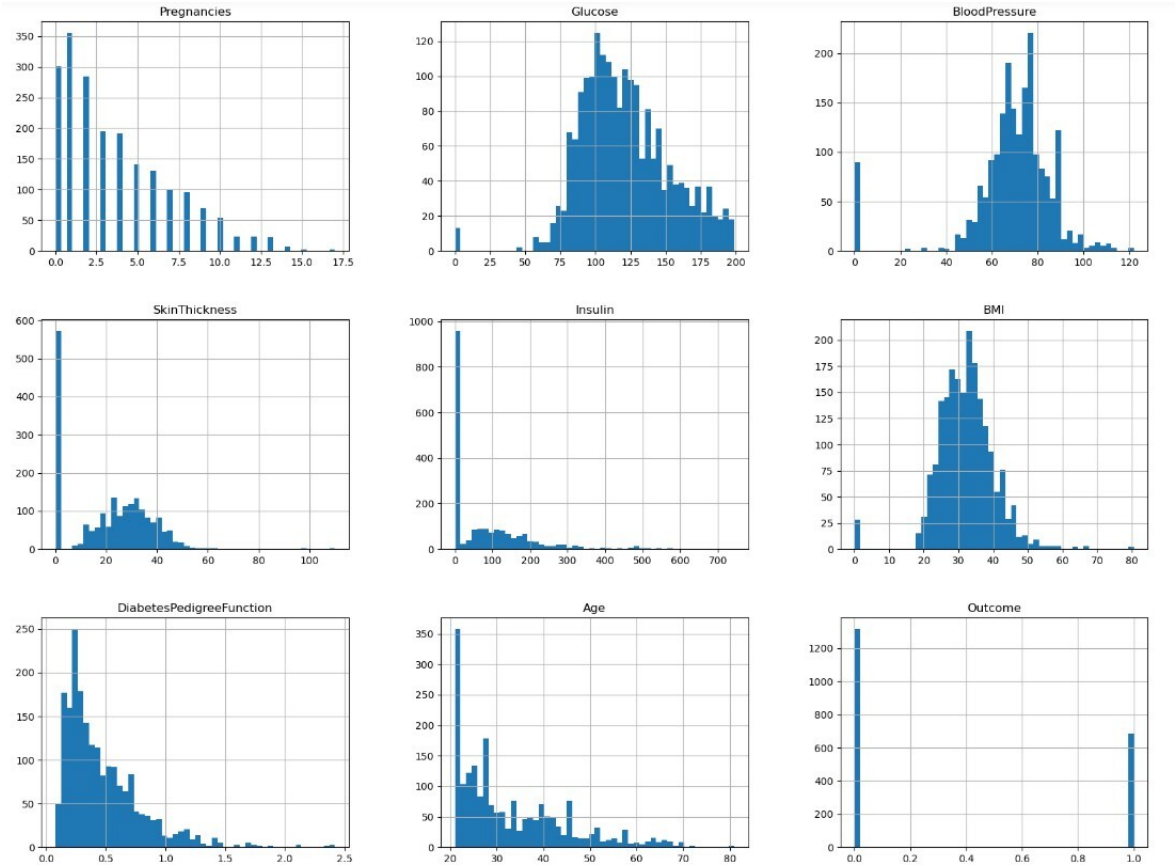

Figure 3.4: Data Distribution for each attribute

3.1.3 Bar plot for Outcome class

Fig 3.5 shows that the data is biased towards data points having outcome value as 0 which means that diabetes was not present actually. The number of non-diabetics is almost twice the number of diabetic patients.

Out of the 2000 instances, 1316 are associated with non-diabetic patients, while the remaining 684 pertain to diabetic patients.

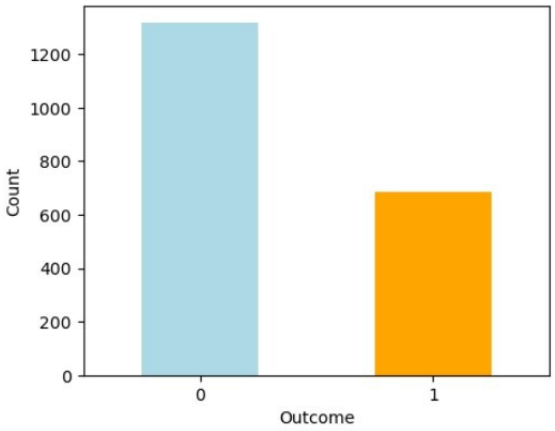

Figure 3.5: Distribution of Outcome (0s and 1s)

3.2 Data Preprocessing

Data preprocessing helps to transform data used to built a model which gives higher performance metrics. This process performs various functions like handling missing values, normalization and feature selection to improve the quality of data.

3.2.1 Missing Values Identification

Fig 3.1 indicates the absence of null values for all attributes. However, Fig 3.2 illustrates instances of zero values for attributes which are irrelevant and need to handled. Table 3.1 shows the number of zero values for each attribute.

Table 3.1: The number of zero missing values in dataset

Attributes	No.of missing values (Zero)
Pregnancies	301
Glucose	13
BloodPressure	90
SkinThickness	573
Insulin	956
BMI	28
DiabetesPedigreeFunction	0
Age	0

Here, we have observed numerous attributes with zero values, impacting the data quality. We replaced the zero values with the corresponding mean values using simpleimputer.

Fig 3.2 and Fig 3.6 shows the top five patients data. If you compare these figures, Fig 3.6 indicates that zero values in Fig 3.2 was replaced with mean for each attribute. Now, there are no missing values either null or zero values in each attribute.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	2.000000	138.0	62.000000	35.000000	153.743295	33.6	0.127	47.0
1	4.359623	84.0	82.000000	31.000000	125.000000	38.2	0.233	23.0
2	4.359623	145.0	72.403665	29.341275	153.743295	44.2	0.630	31.0
3	4.359623	135.0	68.000000	42.000000	250.000000	42.3	0.365	24.0
4	1.000000	139.0	62.000000	41.000000	480.000000	40.7	0.536	21.0

Figure 3.6: Top 5 patients data after handling missing values

3.2.2 Feature Selection based on Correlation Coefficient

After handling missing values, the correlation coefficient is calculated in this method which correlates with the output and input attributes. This was discussed under section 3.1.1. We have created a Table 3.2 to show correlation coefficient values of each attribute towards target attribute.

Table 3.2: The correlation coefficient values

Attributes	Correlation Coefficient(Previous)	Correlation Coefficient(Current)
Pregnancies	0.22	0.25
Glucose	0.46	0.49
BloodPressure	0.076	0.17
SkinThickness	0.076	0.21
Insulin	0.12	0.21
BMI	0.28	0.28
DiabetesPedigreeFunction	0.16	0.16
Age	0.24	0.24

We used 0.2 as a cut-off for relevant attributes. Hence ‘BloodPressure‘ and ‘DiabetesPedigreeFunction‘ features are removed. ‘Pregnancies‘, ‘Glucose‘, ‘SkinThickness‘, ‘Insulin‘, ‘BMI‘, and ‘Age‘ are our most relevant six input attributes.

3.2.3 Data Normalization

Normalization refers to the process of scaling and transforming numeric features to a standard scale or distribution. Feature scaling is a technique to standardize or normalize the range of independent features or variables of a dataset. The goal of feature scaling is to ensure that all features contribute equally to the learning process, preventing certain features from dominating others based on their scale. In Table 3.2, we can see that ‘Glucose‘ and ‘Outcome‘ have a 0.49 correlation coefficient. Hence these are highly correlated. After completing data preprocessing, we have total 2000 instances.

3.3 Dataset Split into Train and Test Data

After data cleaning and preprocessing, the dataset becomes ready to train and test. We are using test split method to split data randomly into the training and testing set. Here, I am partitioning

the data into a training set comprising 70% and a test set comprising 30%.

3.4 Model Implementation

This is the important phase which includes model building for diabetes prediction. In this, we have implemented logistic regression and support vector machine algorithms.

3.4.1 Logistic Regression

Logistic Regression is one of the most common classification models. It is used for classification task where the goal is to predict the probability that an instance belongs to a given class or not.

Algorithm 1 Diabetes Prediction using Logistic Regression

Input: Input features X and labels y for training data
Output: Generate performance metrics like accuracy, confusion matrix, precision, recall, f1-score

1: Create a standard logistic regression model with default hyperparameters such as regularization strength (C), solver, penalty
2: Fit the model with training data
3: Calculate accuracy, confusion matrix, precision, recall and f1-score for the trained model
4: Predict outcomes for testing data using the trained model
5: Evaluate the model performance on testing data
6: Calculate accuracy, confusion matrix, precision, recall and f1-score

3.4.2 Support Vector Machine

Support Vector Machine is used for linear or nonlinear classification, regression, and outlier detection tasks. This classifier aims to establish a hyperplane that can separate the classes by adjusting the distance between data points and the hyperplane.

Algorithm 2 Diabetes Prediction using Support Vector Machine

Input: Input features X and labels y for training data
Output: Generate performance metrics like accuracy, confusion matrix, precision, recall, f1-score

1: Create a standard svm model with default hyperparameters such as regularization parameter (C), kernel, gamma, degree
2: Fit the model with training data
3: Calculate accuracy, confusion matrix, precision, recall and f1-score for the trained model
4: Predict outcomes for testing data using the trained model
5: Evaluate the model performance on testing data
6: Calculate accuracy, confusion matrix, precision, recall and f1-score

3.4.3 Hyperparameter Tuning with GridSearchCV

Hyperparameter Tuning is a process to select optimal values for a machine learning models hyperparameters. The goal of hyperparameter tuning is to find the values that leads to the best performance for a given problem. We should consider the factors like hyper parameters, meta parameter search strategies to achieve more predictive performance metrics.

Grid Search

Grid Search is a method for hyper parameter optimization that involves specifying a list of values for each hyper parameter to optimize. Subsequently, the model is trained for each combination of these values, and the optimal values for the hyper parameters are selected based on the models' performance.

Hyperparameter Tuning for Logistic Regression

Logistic Regression is one of the most common classification algorithms. It is used for classification task where the goal is to predict the probability that an instance belongs to a given class or not. There are multiple hyper parameters such as regularization strength (C), solver, penalty. We have solvers namely, 'lbfgs', 'newton-cg', 'liblinear', 'sag', 'saga'. We have penalties namely, 'l1', 'l2', 'elasticnet', 'none'.

Algorithm 3 Diabetes Prediction using hyperparameter tuning technique for Logistic Regression

Input: Input features X and labels y for training data

Output: Generate best parameters and calculate performance metrics like accuracy, confusion matrix, precision, recall, f1-score using best parameters

1: Create a standard logistic regression model with no hyperparameters such as regularization strength (C), solver, penalty
2: Create a paramgrid dictionary with all hyperparameters related to logistic regression model you would like to pass to gridsearch for tuning
3: Create a gridsearchcv model by passing parameters such as model estimator, paramgrid, cross validation, scoring methods, refit, verbose
4: Generate the best estimators, best parameters and best scores for training set
5: Calculate accuracy, confusion matrix, precision, recall and f1-score for trained model
6: Predict outcomes for testing data using the trained model
7: Evaluate the model performance on testing data
8: Calculate accuracy, confusion matrix, precision, recall and f1-score

Hyperparameter Tuning for Support Vector Machine

This classifier aims at forming a hyper plane that can separate the classes as much as possible by adjusting the distance between the data points and the hyper plane. There are several kernels based on which the hyper plane is decided. There are multiple hyper parameters such as regularization parameter (C), kernel, gamma, degree. We have kernels namely, 'linear', 'poly', 'rbf', and 'sigmoid'.

Algorithm 4 Diabetes Prediction using hyperparameter tuning technique for Support Vector Machine

Input: Input features X and labels y for training data
Output: Generate performance metrics like accuracy, confusion matrix, precision, recall, f1-score

1: Create a standard svm model with no hyperparameters such as regularization parameter (C), kernel, gamma, degree
2: Create a paramgrid dictionary with all hyperparameters related to svm model you would like to pass to gridsearch for tuning
3: Create a gridsearchcv model by passing parameters such as model estimator, paramgrid, cross validation, scoring methods, refit, verbose
4: Generate the best estimators, best parameters and best scores for training set
5: Calculate accuracy, confusion matrix, precision, recall and f1-score for trained model
6: Predict outcomes for testing data using the trained model
7: Evaluate the model performance on testing data
8: Calculate accuracy, confusion matrix, precision, recall and f1-score

3.5 Example of a Figure in LaTeX

Figure 3.7 is an example of a figure in LaTeX. For more details, check the link:
wikibooks.org/wiki/LaTeX/Floats,_Figures_and_Captions.
Keep your artwork (graphics, figures, illustrations) clean and readable. At least 300dpi is a good resolution of a PNG format artwork. However, an SVG format artwork saved as a PDF will produce the best quality graphics. There are numerous tools out there that can produce vector graphics and let you save that as an SVG file and/or as a PDF file. One example of such a tool is the "Flow algorithm software". Here is the link for that: flowgorithm.org.

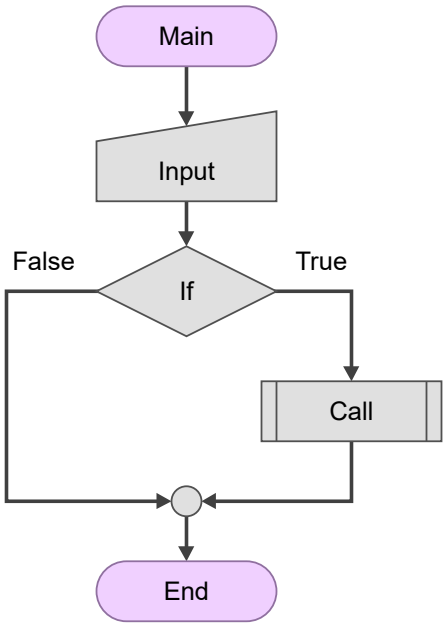

Figure 3.7: Example figure in LaTeX.

3.6 Example of an algorithm in LaTeX

Algorithm 5 is a good example of an algorithm in LaTeX.

Algorithm 5 Example caption: sum of all even numbers

Input: $\mathbf{x} = x_1, x_2, \ldots, x_N$
Output: $EvenSum$ (Sum of even numbers in $\mathbf{x}$)

1: **function** EvenSummation($\mathbf{x}$)
2: $EvenSum \leftarrow 0$
3: $N \leftarrow length(\mathbf{x})$
4: **for** $i \leftarrow 1$ to N **do**
5: **if** $x_i \mod 2 == 0$ **then** ▷ check if a number is even?
6: $EvenSum \leftarrow EvenSum + x_i$
7: **end if**
8: **end for**
9: **return** $EvenSum$
10: **end function**

3.7 Example of code snippet in LaTeX

Code Listing 3.1 is a good example of including a code snippet in a report. While using code snippets, take care of the following:

- do not paste your entire code (implementation) or everything you have coded. Add code snippets only.
- The algorithm shown in Algorithm 5 is usually preferred over code snippets in a technical/-scientific report.
- Make sure the entire code snippet or algorithm stays on a single page and does not overflow to another page(s).

Here are three examples of code snippets for three different languages (Python, Java, and CPP) illustrated in Listings 3.1, 3.2, and 3.3 respectively.

```python
import numpy as np

x  = [0, 1, 2, 3, 4, 5] # assign values to an array
evenSum = evenSummation(x) # call a function

def evenSummation(x):
    evenSum = 0
    n = len(x)
    for i in range(n):
        if np.mod(x[i],2) == 0: # check if a number is even?
            evenSum = evenSum + x[i]
    return evenSum
```

Listing 3.1: Code snippet in LaTeX and this is a Python code example

Here we used the "\clearpage" command and forced-out the second listing example onto the next page.

```java
public class EvenSum{
    public static int evenSummation(int[] x){
        int evenSum = 0;
        int n = x.length;
        for(int i = 0; i < n; i++){
            if(x[i]%2 == 0){ // check if a number is even?
                evenSum = evenSum + x[i];
            }
        }
        return evenSum;
    }
    public static void main(String[] args){
        int[] x  = {0, 1, 2, 3, 4, 5}; // assign values to an array
        int evenSum = evenSummation(x);
        System.out.println(evenSum);
    }
}
```

Listing 3.2: Code snippet in LaTeX and this is a Java code example

```cpp
int evenSummation(int x[]){
    int evenSum = 0;
    int n = sizeof(x);
    for(int i = 0; i < n; i++){
        if(x[i]%2 == 0){ // check if a number is even?
            evenSum = evenSum + x[i];
      }
    }
    return evenSum;
}

int main(){
    int x[]  = {0, 1, 2, 3, 4, 5}; // assign values to an array
    int evenSum = evenSummation(x);
    cout<<evenSum;
    return 0;
}
```

Listing 3.3: Code snippet in LaTeX and this is a C/C++ code example

3.8 Example of in-text citation style

3.8.1 Example of the equations and illustrations placement and reference in the text

Make sure whenever you refer to the equations, tables, figures, algorithms, and listings for the first time, they also appear (placed) somewhere on the same page or in the following page(s). Always make sure to refer to the equations, tables and figures used in the report. Do not leave them without an **in-text citation**. You can refer to equations, tables and figures more them once.

3.8.2 Example of the equations and illustrations style

Write **Eq.** with an uppercase "Eq" for an equation before using an equation number with (\eqref{.}). Use "Table" to refer to a table, "Figure" to refer to a figure, "Algorithm" to

refer to an algorithm and "Listing" to refer to listings (code snippets). Note that, we do not use the articles "a," "an," and "the" before the words Eq., Figure, Table, and Listing, but you may use an article for referring the words figure, table, etc. in general.

For example, the sentence "A report structure is shown in **the** Table should be written as "A report structure is shown **in**

3.9 Summary

Write a summary of this chapter.

Note: In the case of **software engineering** project a Chapter "**Testing and Validation**" should precede the "Results" chapter. See for report organization of such project.

Chapter 4

Results

The results chapter tells a reader about your findings based on the methodology you have used to solve the investigated problem. For example:

- If your project aims to develop a software/web application, the results may be the developed software/system/performance of the system, etc., obtained using a relevant methodological approach in software engineering.
- If your project aims to implement an algorithm for its analysis, the results may be the performance of the algorithm obtained using a relevant experiment design.
- If your project aims to solve some problems/research questions over a collected dataset, the results may be the findings obtained using the applied tools/algorithms/etc.

Arrange your results and findings in a logical sequence.

4.1 A section

...

4.2 Example of a Table in LaTeX

Table 4.1 is an example of a table created using the package LaTeX"booktabs." do check the link: wikibooks.org/wiki/LaTeX/Tables for more details. A table should be clean and readable. Unnecessary horizontal lines and vertical lines in tables make them unreadable and messy. The example in Table 4.1 uses a minimum number of liens (only necessary ones). Make sure that the top rule and bottom rule (top and bottom horizontal lines) of a table are present.

Table 4.1: Example of a table in LaTeX

Bike		
Type	Color	Price (£)
Electric	black	700
Hybrid	blue	500
Road	blue	300
Mountain	red	300
Folding	black	500

4.3 Example of captions style

- The **caption of a Figure (artwork) goes below** the artwork (Figure/Graphics/illustration). See example artwork in .
- The **caption of a Table goes above** the table. See the example in Table 4.1.
- The **caption of an Algorithm goes above** the algorithm. See the example in Algorithm 5.
- The **caption of a Listing goes below** the Listing (Code snippet). See example listing in Listing 3.1.

4.4 Summary

Write a summary of this chapter.

Chapter 5

Discussion and Analysis

Depending on the type of project you are doing, this chapter can be merged with "Results" Chapter as " Results and Discussion" as suggested by your supervisor.

In the case of software development and the standalone applications, describe the significance of the obtained results/performance of the system.

5.1 A section

Discussion and analysis chapter evaluates and analyses the results. It interprets the obtained results.

5.2 Significance of the findings

In this chapter, you should also try to discuss the significance of the results and key findings, in order to enhance the reader's understanding of the investigated problem

5.3 Limitations

Discuss the key limitations and potential implications or improvements of the findings.

5.4 Summary

Write a summary of this chapter.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Typically a conclusions chapter first summarizes the investigated problem and its aims and objectives. It summaries the critical/significant/major findings/results about the aims and objectives that have been obtained by applying the key methods/implementations/experiment set-ups. A conclusions chapter draws a picture/outline of your project's central and the most signification contributions and achievements.

A good conclusions summary could be approximately 300–500 words long, but this is just a recommendation.

A conclusions chapter followed by an abstract is the last things you write in your project report.

6.2 Future work

This section should refer to Chapter 4 where the author has reflected their criticality about their own solution. The future work is then sensibly proposed in this section.

Guidance on writing future work: While working on a project, you gain experience and learn the potential of your project and its future works. Discuss the future work of the project in technical terms. This has to be based on what has not been yet achieved in comparison to what you had initially planned and what you have learned from the project. Describe to a reader what future work(s) can be started from the things you have completed. This includes identifying what has not been achieved and what could be achieved.

A good future work summary could be approximately 300–500 words long, but this is just a recommendation.

Chapter 7

Reflection

Write a short paragraph on the substantial learning experience. This can include your decision-making approach in problem-solving.

Some hints: You obviously learned how to use different programming languages, write reports in LaTeXand use other technical tools. In this section, we are more interested in what you thought about the experience. Take some time to think and reflect on your individual project as an experience, rather than just a list of technical skills and knowledge. You may describe things you have learned from the research approach and strategy, the process of identifying and solving a problem, the process research inquiry, and the understanding of the impact of the project on your learning experience and future work.

Also think in terms of:

- what knowledge and skills you have developed
- what challenges you faced, but was not able to overcome
- what you could do this project differently if the same or similar problem would come
- rationalize the divisions from your initial planed aims and objectives.

A good reflective summary could be approximately 300–500 words long, but this is just a recommendation.

Note: The next chapter is "**References**," which will be automatically generated if you are using BibTeX referencing method. This template uses BibTeX referencing. Also, note that there is difference between "References" and "Bibliography." The list of "References" strictly only contain the list of articles, paper, and content you have cited (i.e., refereed) in the report. Whereas Bibliography is a list that contains the list of articles, paper, and content you have cited in the report plus the list of articles, paper, and content you have read in order to gain knowledge from. We recommend to use only the list of "References."

References

Johndasilva (2018), 'Diabetes dataset'. (accessed January 25, 2024).
URL:

Mujumdar, A. and Vaidehi, V. (2019), 'Diabetes prediction using machine learning algorithms', *Procedia Computer Science* **165**, 292–299.

Soni, M. and Varma, S. (2020), 'Diabetes prediction using machine learning techniques', *International Journal of Engineering Research & Technology (Ijert) Volume* **9**.

Swapna, G., Vinayakumar, R. and Soman, K. (2018), 'Diabetes detection using deep learning algorithms', *ICT express* **4**(4), 243–246.

Appendix A

An Appendix Chapter (Optional)

Some lengthy tables, codes, raw data, length proofs, etc. which are **very important but not essential part** of the project report goes into an Appendix. An appendix is something a reader would consult if he/she needs extra information and a more comprehensive understating of the report. Also, note that you should use one appendix for one idea.

An appendix is optional. If you feel you do not need to include an appendix in your report, avoid including it. Sometime including irrelevant and unnecessary materials in the Appendices may unreasonably increase the total number of pages in your report and distract the reader.

Appendix B

An Appendix Chapter (Optional)

...