A Machine Learning Approach to Predicting Diabetes Complications

A MACHINE LEARNING APPROACH

FOR PREDICTING DIABETES COMPLICATIONS

by

Yazan Khaled Jian

A Thesis presented to the Faculty of the
American University of Sharjah
College of Engineering
In Partial Fulfillment
of the Requirements
for the Degree of

Master of Science in
Computer Engineering

Sharjah, United Arab Emirates

December 2021

Declaration of Authorship

I declare that this thesis is my own work and, to the best of my knowledge and belief, it does not contain material published or written by a third party, except where permission has been obtained and/or appropriately cited through full and accurate referencing.

Signature…Yazan Khaled Jian.………………………………….

Date……05/12/2021……………………………….…………

Approval Signatures

We, the undersigned, approve the Master's Thesis of Yazan Jian

Thesis Title: A Machine Learning Approach to Predicting Diabetes Complications

Date of Defense: 29/11/2021

Name, Title and Affiliation	**Signature**
Dr. Assim Sagahyroon Professor, Department of Computer Science and Engineering Thesis Advisor	
Dr. Fadi Aloul Professor, Department of Computer Science and Engineering Thesis Co-Advisor	
Dr. Michel Pasquier Associate Professor, Department of Computer Science and Engineering Thesis Co-Advisor	
Dr. Salam Dhou Associate Professor, Department of Computer Science and Engineering Thesis Committee Member	
Dr. Usman Tariq Associate Professor, Department of Electrical Engineering Thesis Committee Member	
Dr. Fadi Aloul Head Department of Computer Science and Engineering	
Dr. Lotfi Romdhane Associate Dean for Graduate Affairs and Research College of Engineering	
Dr. Sameer Al-Asheh Interim Dean College of Engineering	
Dr. Mohamed El-Tarhuni Vice Provost for Research and Graduate Studies	

Acknowledgements

I would like to give my warmest thanks to my advisors Dr. Assim Sagahyroon, Dr. Fadi Aloul and Dr. Michel Pasquier for trusting my work, providing knowledge, guidance, support, and motivation throughout the different phases of this scientific work. I really appreciate their assistance and I'm deeply beholden for their great discussion and suggestions.

I would also like to thank my family, for their continuous love and support. Thank you for always encouraging me and believing in me. Special thanks to my brother, thank you for always being there for me.

I would like also to thank the American University of Sharjah and the Department of Computer Science and Engineering for providing me with such a great opportunity and assistantship.

Dedication

To my family…

Abstract

Machine learning and data mining techniques have been widely used over the years to extract knowledge from data. The goal of this thesis is to study several diabetes complications. Diabetes Mellitus (DM) is a chronic disease that is considered to be life threatening. It can affect any part of the body over time resulting in more serious complications such as impacts on eyesight, perception, motor control and more. To study diabetes complications, a dataset collected by the Rashid Centre for Diabetes and Research (RCDR) located in Ajman, UAE was utilized. The dataset consists of 884 records with 79 features and 8 complications. The complications' set contains metabolic syndrome, dyslipidemia, neuropathy, nephropathy, diabetic foot, hypertension, obesity, and retinopathy. Some essential preprocessing steps were needed to handle the missing values and imbalanced data problems. Moreover, several techniques were used to study the problem in hand. The first part of this thesis focused on generating association rules from the dataset using unsupervised learning techniques. This step was essential to extract valuable knowledge and relations between several attributes in the dataset and helped to develop a better understanding of DM and its complications. For instance, we extracted several rules indicating some possible relations between metabolic syndrome, hypertension and dyslipidemia. Further preprocessing steps were needed such as data discretization. For the second part of the research, different supervised classification algorithms were utilized to build several models to predict and diagnose eight diabetes complications. Furthermore, feature selection was performed to select the top 5 and 10 features for each complication. Repeated stratified k-fold cross validation was employed for a better estimation of the performance with a k=10 and a total of 10 repetitions. Accuracy and F1-score were used to evaluate the models' performance reaching a maximum of 97.8% and 97.7% for accuracy and F1-scores, respectively.

Keywords: Diabetes Prediction, Diabetes Complications, Supervised Learning, Association Rule Mining.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ANN	Artificial Neural Network
AUC	Area Under the Curve
ARM	Association Rule Mining
CI	Confidence Interval
DM	Diabetes Mellitus
ID3	Iterative Decision Tree
KCV	k-fold Cross Validation
k-NN	*k*-Nearest Neighbours
LR	Logistic Regression
ML	Machine Learning
NB	Naïve Bayes
PIDD	PIMA Indians Diabetes Dataset
RF	Random Forest
RMSE	Root Mean Squared Error
RMSEN	Normalized Root Mean Squared Error
ROC	Receiver Operating Characteristics
RSCV	Repeated Stratified Cross Validation
T2DM	Type 2 Diabetes Mellitus

Chapter 1. Introduction

Chronic diseases are defined broadly as conditions that last 1 year or more and require ongoing medical attention or limit activities of daily living or both [1]. Diabetes is one major example of such chronic diseases.

Diabetes mellitus, or diabetes for short, is a disease that occurs either when the pancreas does not produce enough insulin or when the body cannot effectively use the insulin it produces [2]. Diabetes has two main types called Type 1 and Type 2. In Type 1, also known as insulin-dependent or childhood-onset, there is a deficiency in producing insulin in the body that requires daily administration of insulin, whereas in diabetes Type 2, known formally as non-insulin-dependent or adult-onset, the body cannot use the insulin effectively. It is worth mentioning that diabetes Type 2 is the most common type in patients [2].

According to the World Health Organization (WHO), the number of people with diabetes in 2014 was 422 million. Moreover, in 2016, diabetes was the direct cause of 1.6 million deaths [2].

There are different risk factors for diabetes, especially diabetes Type 2. For instance, age, family history of diabetes, high blood pressure, high levels of triglycerides, heart disease or stroke are all considered as risk factors for diabetes [3]. As mentioned by The Centers for Disease Control and Prevention (CDC) [4], diabetes can affect any part of the body over time. For example, diabetes can lead to different complications such as metabolic syndrome (a cluster of metabolic disorders), dyslipidemia (an abnormal amount of lipids in the blood), neuropathy (nerve damage), nephropathy (disease of the kidneys) and much more. As a result, it is very important to understand how to deal with diabetes and especially how to prevent such possible complications.

To reduce the possibility of developing some serious complications related to diabetes, machine learning and data mining techniques can be applied on diabetes-related datasets. Machine learning is a branch of artificial intelligence and computer science which focuses on the use of data and algorithms to automatically learn and improve from experience. Machine learning itself can be divided into several categories including supervised, and unsupervised learning [5]. The main goal in both cases is to make use of a given dataset to enhance our understanding of the data and discover useful knowledge. Supervised machine learning is characterized by its use of labeled data to train its algorithms and can be utilized for classification or regression tasks. The goal of classification is to assign each unknown instance to one of possible classes or categories, for prediction or diagnosis purposes, such as classifying emails to either spam or not spam. In contrast with classification, the regression task deals with continuous data such as predicting the price of a home. On the other hand, unsupervised learning uses machine learning (ML) algorithms to analyze and cluster unlabeled datasets. The use of unsupervised learning targets the needs to discover and analyze hidden patterns or groups of the data such as customer segmentation [6].

The focus of this research is to make use of both supervised and unsupervised learning to study a set of complications related to diabetes. The complications' set contains metabolic syndrome, dyslipidemia, neuropathy, nephropathy, diabetic foot, hypertension, obesity, and retinopathy. For the unsupervised part, Association Rule Mining (ARM) techniques were utilized to discover possible hidden relations between the attributes represented in the dataset in hand. The techniques include both Apriori algorithm and FP-growth. For the supervised learning part, several classification algorithms were utilized to build different models that can predict and classify sick and healthy patients. The algorithms' set consists of Logistic regression (LR), Support Vector Machine (SVM), Decision Tree (DT CART), Random Forest (RF), AdaBoost and XGBoost. The performance of each algorithm was evaluated using its classification accuracy and F1-Score. The reason behind this selection of evaluation methods is clarified in section 5.1.2. Moreover, a feature selection step was carried out using the previously mentioned supervised models to help in understanding the importance of the features in the dataset. The top 5 and 10 features were selected and used to build focused models. Several experiments were also conducted to study the effect of choosing different preprocessing methods on the performance of the models.

Chapter 5 discusses several interesting findings. For instance, by the use of ARM we extracted large number of rules between metabolic syndrome, hypertension, dyslipidemia, and obesity. For example, metabolic syndrome and hypertension were found to be strongly connected with each other, meaning that if a patient develops hypertension, it is more likely to develop metabolic syndrome as well. Other relations were found in the dataset. For example, patients from UAE with Diabetes Type 2 are probably more likely to be at risk of having or developing dyslipidemia. For the supervised classification models, LR, RF, AdaBoost and XGBoost mainly achieved the best performance. Precisely, the best prediction performance was achieved by XGBoost to predict diabetic foot with a maximum accuracy of 97.8% and F1-Score of 97.7%. The feature selection step was also found to be useful with a slight degradation on the performance. Furthermore, Total cholesterol, Diabetes Age, Gender, BMI, and blood pressure were identified as the most dominant features in constructing the models.

The rest of this document is organized as follows: After the literature review in Chapter 2, a problem statement is presented in Chapter 3 to highlight the weaknesses and possible needs uncovered in the literature. In Chapter 4, the dataset and the research methodology are explained in detail. Chapter 5 presents the experimental results and their analysis. Finally, Chapter 6 concludes the document and discusses future work.

Chapter 2. Background and Literature Review

Data mining can be utilized in different sectors such as education, healthcare, business, and many other fields. The applications of data mining in healthcare enable disease diagnosis, prognosis, and a deep understanding of medical data [7], [8]. For instance, it may provide a better understanding of the correlation between different chronic diseases [5]such as Diabetes Mellitus (DM), which is a serious health problem and a cause of death. In this chapter, a description of the ML algorithms used in this study is introduced along with a discussion of some of the available studies related to diabetes.

2.1. Machine Learning Algorithms

In this section, the unsupervised and supervised ML algorithms used in this study are discussed and clarified including ARM and classification techniques.

2.1.1 Association rule mining

ARM is a class of algorithms that is used to explore and interpret large transactional datasets to identify unique patterns and rules [9]. The patterns identified by ARM define and represent different relationships and interactions between different items or features. Moreover, these identified relationships can sometimes lead to new discoveries and enhancements. ARM proved to be useful and effective in several applications such as in retail, online cross selling, recommendation engines, text analysis, document analysis, and web analysis [10].

One of the popular case studies for ARM is market basket analysis, where multiple rules can be extracted from a transaction data recorded by point-of-sale (POS) systems in supermarkets. For instance, one of the rules extracted by using ARM on such data indicates that men who buy diapers are also likely to buy beer. As noticed in this example, ARM helped to find a rule which seems to be fictional. Such information can be used as the basis for decisions about marketing activities such as promotional pricing or product placements.

Following the same approach explained above, ARM can exceed the given examples to cover other fields like extracting rules from medical health records. The improvement of hospital resources and the increase of using electronic devices to perform and store medical tests and records encourage the use of modern approaches such as ARM algorithms on the data collected. Utilizing such modern techniques in studying medical data may improve the understanding of the association between patient's diagnosis and diagnostic test requirement which can improve decision-making and efficient use of resources.

It is worth mentioning that ARM covers multiple algorithms that have the same goal of extracting different rules from a given dataset. Apriori algorithm is a well-known algorithm for rules extraction. It was proposed by Agrawal and Srikant in 1994 [11]. The algorithm proceeds by identifying the frequent individual items in the dataset and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the dataset. Apriori uses breadth-first search and a hash tree to count candidate item sets efficiently [11]. Moreover, the algorithm uses a given threshold C to identify the item sets which are subsets of at least C transactions in the dataset. Another ARM algorithm that is worth testing is FP-growth [12]. This algorithm can be considered as an improvement of the Apriori algorithm in which a frequent pattern is generated without the need for candidate generation. Furthermore, FP-growth represents the dataset in the form of a tree called a frequent pattern tree or FP tree, which maintains the association between the item sets [12]. As mentioned earlier, for the purpose of this research, both Apriori and FP-growth were utilized to extract useful medical rules from the dataset in hand and to help identify and discover the relations between the different medical tests' attributes in the dataset.

2.1.2 Classification and prediction

The dataset in hand consists of multiple attributes as well as consisting of eight different complications (or targets). Each record (patient) may suffer from one or more of these complications. For example, patient #1 suffers from metabolic syndrome, dyslipidemia, and neuropathy. Since we are dealing with a labeled data (eight labels in eight separate columns), some supervised machine learning techniques can be applied and evaluated to build models that can predict and classify the complications occurrence in patients. The rest of this section describes these algorithms and discusses the basic format as well as highlighting some advantages and drawbacks for each technique.

2.1.2.1. Logistic Regression

The first algorithm used is Logistic Regression classifier (or logit model). It is a supervised model that can be utilized to model the probability of a target class using independent features (input variables) [13]. The target variable in the case of LR is typically a categorical variable with a form of 0 or 1, true or false, etc. Figure 2-1 depicts an example of the LR model. In this study, diabetes complications are considered to be the targets. More precisely, each complication independently can be considered as a single target. Hence, 8 different LR models to classify all 8 complications were needed.

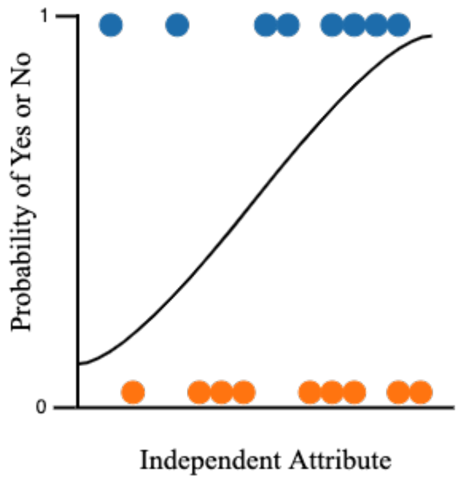


Figure 2-1: Logistic Regression Example

Equation (1) represents the general formula of LR, where $p(X)$ is the dependent variable, X is the independent variable, β_0 is the intercept, and β_1 is the slope coefficient. The algorithm calculates the probability of the target class by utilizing a simple yet effective linear equation which uses the intercept, and the slope coefficients, for the features in the dataset.

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X \quad (1)$$

Although the assumption of linearity between the dependent variable and the independent variables may not be correct in all cases, the simplicity of the algorithm as well as the proven effectiveness of LR in the literature make the logistic regression an attractive algorithm to test in this study.

2.1.2.2. Support Vector Machine

Another algorithm utilized is Support Vector Machine (SVM). It is a supervised algorithm used for both regression and classification problems. The objective of the SVM classifier is to find a hyperplane in an N-dimensional space (N - the number of features) that distinctly classifies the data points by deciding on which side of the plane they fall on. SVM in its default settings is a binary classifier that handles only two classes. As mentioned in [14], SVM can work as a linear classifier that attempts to find a linear hyperplane which separates the two classes. The algorithm tries to find the optimal separating hyperplane by maximizing the distance between the hyperplane and the critical points at the decision boundary, known as support vectors. Therefore, SVM's task is to try to maximize the distance between support vectors, shown as dotted lines. This results in a quadratic programming problem. Figure 2-2 shows an example of SVM.

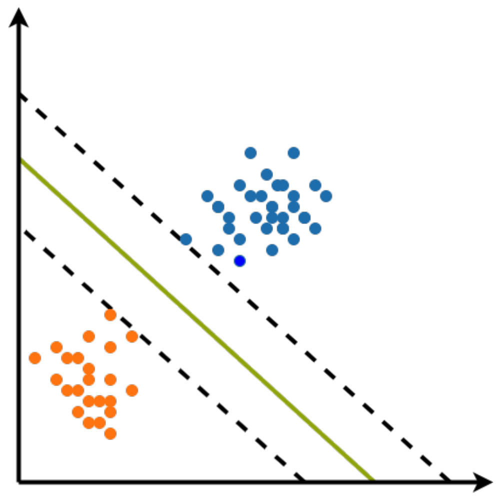


Figure 2-2: Linear SVM

SVM helps to reduce the complexity of the problem and to achieve better results compared to more complex algorithms in some cases. However, some datasets are non-linearly separable and linear SVM might fail to find a plane that separates the two classes, and more sophisticated approaches are needed to deal with the mentioned problem. According to [14], SVM is effective in high dimensional spaces. Therefore, one of the possible configurations of SVM is to use a kernel-based classifier that tries to map the dataset into a higher-dimensional space in an attempt to find a hyperplane that could potentially separate the two classes in that space. Figure 2-3 depicts the non-linear case as well as the finding the separating plane using non-linear SVM.

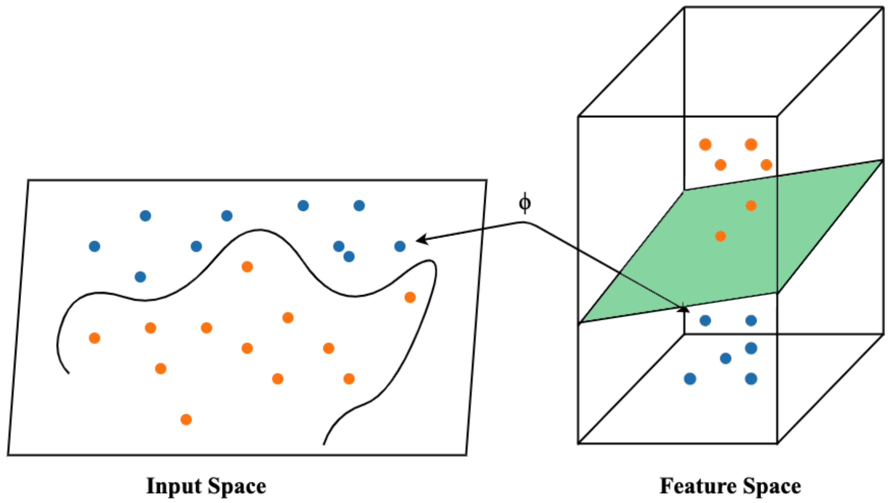


Figure 2-3: Non-Linear SVM

In this research, both linear and kernel-based approaches were tested and evaluated on the dataset in hand.

2.1.2.3. Decision Tree – CART

One of the popular supervised ML techniques is decision trees (DT). It can be used generally as a supervised method which tries to predict a target value based on some given attributes. The decision relies on different nodes and branches starting from top (root node) down to leaves (decisions). The root in the case of the decision tree represents the feature that is used to split the dataset first. This model can be utilized to enhance the understanding of each of the diabetes complications by visualizing the model tree which gives clear and easy to follow information. However, decision trees can be prone to overfitting as well as being unstable since adding a new attribute may result in a totally new tree (variance). These challenges can be addressed by tuning different hyperparameters such as the depth of the tree or the number of samples allowed per branch [15]. Figure 2-4 represents an example of a decision tree we developed using a subset of the dataset in hand to predict the presence of dyslipidemia in patients. To keep the example simple, Gender, Nationality Name and Diabetes type are the only three attributes used. The model achieved an accuracy of 55.7%. It is expected to reach better results after applying some more preprocessing steps, tuning the hyperparameters as well as using all the attributes.

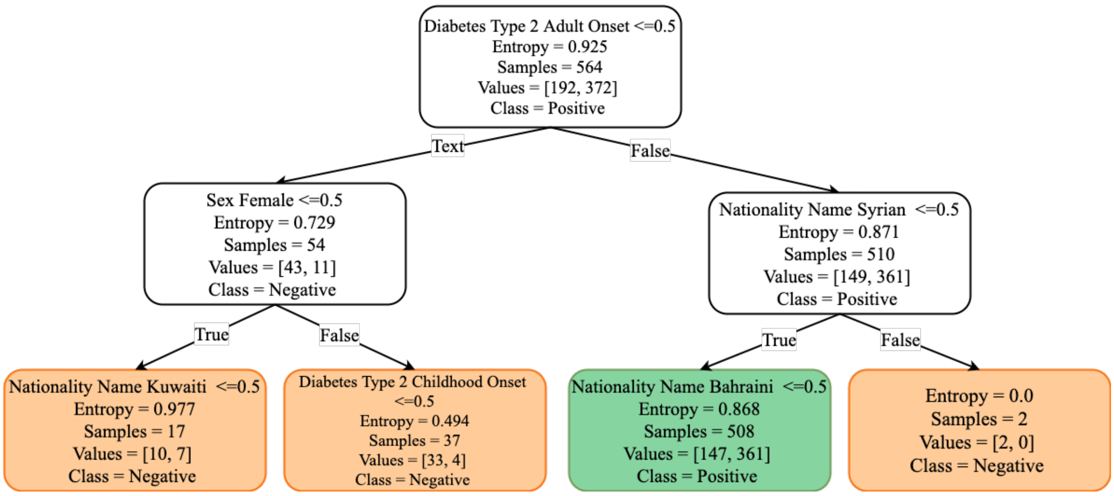


Figure 2-4: Decision Tree Example

The tree generated in Figure 2-4 is only the first three levels of the generated tree in this model. The tree in Figure 2-4 can be divided into three parts: root, internal nodes, and leaves. The leaves represent the output class, in other words the decision, whereas the root and the internal nodes represent the split conditions. Hence, in order to read the generated tree properly, we start from the root, if the patient has Type 2 Diabetes Mellitus (T2DM) Adult onset then we go down to the right node. In the new node, if the patient is Syrian then he/she most probably does not have dyslipidemia. In each node we can find different information. For example, 'samples' mean the number of records that appear at this node, whereas 'value' splits the records into [negative, positive] values. The criteria to select the attribute to split the data on in each of these nodes depends on two measurements, Entropy and Information Gain. Entropy is a measure of disorder or uncertainty and the goal of machine learning models in general is to reduce uncertainty. Information gain on the other hand is calculated by comparing the entropy of the dataset before and after a transformation [16]. Equations (2) and (3) can be used to calculate entropy and information gain, respectively.

$$\mathrm{Entropy}(S) = \sum_{i \in I} -p_i \log_2 p_i \tag{2}$$

where,

S: the current dataset for which entropy is being calculated.

I: the set of classes in S.

p_i: the proportion of the number of elements in class i to the number of elements in set S.

$$\mathrm{Gain(S, A)} = \mathrm{E(S)} - \sum_{t \in T} p_i E(t) \tag{3}$$

where,

$\mathrm{E(S)}$: entropy of set S

T: the subsets created from splitting set S by attribute A such that $\mathrm{S} = \cup_{\mathrm{t} \epsilon \mathrm{T}} \mathrm{t}$

$\mathrm{p_t}$: the proportion of the number of elements in t to the number of elements in set S.

$\mathrm{E(t)}$: entropy of subset t.

2.1.2.4. Random Forest

In addition to the above three algorithms, ensemble algorithms can be used for classification problems as well. Ensemble approaches, which use multiple learning algorithms, have proven to be an effective way of improving classification accuracy. Usually, ensemble methods are decision-tree-based algorithms [17]. There are two main categories of ensemble methods: Bagging and Boosting. Bagging methods such as Random Forest (RF) algorithm [18], works by building several decision trees instead of relying only on one tree. This approach attempts to improve the classification performance.

Random Forest can be used as a classification problem where the output is chosen based on a predefined criteria. For instance, RF can use what is called majority voting to select the final output by choosing the class that is selected by the majority trees in the forest. Another implementation of a random forest uses probabilities to select the class with the highest probability across all trees in the forest [17]. The later approach was followed in this study.

As mentioned in section 2.1.2.3, decision trees are prone to overfitting [19], which means that they could suffer from a very high variance in their predictions. This overfitting in DTs is mainly because they can learn rare and irregular behavior in the dataset, hence, they might not be able to generalize well. Nevertheless, random forests are always a great solution to the overfitting problem of decision trees.

Since Random Forest is a bagging algorithm, this means in order to avoid overfitting, RF selects a subset of the features available in the dataset to create a 'bag' of features to be used to build a tree in the forest. Each tree in the forest has a different subset of features. It is worth mentioning that in this case, it is possible to select the same attribute in more than one tree. In contrast, it is possible for another feature to not be selected at all [18]. Another important parameter to choose for RF is the number of trees needed in the algorithm, the selection of the best number of trees is necessary for achieving the best possible performance. The forest has an N number of different trees, all have shapes as shown in Figure 2-4. The final result (or classification output) is selected based on majority voting or probability calculations.

2.1.2.5. AdaBoost

In contrast to bagging methods, boosting algorithms build a model sequentially by minimizing the errors from previous models while increasing or boosting the influence of high-performance models. AdaBoost [17] is one example of boosting methods, which can be utilized for classification as well as regression. AdaBoost consists of multiple 'weak' models or learners that aim to classify and predict the output. In its basic format, the weak learners in AdaBoost are decision trees with a single split, called decision stumps. Each learner is responsible to handle a subset of the data that is best to predict. By using a group of such learners, the final model can be proven to converge into a strong learner. Since the base model used in AdaBoost to build the learners is Decision Tree, the most important hyperparameters to adjust in the model are the depth of each tree, hence the depth of the 'weak' learner, as well as the number of the learners.

AdaBoost can be affected by outliers, this is because such boosting algorithm builds each tree on previous trees' residuals. However, by using such an ensemble method, bias and variance problems could be avoided. Bias occurs when the model has limited flexibility to learn the true signal from the dataset. On the other hand, when a model has high variance, this means that it changes drastically to meet the needs of every point in our dataset (overfitting). By combining several models such as in AdaBoost, the severity of bias and variance problems can be reduced.

2.1.2.6. XGBoost

Another state-of-the-art algorithm that proved to be effective in different classification and regression tasks is XGBoost [20]. XGBoost stands for "Extreme Gradient

Boosting”. It is another example of tree boosting algorithms that uses multiple learners to build the final model. Moreover, since it relies on the gradient algorithm, the errors are minimized by the gradient descent algorithm. Furthermore, it is a perfect combination of software and hardware optimization techniques to yield superior results using fewer computing resources in the shortest amount of time.

The optimizations achieved by XGBoost can be split into two main groups, system optimizations and arithmetic enhancements. For system optimizations, XGBoost relies on a parallelized implementation. This parallelization enhances and reduces the training time needed. Another systematic enhancement can be noticed for tree pruning method. XGBoost uses max depth parameter as specified at the point of split instead of using negative loss criterion. This approach improves the computational performance significantly. For the arithmetic enhancements, the algorithm uses regularization to prevent overfitting as well as handling missing values problems. However, XGBoost requires more attention when it comes to hyperparameter tuning due to the large number of parameters that need to be tuned. For all the earlier mentioned points, XGBoost is going to be tested and evaluated on the dataset in hand. It is also expected to achieve good performance [20].

Table 2-1 summarizes some pros and cons for the algorithms discussed in this section. It is worth noting that Table 2-1 specifically summarizes the pros and cons that were mentioned earlier.

Table 2-1: Supervised techniques: Pros and Cons

Algorithm Name	Pros	Cons
Logistic Regression	Simple to implement	The assumption of linearity between the dependent variable and the independent variables.
SVM	Effective in higher dimensions.	Takes time to process for large datasets
Decision Tree	Simple to understand, interpret, visualize.	Prone to overfitting

Random Forest	Reduce overfitting possibility, Good handling of missing	Slow with large datasets
Adaboost	Easy to implement, Reduce overfitting possibility	affected to outliers
XGBoost	Reduce the impact of outliers	Harder to tune

2.2. Diabetes Diagnosis

Diabetes is one of the diseases that have been studied by researchers over years. Therefore, there is a significant number of available publications in this field. However, there is still much to be discovered and clarified. In the following sections, a detailed description of some of the available literature related to DM is introduced. The summary of each research contains the objectives, the dataset and algorithms used, as well as the evaluation metrics and the results indicated in each study.

Diagnosis and prognosis of DM received a lot of attention. Hasan et al. [21] proposed a new approach for diabetes prediction from the PIMA Indians Diabetes (PIDD) dataset. The dataset consists of 768 female patients, specifically 268 diabetic patients (positive) and 500 non-diabetic patients (negative) with eight different attributes: pregnant, glucose, pressure, triceps, insulin, BMI, pedigree, and age. As mentioned by the authors, preprocessing is the heart of achieving state-of-the-art results, which consists of outlier rejection, substitute the mean for missing values, data standardization, feature selection, and K-fold cross-validation (5 folds in this case). *k*-NN, Decision trees, AdaBoost, Random Forest, Naïve Bayes, and XGBoost were all used and tested in this study. The authors also used an ensemble technique that aims to boost the performance using a group of classifiers. In ensemble methods, the aggregation of the output from different models can improve the precision of the prediction. The best models used together were AdaBoost and XGBoost. The Area Under the Curve (AUC) was chosen as the performance metric. The paper was able to achieve an AUC score of 0.95 which outperformed other studies.

Sisodia et al. [22] aim to prognosticate the likelihood of diabetes in patients with maximum accuracy. The PIDD dataset used in this paper is the same as the previous one [21]. Decision tree, SVM and Naïve Bayes were all used in this experiment to detect diabetes at an early stage. Accuracy, precision and recall as well as F1-score were used to measure the best model performance. As reported in the paper, Naïve Bayes achieved the best performance results with a maximum accuracy of 76.3%.

Meng et al. [23] conducted a performance comparison between three data mining models for predicting diabetes or prediabetes. The data mining models are Logistic Regression, Artificial Neural Network (ANNs) and Decision Trees. The balanced dataset used consists of 735 patients and 752 normal controls. The 12 attributes used in building the models are gender, age, marital status, educational level, family history of diabetes, BMI, coffee drinking, physical activity, sleep duration, work stress, consumption of fish, and preference for salty foods. All the previous attributes were gathered by a questionnaire. The authors concluded that the C5.0 decision tree performed the best on classification accuracy.

2.3. Diabetes Complications

As we can notice from the previous section, predicting and diagnosing diabetes in patients received a lot of attention. However, there is still a need to study and cover other fields related to diabetes. For instance, there is a necessity to focus on the complications related to diabetes. As indicated in [24], the global decrease of mortality rate and the increasing in diabetes prevalence has increased the overall mean years lived with diabetes and could lead to a diversification of diabetes morbidity, including renal disease or nerve damage.

2.3.1 Prediction

In addition to predicting the presence of diabetes in patients, few existing studies have reported the use of machine learning to develop prediction models of diabetes complications. For instance, in [15], the authors built a model to predict some chronic diabetes complications, especially eye disease, kidney disease, coronary heart disease, and hyperlipidemia. The authors started with a dataset of 455 records. The number of records was decreased through data selection and cleaning. The final number of records as well as the number of features used to build the model were not mentioned in the paper. The authors used iterative decision tree (ID3) algorithm for building the model [25]. To evaluate its performance, 10-fold cross validation was used, yielding an accuracy of 92.35%.

Dagliati et al. [26] focused on predicting the onset of retinopathy, neuropathy, or nephropathy in T2DM patients at different time scenarios, at 3, 5 and 7 years from the first visit at the hospital. The first visit at the hospital provides the patient's health status. The selection of patients in this study consists of the following criteria: patient has a follow-up time longer than the corresponding temporal threshold (3, 5 or 7); patient develops the complication after the first visit; patient's complication onset date has been registered. The dataset has been collected by the Istituto Clinico Scientifico Maugeri (ICSM), Hospital of Pavia, Italy for over 10 years. It contains 943 records with the following features: gender, age, time from diagnosis, body mass index (BMI), glycated hemoglobin (HbA1c), hypertension, and smoking habit. The classification models used were LR, NB, SVM and RF. The missing data problem was handled using MissForest [27], whereas the unbalanced class problem was solved by oversampling the minority class. According to the paper, the maximum accuracy score was achieved by using LR algorithm with 83.8%.

The objective of paper [13] is to study and predict diabetes induced nephropathy and cardiovascular disease by building different machine learning algorithms. The dataset used in this paper is a result of a study conducted at the Tokyo Women's Medical University Hospital and 69 collaborating institutions in Japan. The dataset consists of 779 T2DM patients with 164 features including results of different pathological tests. The data was cleaned by dropping instances with a significant number of missing values. Following the approach in [26], MissForest was used for data imputation with 64 trees and 10 iterations. Furthermore, feature scaling was also applied. This step is important to enhance the model's performance. Because the dataset used contains a large number of features, Principal Component Analysis (PCA) was used for feature extraction. This step converted the dataset into a smaller number of features with the same number of records. SMOTE was used to help solving data unbalanced problem. Methods such as Logistic Regression, SVM, Naïve Bayes, Decision Tree and Random Forest were used in a supervised environment. Decision Tree without PCA produced the best results for nephropathy with an AUC score of 0.87. While Naïve Bayes without PCA produced the best results for cardiovascular disease, with an AUC score of 0.74.

In [28], the authors focused only on studying one complication which is sarcopenia. Sarcopenia is a geriatric syndrome, and it is closely related to the prevalence of type 2 diabetes mellitus (T2DM). The goal for this paper is to make risk assessment of sarcopenia easier by building ML models using SVM and RF. The dataset used in the paper is limited in size with only 132 records of patients aged over 65 and diagnosed with T2DM. It contains several records for each patient, such as age, duration of diabetes, history of hypertension, smoking and drinking habits as well as some medical records like serum albumin and 25-OH-Vitamin D3. The missing value problem was solved using a *k*-NN classifier with a *k* set to 10. The area under the receiver operating characteristic (ROC) curve (AUC) was over 0.7, and the mean AUC of SVM models was higher than that of RF. It is worth mentioning that such AUC scores reflect a limitation in the model's performance since typically a proper model should achieve higher AUC scores.

2.3.2 Medical rules extraction

The use of unsupervised machine learning techniques on diabetes is very limited. According to [29], only 15% of the available literature on diabetes is characterized by unsupervised learning approaches, which makes it an attractive subject in this thesis. This section discusses some available literature on association rule mining in the health sector in general and for DM in particular.

Lakshmi K.S et al. [30] described a method to extract association rules from medical health records using various data mining algorithms. The dataset used in this paper is text-based health records documents. Hence, the authors proposed two sequential steps for data processing. The first step indicated in the paper was extraction of medical terms. This phase was conducted using Natural Language Processing (NLP) techniques to extract useful terms from the soft copies of the medical documents. After that, association rule mining methods were applied on the output of the previous step. In the proposed method, Apriori algorithm, FP Growth, Eclat, Apriori TID, RElim and Close algorithms were all used for finding association rules. After generating association rules using the previous algorithms, multi-criteria decision analysis was carried out to choose the best algorithm. Electre 1 method was chosen for multi-criteria decision analysis. The Algorithms were evaluated based on 3 parameters: performance, memory space and response time. The performance was measured based on the average number of rules generated on varying values of support and confidence. According to the paper, the rules generated by the association rule mining algorithms were almost the same.

The authors started with a confidence value of 100 and a support value of 10. After that, they kept decreasing the confidence and increasing the support. Based on the multiple criteria decision analysis, RElim, Apriori TID and FP-Growth algorithms proved to be more effective compared to other algorithms.

The work in [31] presents another study that used association rule mining for early prediction of diabetes in patients. The dataset utilized in this work consists of a few risk factors such as age, conditions, medication, and comorbidities. It is worth mentioning that a discretization step was applied on the numerical attributes. Apriori algorithm was applied to the new dataset as well as four different methods for rule set summarization. According to the paper, the rule set summarization step helps in compressing a ruleset ‘L’ to a smaller set ‘A’ such that L can be recovered from A with marginal loss of information. APRX-Collection, RPGlobal, Top-K, and BUS are the four techniques used. As mentioned in the paper, bottom-up summarization (BUS) was most suitable and used for accurate results.

The proposed work in [32] searched for association rules of diabetes mellitus (DM) patient data with renal, ophthalmic, neurological, and peripheral circulatory complications. The paper’s aim was to extract useful medical information by applying Apriori algorithm on a diabetes dataset using WEKA [33]. The dataset consists of some attributes such as age, gender, and occupation.

Chapter 3. Problem Statement

From the previous literature review, it is observed that research has focused on predicting the presence of type 2 diabetes in patients, whereas predicting diabetes complications has received less attention. Moreover, the number of complications considered in most of the available literature is very limited, as it does not exceed two or three complications. Moreover, there is a clear limitation when it comes to the number of features used in each study and the nature of these features. For instance, the number of the available medical tests in [28] is limited to five medical tests. Another important finding is the limited number of studies that cover the relations between medical records and different types of complications, which may significantly help doctors to have a better understanding of the disease and its effects. The work presented here attempts to address some of these shortcomings while focusing mainly on UAE diabetic patients as subjects. In particular, diabetes complications were investigated thoroughly. A dataset with comprehensive features and records is used, which was collected from the Rashid Centre for Diabetes and Research (RCDR) located in Ajman, UAE [34]. The dataset consists of 79 features and 884 diabetic patients and includes eight complications, which can be considered as one of the highest number of complications available in a dataset. The complications covered in this study are: metabolic syndrome, dyslipidemia, neuropathy, nephropathy, diabetic foot, hypertension, obesity, and retinopathy. A further description of the dataset is provided later in section 4.1.

The first objective of this study aims to mitigate the lack of research that focuses on studying the relations between dataset attributes. This step is essential to extract valuable knowledge which may lead to new hypotheses targeting deeper understanding and further investigation in DM. To achieve this objective, several association rule mining algorithms were used to generate the rules that describe these relations. In order to maximize the benefit of this step, the attributes involved in the best generated rules were utilized for further processing.

As described earlier, some studies applied machine learning techniques to predict the presence of diabetes complications. However, most of these studies suffered from data availability, especially to predict diabetes complications. As a result, the second objective of this research is to achieve reliable and improved results in predicting diabetes complications in diabetic patients using state-of-the-art machine learning algorithms. Different models were built and evaluated by using the available set of attributes.

Since we have 79 features in the dataset, it makes sense to try to understand the most dominant risk factors (features) that affect each one of the eight complications under study. This feature selection objective was achieved by utilizing the supervised ML models built using all the attributes. The top 5 and 10 features were selected for further considerations. For instance, both subsets were used to build more specialized and focused models. Moreover, a comparison between these feature sets and the features found in the rules generated in ARM step was conducted and analyzed.

To the best of our knowledge, and by considering the aforementioned objectives and goals, this research could be considered one of very few studies that go beyond building a classification model to cover and establish a full understanding of diabetes data and extract hidden knowledge by using a variety of medical tests and features.

Chapter 4. Methodology

In this chapter, we introduce and analyse the dataset used in this study as well as the needed preprocessing techniques. Afterwards, a description of the experimental phases needed to meet each of our objectives is provided in detail. Figure 4-1 summarizes the workflow followed in this research, which is discussed in detail in the following sections.

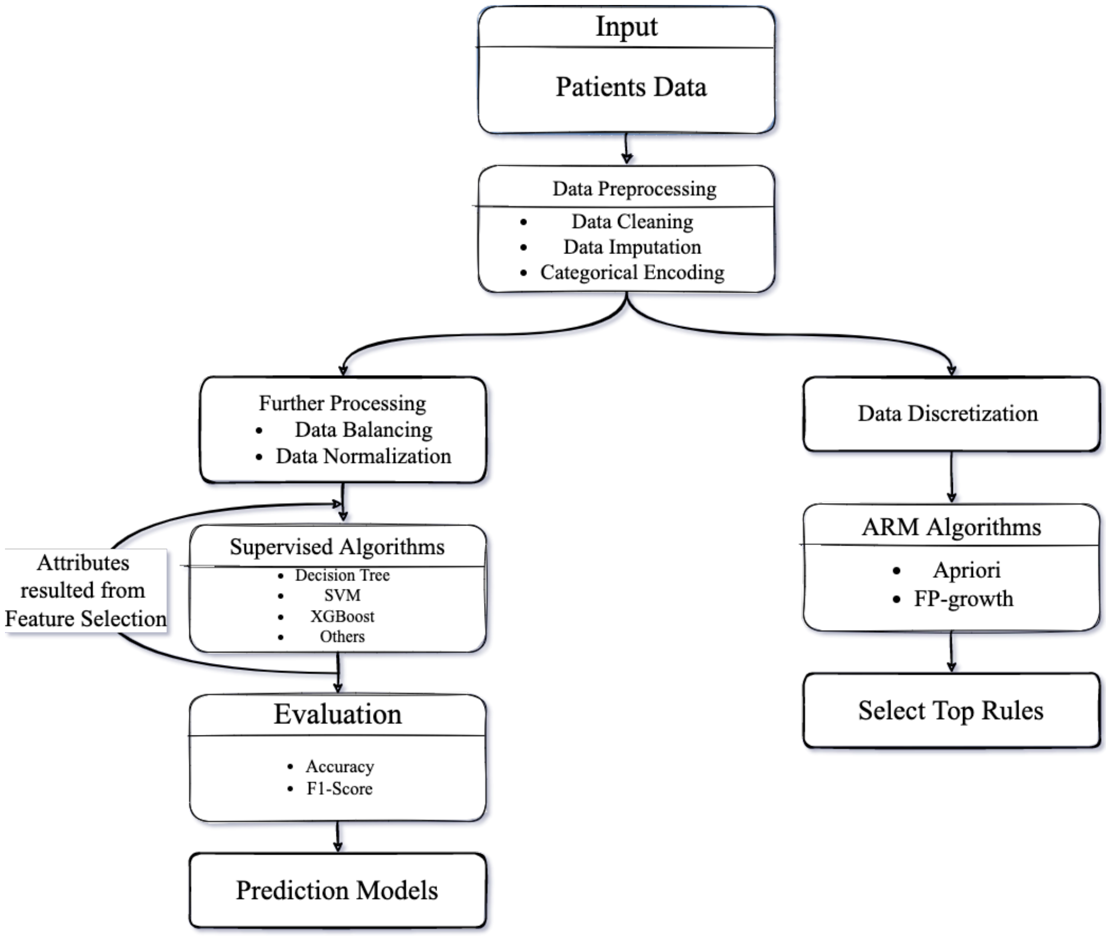


Figure 4-1: Modeling workflow

4.1. Research Dataset

To build reliable machine learning models, a sizable amount of data is required for training. In this research, a dataset collected from the Rashid Centre for Diabetes and Research (RCDR) located in Ajman, UAE [34] was utilized. The center focuses mainly on providing diagnosis, counselling, education, treatment and lifestyle options, support and motivation to the patients. The selection criteria for the collected data must confirm to the following, all the patients included in this study were already diagnosed with diabetes and any of its complications under study. Moreover, the dataset mainly consists of medical records which were reported by the RCDR. It is worth mentioning that the data is private and not publicly available.

4.1.1 Dataset description

The data consists of 884 patients with 79 input attributes and 8 output classes (8 separate attributes). The 79 input attributes are distributed as follows, 73 are numerical attributes and 6 nominal attributes. From the 73 attributes we have 64 medical tests all of them have numerical values. Some of these features are described below:

- Age: the age of the patients available in the dataset ranges from 10 years old to 91 years old.
- Gender: Male or Female.
- BMI: body mass index, which is a person’s weight in kilograms divided by the square of height in meters [35].
- HbA1c: it is an indication of the average level of blood sugar over the past 2 to 3 months [36].
- Vitamin D: as indicated by National Institutes of Health (NIH) [37], Vitamin D is a numerical value that promotes calcium absorption in the gut and maintains adequate serum calcium and phosphate concentrations to enable normal bone mineralization and to prevent hypocalcemic tetany.
- Diabetes Type: Type 2 and Type 1 in Adults and children. The four possible combinations are found in the dataset in hand.
- Blood Pressure: it contains two types of pressures: a systolic Blood pressure (normal below 120) and Diastolic Blood Pressure (normal below 80) [38]. Both values are included as two separate numerical attributes in our dataset.

For the output (target) attributes, we have 8 main complications: metabolic syndrome, dyslipidemia, neuropathy, nephropathy, diabetic foot, hypertension, obesity and retinopathy. A brief description of these complications is provided below:

- Hypertension: according to WHO [39], hypertension is a serious medical condition that significantly increases the risks of heart, brain, kidney and other diseases. Hypertension occurs when blood pressure is too high.

- Obesity: overweight and obesity are defined as abnormal or excessive fat accumulation that may impair health [40]. For adults, WHO defines obesity as a BMI greater than or equal to 30.
- Dyslipidemia: is defined as having a high plasma triglyceride concentration, low high-density lipoprotein cholesterol (HDL-C) concentration, and decreased concentration of low-density lipoprotein cholesterol (LDL-C) [41].
- Metabolic Syndrome: is a cluster of metabolic disorders. For example, high blood pressure alone is a serious condition, but when a patient has high blood pressure along with high fasting glucose levels and abdominal obesity, this patient may be diagnosed with metabolic syndrome [42].
- Diabetic Foot: is defined as the foot of diabetic patients with ulceration, infection and/or destruction of the deep tissues, associated with neurological abnormalities and various degrees of peripheral vascular disease in the lower limb [43].
- Neuropathy: nerve damage from diabetes is called diabetic neuropathy. According to CDC [44], High blood sugar can lead to this nerve damage.
- Nephropathy: is a disease of the kidneys caused by damage to the small blood vessels or to the units in the kidneys that clean the blood. People who have had diabetes for a long time may develop nephropathy [45].
- Retinopathy: is any damage to the retina of the eyes, which may cause vision impairment. Diabetic retinopathy (DR) occurs when high blood sugar damages the blood vessels below the retina [46].

4.1.2 Dataset statistics

After exploring the data using Scikit-learn, it is noticed that most of the attributes are prone to missing values ranging from 1% to 99%. Table 4-1 below introduces all the dataset input features sorted based on the percentage of the missing values.

Table 4-1: Percentage of missing values for each attribute

Missing Percentage	Attributes Names
100% - 90%	TestUrine Creatinine SPOT, TestB 12, TestFerritin, TestAnti TG, TestTransferrin, TestFree T4, TestFree T3, TestFolate, TestIron, CriticalRangeTo, CriticalRangeFrom, TestGlucose Random, TestPTH, TestaTPO, TestAnti GAD 65, TestFT4, TestFT3
89% - 80%	N/A

79% - 70%	TesteGFR
69% - 60%	N/A
59% - 50%	N/A
49% - 40%	TestCystatin C
39% - 30%	TestMicroalbumin creatinine ratio, TestMicroalbumin, TestHbA1c (mmol), TestEAG
29% - 20%	TestUrine creatinine, TestAST, TestBilirubin DIR
19% - 10%	TestNEUT Absolute, TestMXD Abs, TestMXD, TestNeutrophils, TestAlkaline Phosphatase, TestC- peptide, TestBICARB, TestGGT, TestBilirubin Total, TestCRP-HS, TestLympacyte Absolute, TestMCV, TestPhosphorous', TestHbA1c (%)', TestTriglycerides, TestHDL, TestTotal cholesterol, Test25-OH Vitamin D Total, TestGlucose Fasting, TestK, TestLDL, TestNa, TestCreatinin, TestCL, TestInsulin, TestALT, TestCalcium, TestWBC, TestPLT, TestRBC, TestUrea, TestAlbumin, TestHb, TestHCT, TestLymphocyte, TestMCH, TestMCHC
9% - 0%	HospitalNumber, EpisodeDate, EpisodeDescription, Gender, Age, BPSystolic, BPDiastolic, NationalityName, HeightMetres, WeightKg, BMI, DiabetesType, NormalRangeFrom, NormalRangeTo, TestTSH, DiabetesAge

Since the dataset used in this study consists mainly of numerical data, we used Scikit-learn to explore some statistics about these attributes as summarized in Table 4-2 below. As it can be noticed from the table, the ranges vary from one attribute to another, with significant differences between some attributes. This issue needs to be addressed especially when building supervised classification algorithms, where such a problem significantly affects the overall performance. More details are provided later in section 4.4.2.

Table 4-2: Statistical Analysis for numerical attributes

Test Name	Minimum	Maximum	Mean	Standard Deviation
Age	10	91	50.876	15.039
BPSystolic	77	219	137.476	20.366
BPDiastolic	50	122	76.077	10.834
Height (m)	1.34	1.86	1.609	0.096
Weight (Kg)	28.5	160.4	79.805	18.111
BMI (Kg/m^2)	15.45	64.89	30.793	6.494
NormalRangeFrom	0	3540	11.705	147.995
NormalRangeTo	2.55	22900	60.043	1025.163
TSH	0.009	155.2	2.644	5.989
DiabetesAge	-1	60	9.05	8.421
25-OH Vitamin D	10.2	226	50.241	28.608
Glucose Fasting	1.92	33.48	10.586	4.739
HbA1c %	4.2	15.1	8.326	2.104
HDL	0.25	2.91	1.194	0.361
Phosphorus	0.48	2.19	1.12	0.169
Total Cholesterol	1.62	13.81	4.579	1.271

Triglycerides	0.35	36.61	1.795	2.028
CL	90.4	110.1	101.082	3.007
Creatinine	8	596	66.547	36.007
K	3.16	6.27	4.315	0.402
LDL	0.66	8.18	2.918	1.08
Na	128	147	137.595	2.65
ALT	5.7	811	28.435	34.608
Calcium	1.58	2.82	2.321	0.104
Insulin	0.549	234.6	18.28	18.462
WBC	2.72	17.81	7.291	2.062
Albumin	26.2	49.7	40.866	3.525
CRP-HS	0.18	211.04	6.807	12.301
Hb	67.2	178.1	131.23	17.941
HCT	0.225	0.533	0.401	0.047
Lymphocyte Absolute	0.4	5.3	2.544	0.783
Lymphocyte	0.12	0.654	0.359	0.095
MCH	16.62	38.81	26.05	3.138
MCHC	258.2	367	327.241	15.591
MCV	56.32	114.21	79.409	7.057
PLT	57	796.1	271.46	82.941
RBC	2.84	8.083	5.076	0.679
Urea	1.1	34.3	4.99	2.604
Alkaline Phosphatase	21	599	85.13	36.269
BICARB	12.8	33.9	25.68	2.723
Bilirubin Total	1.6	52.9	7.661	4.656
C-Peptide	0.003	4.35	1.032	0.527
GGT	6	729	43.023	55.723
MXD Abs	0.2	2.5	0.625	0.248
MXD	0.3	3.111	0.876	0.276
NEUT Absolute	1.1	12.8	4.116	1.609
Neutrophils	0.236	0.815	0.553	0.099
AST	7.9	662.3	21.756	26.983
Bilirubin DIR	0.7	34	3.188	2.008
Urine Creatinine	953	40763	11411.484	6780.629
EAG	4	21	10.533	3.395
HbA1c mmol	21.4	141	66.729	23.304
Microalbumin Creatinine ratio	1	5514	178.213	559.574
Microalbumin	0.7	11138	171.083	630.835
Cystatin	0.388	6.512	1.009	0.503
eGFR	8	575	118.939	69.931
FT3	2.32	7.11	4.456	0.799
FT4	3.73	20.46	14.145	3.112
Anti GAD 65	5.286	172.54	47.873	43.14
aTPO	6.77	459.3	153.335	161.468
Glucose Random	5.91	15.99	11.775	3.729
PTH	2.84	9.96	5.814	3.097
CriticalRangeFrom	1.5	3.3	3	0.735
CriticalRangeTo	3	20	17.176	6.94
Anti TG	54.47	288.8	171.635	165.696
B 12	312.4	312.4	312.4	NaN
Ferritin	6.9	137.3	72.1	92.207
Folate	18.51	37.9	28.663	9.727
Free T3	3.3	5.02	4.347	0.919
Free T4	11.48	13.28	12.593	0.973

Iron	3.51	12.96	8.22	4.974
Transferrin	2.33	3.5	2.863	0.592

One of the common challenges that can be faced when building and training machine learning and data mining models is dealing with unbalanced dataset. Figure 4-2 represents this challenge which occurs in the dataset in hand. It is noticed that most of the complications are unbalanced. To be more precise, neuropathy, nephropathy, retinopathy and diabetic foot attributes all suffer from a severe unbalanced distribution, urging the need to utilize an appropriate balancing method to solve the problem. Section 4.4.1 describes the approaches followed to solve such a problem.

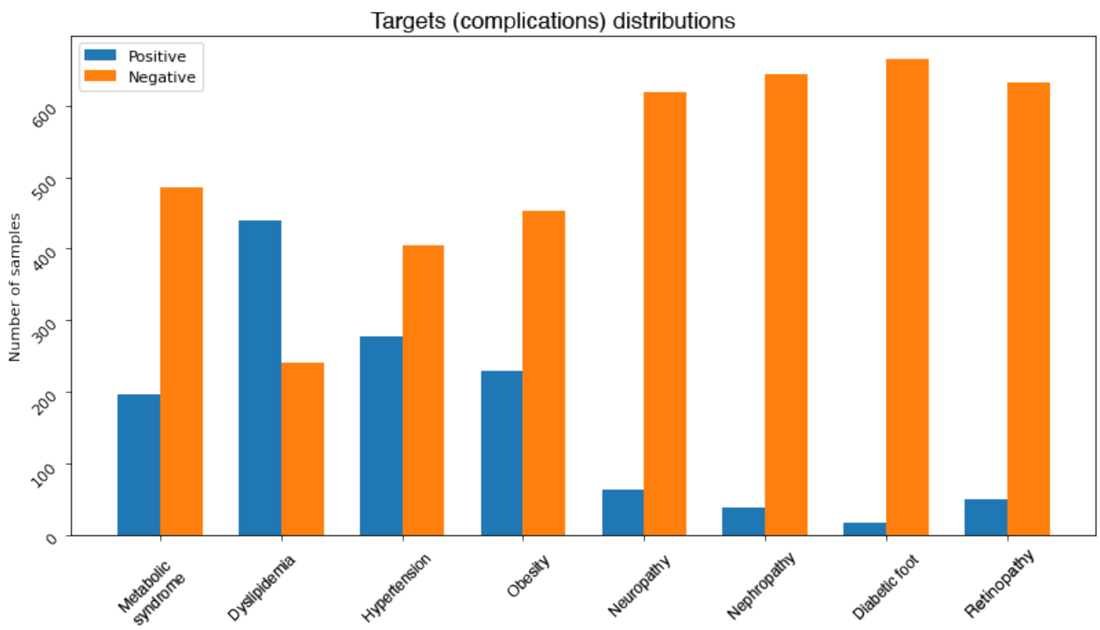


Figure 4-2: Classes distributions for each complication

4.2. Data Pre-processing

As noticed in the previous section, the dataset in hand requires several preprocessing steps. Such steps are important to adequately train the models. For instance, some of the supervised models cannot be trained at all when the dataset contains missing values. Moreover, processing the dataset can enhance the overall performance of the models. In this section, a detailed explanation is given for the general preprocessing steps needed in all the experiments, starting from basic data cleaning ending with testing several methods for data imputation to solve the missing values problem.

4.2.1 Data cleaning

The first step towards processing the dataset is cleaning it and dropping the unnecessary records and attributes by following a systematic procedure. For instance, the dataset consists of several categorical values that need to be removed for confidentiality purposes. Such features are 'HospitalNumber', 'EpisodeDate' and 'EpisodeDescription'. Furthermore, the dataset consists of missing values for the diabetes type for some patients, which is a critical information in this research since we are studying diabetes complications in diabetic patients. Therefore, all the instances suffering from this problem were dropped. The total number of such records are 26 patients.

Another step found to be needed in this study is checking the total number of missing values per record (or patient). By conducting several experiments, a threshold of 60% was set for this step, meaning that any record with missing values more than this threshold should be dropped from the dataset. It was found that removing all such records achieved better performance compared to other experiments where this problem was ignored. Moreover, since data imputation is needed to proceed with the experiments and build the models later in this study, keeping records with such a high percentage of missing values may severely affect the dataset and result in having unrealistic records. The total number of dropped values in this step reached 130 patients.

Following the approach in [13], the missing values were also investigated per attribute. Based on several experiments, a threshold of 40% was set for this step. It was found that 16 numerical attributes have missing values of more than 40%. More precisely, and as noticed in Table 4-1, most of these attributes have more than 90% missing values. This specific threshold was selected experimentally and affected by the literature findings [13].

4.2.2 Data imputation

In general, most of the available machine learning algorithms cannot handle the missing values problem that occurs in the dataset. Having such a problem in a dataset is a very common case in data mining and machine learning problems. Several data imputation methods should be studied and tested to reduce the impact of this issue.

The missing values problem can be found in categorical values as well as numerical ones. For the categorical values in the dataset in hand, this issue can be considered mild which occurs only in the Nationality attribute. The most frequent value in that attribute i.e., *United Arab Emirates,* was used to fill the missing values.

On the other hand, three different methods were extensively tested and evaluated to solve the missing values problem in numerical attributes. Following the same approach in [21], the first method used to overcome this challenge is by using the mean substitution method. Mean substitution is a statistical way to represent and fill any missing value in an attribute (feature) with the average of observed data for that attribute in other records or patients. While this imputation method is the simplest, it is not the best choice in most cases. This is mainly because it may lead to biased results, hence, not reflecting the reality.

Another technique to solve this problem is by using a *k*-NN model to impute the missing values. The *k*-NN model goal is to find the nearest neighbors of the missing value based on some predefined distance metric. Each missing feature is imputed using values from N nearest neighbors that have a value for the feature. The features of the neighbors are averaged uniformly or weighted by distance to each neighbor. If a sample has more than one feature missing, then the neighbors for that sample can be different depending on the particular feature being imputed [47]. Following the approach in [28], the number of neighbors selected for this model is 10.

The third and last method used in this research was MissForest [27]. This method imputes missing values using Random Forests in an iterative fashion. The first step in this algorithm is selecting the first attribute which has the least number of missing values (candidate attribute). After the selection, the missing values in the candidate attribute are filled by the mean of that attribute. Moreover, the candidate attribute is fitted on a Random Forest model and treated as the output where other attributes in the dataset are treated as inputs to this model. After training the Random Forest model, the missing records of the candidate attribute are imputed using the prediction from the fitted Random Forest. These steps continue to cover all other attributes in the dataset.

In order to evaluate and compare the performance of all the three algorithms, Root Mean Squared Error (RMSE) was calculated as per equation (4), for all the three methods as follows. The first step was to simulate the missing value problem by choosing a complete subset of the dataset with no missing values. The total number of records in the complete subset was 217 records. After that, the missing values percentage in the original dataset was calculated and utilized to drop random values from each attribute in the complete dataset. More precisely, the percentage found was 4.4%, resulting in dropping 9 records per attribute in the complete dataset. After building the artificial dataset, the three mentioned methods were used to impute the missing values. As noticed in Table 4-3, it was found that MissForest results in the minimum RMSE value followed by k-NN and Mean methods. It is worth mentioning that Table 4-3 represents the RMSE for some randomly selected attributes as well as the total RMSE for all attributes.

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2} \tag{4}$$

n: is the total number of samples,
y_j: is the actual value of point j,
$\hat{y}_j$: is the predicted value of point j

Table 4-3: RMSE results for each imputation method

Method	BMI	Triglycerides	Total RMSE
MissForest	0.6264	1.2051	15.962
k-NN	0.9711	1.3514	18.56
Mean Substitution	0.8972	1.3378	19.788

In addition to calculating the RMSE values, a visual inspection was performed on the dataset imputed by MissForest. Figure 4-3 and Figure 4-4 show some examples of generated values for Albumin and Lymphocyte tests using MissForest. The values in blue represent the actual values found originally in the dataset, where the orange points show the calculated missing values. It is observed that the generated new values fit naturally since they follow the same trend found in the real data.

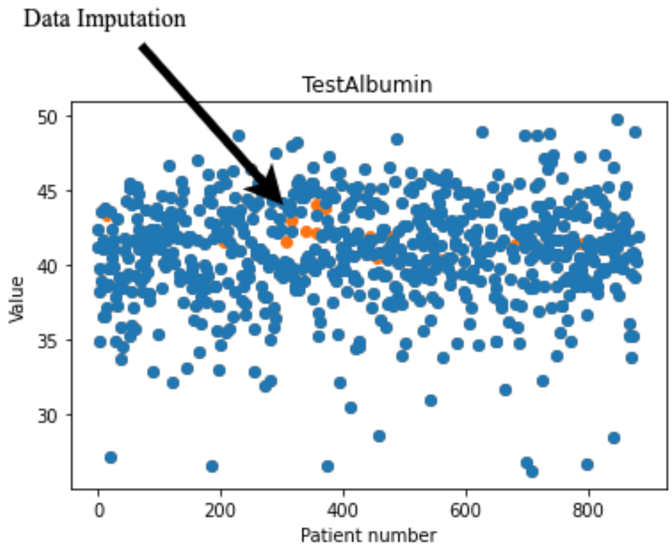


Figure 4-3: Original and generated Albumin Test values using MissForest

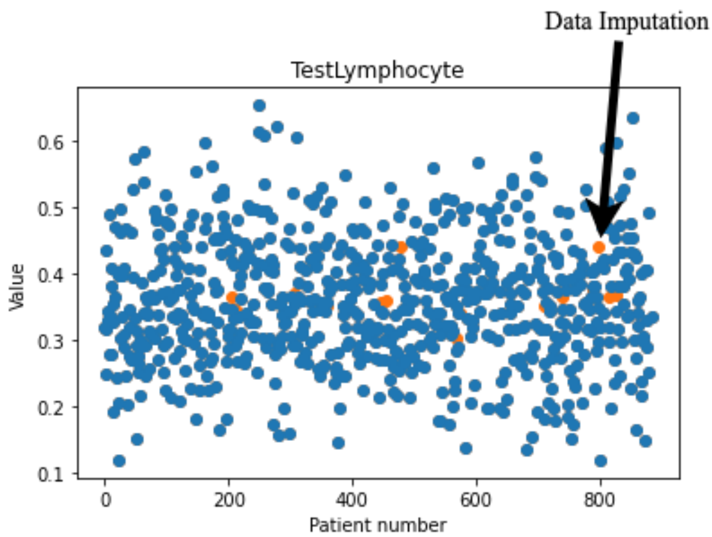


Figure 4-4: Original and generated Lymphocyte Test values using MissForest.

4.2.3 Categorical encoding

The last step that is needed for all the experiments to follow is applying categorical encoding. For this purpose, one-hot encoding was utilized due to its simplicity and its ability to preserve the information found in the attributes compared with other encoding schemes such as hash encoding. In general, one-hot encoding is mainly used when ML algorithms cannot handle the categorical values and can only use numerical data. One-hot encoding creates a "dummy" variable for each possible category of each non-numeric feature. Table 4-4 shows a small subset of the dataset before applying one-hot encoding where Table 4-5 represents the data after one-hot encoding is applied. As indicated before, the categorical values that need to be encoded to numerical values are 'Gender', 'NationalityName' and 'DiabetesType'. The dotted attribute in Table 4-5 indicates that there are other encoded options for 'NationalityName' in the actual dataset that are only skipped for this example.

Table 4-4: Categorical data before applying One-Hot Encoding

idx	Gender	Nationality Name	Diabetes Type
0	Male	United Arab Emirates	Type 2 Adult Onset
1	Male	United Arab Emirates	Type 2 Adult Onset
2	Female	United Arab Emirates	Type 1 Adult Onset
3	Female	United Arab Emirates	Type 2 Adult Onset
4	Female	United Arab Emirates	Type 2 Adult Onset

Table 4-5: Categorical data after applying One-Hot Encoding

idx	Gender Female	Gender Male	Nationality Name Bahrani	Nationality Name UAE	…	Diabetes Type 2 Adult Onset	Diabetes Type 1 Adult Onset
0	0	1	0	1		1	0
1	0	1	0	1		1	0
2	1	0	0	1		0	1
3	1	0	0	1		1	0
4	1	0	0	1		1	0

4.3. Association Rule Mining

One of the questions we are trying to address in this study is of what the possible relations between the medical records and different diabetes complications are. Answering this question will provide doctors with simple-to-read and easy-to-follow rules and may open the door for new diabetes-related research and discoveries. To achieve this goal, we make use of association rule mining techniques. As mentioned in [12], association rule mining (ARM) techniques are widely used to discover hidden relationships established by multiple attributes that characterize a complex process under investigation.

In addition to the previous steps of cleaning the data, filling the missing values as well as applying one-hot encoding to categorical data, data discretization is a very essential step that needs to be taken into consideration in order to use association rule mining methods efficiently. The rest of this section discusses data discretization.

4.3.1 Data discretization

In general, to build an association rule mining model categorical input data is required. From section 4.1, we know that our dataset consists mainly of numerical attributes, which need therefore to be discretized. In other words, each numerical value needs to be assigned into its appropriate category. Categorization step can be done in an automated manner depending mainly on the distribution of the data, or it can be achieved manually as well.

To illustrate the data discretization process, consider the example in Table 4-6, where Age and BMI are numerical values. According to the Center for Disease Control and Prevention (CDC) [48], it is possible to define four main categories for BMI in adults as follows: Underweight (BMI is less than 18.5), Normal (BMI is 18.5 to <25), Overweight (BMI is 25.0 to <30) and obesity (BMI is 30.0 or higher). Furthermore, by following the approach in [49], Age can also be divided into three groups: Adult (Age ≤ 49), Middle Age (50 - 59), Senior Adult (Age ≥ 60).

Table 4-6: Dataset before discretization

index	Gender	Age	BMI	Dyslipidemia[1]
0	Male	62	25.27	1
1	Male	53	26.64	0
2	Female	20	21.48	0
3	Female	33	31.05	0
4	Female	53	25.00	1
5	Female	61	32.04	1
6	Male	70	30.00	0

After defining the groups or categories for all numerical values available in the dataset, the data of Table 4-6 is discretized as shown in Table 4-7.

[1] A value of 1 represents a patient with the disease, 0 represents a patient without the disease.

Table 4-7: Dataset after discretization

	Gender	Age	BMI	Dyslipidemia
0	Male	Senior Adult	Overweight	1
1	Male	Middle Age	Overweight	0
2	Female	Adult	Normal	0
3	Female	Adult	Obesity	0
4	Female	Middle Age	Overweight	1
5	Female	Senior Adult	Obesity	1
6	Male	Senior Adult	Obesity	0

Since the processed dataset in hand consists of 55 numerical attributes, following a manual categorization method will require huge time and efforts. Hence, frequency-based binning was used in this study as an automated categorization approach. The goal of this method is to assign the data instances into different groups which all have the same number of instances.

It is worth mentioning that the best number of categories found to be useful in this dataset is NBins = 3, meaning that 3 categories were used for each attribute in the dataset. Moreover, we compared the manual approach and this automated approach using a subset of attributes. For instance, we found that by using the automated approach, the BMI is divided into three subsets, <=27.4, (27.4, 32.8], >32.8, which pretty much defines the main subsets found using the manual approach, especially the "overweight" category which is the most important and critical category since it directly affects several complications [4].

4.4. Diabetes Complications Prediction

The dataset with the available features allows us to diagnose some diabetes complications in patients. For this goal, supervised machine learning algorithms need to be applied on the dataset. Each generated model targets one complication (output class) at a time. Therefore, the final result consists of the top 8 performing models, one for each target (complication).

Before discussing the supervised algorithms, it is necessary to explain some additional preprocessing steps that need to be applied to achieve this goal, namely data balancing and data normalization. Figure 4-5 represents the block diagram of the proposed framework for predicting diabetes complications. It consists of several steps including the above-mentioned preprocessing steps, the use of repeated and stratified K-fold cross validation, hyperparameters tuning, and models evaluation. All these steps will be discussed in detail in the following sections.

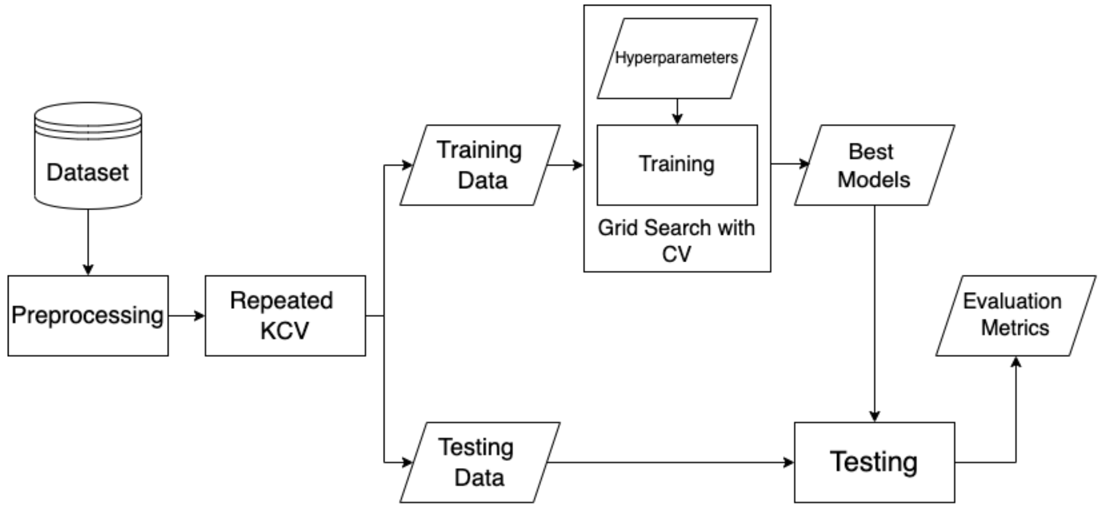


Figure 4-5: The proposed workflow for diabetes complications prediction using supervised ML algorithms

4.4.1 Data balancing

For handling data imbalance issue, several methods can be utilized and tested. For instance, one can reduce the number of instances in the majority class, or else increase the number of instances in the minority class. The rest of this section discusses the various methods and possible variations for each one and highlighting their pros and cons.

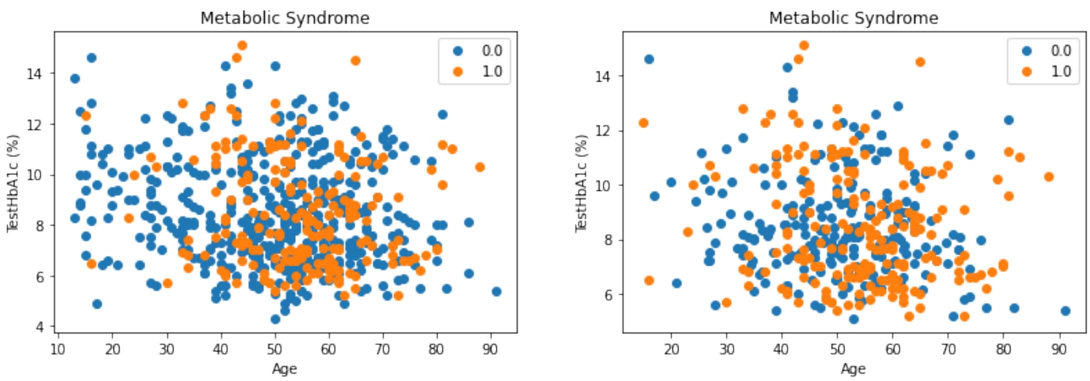
4.4.1.1. Under sampling

Under sampling is a general technique that is used to reduce the number of instances in the majority class. Generally speaking, performing under sampling on a dataset has some pros and cons. For instance, it can reduce the computational power needed as well solving possible memory issues for large datasets. On the other hand, applying under sampling can lead to a loss of critical information. There are several strategies that can be followed to perform under sampling on a dataset. Each technique itself has some pros and cons. To select the best one, several tests need to be carried out.

The first approach consists of randomly reducing the number of instances of the majority class. This is a simple technique that removes some samples from the most frequent class by using a given percentage. Despite the simplicity of this method, removing random samples may lead to the loss of valuable information which may be preserved in the majority class.

To overcome the limitation of losing potential useful information by randomly removing samples from the majority class, Cluster Centroids should be applied [50]. This method under-samples the majority class by replacing a cluster of majority samples by the cluster centroid of a KMeans algorithm. This algorithm keeps N majority samples by fitting the KMeans algorithm with N clusters to the majority class and using the coordinates of the N clusters centroids as the new majority samples.

For this study, several experiments were conducted testing both methods. It is worth mentioning that cluster centroids was used for the final built models. In addition to experimenting with both methods, a visual inspection of cluster centroids method was conducted. Figure 4-6 shows an example of preserving information in the majority class. It can be observed that most of the dropped data points (blue points) belong to clusters that still have other instances after performing under sampling.



(a) **(b)**

Figure 4-6: Applying undersampling using Cluster Centroids.
(a) represents the data points before applying cluster centroids whereas (b) represents the final result of performing undersampling on the dataset.

4.4.1.2. Over sampling

Another technique that can be applied to handle imbalance problem is oversampling the minority class. Like under-sampling, it has some advantages and drawbacks. For example, increasing the number of instances avoids losing any potential information that exists in the majority class. However, if the data suffers from a severe imbalance issue, this technique alone can end up with overfitting, since the new synthesized data should be nearly similar to the already existing one.

Several methods could be applied to achieve oversampling. For instance, one can duplicate N instances of the minority class. Despite the simplicity of this technique, having duplicates in the dataset may not help the model learning any new information.

Another approach can be followed is by using SMOTE or Synthetic Minority Oversampling Technique [51]. SMOTE first selects a minority class instance a at random and finds its *k* nearest minority class neighbors. The synthetic instance is then created by choosing one of the *k* nearest neighbors b at random and connecting a and b to form a line segment in the feature space. The synthetic instances are generated as a convex combination of the two chosen instances a and b.

Section 4.5.2 represents the extensive experiments which were conducted to evaluate and test all the mentioned methods for handling the imbalanced problem. The final selection consists of a combination of both SMOTE for synthesizing samples from the minority class as well as cluster centroids to reduce the number of majority class. Figure 4-7 shows the final class distributions for all the complications.

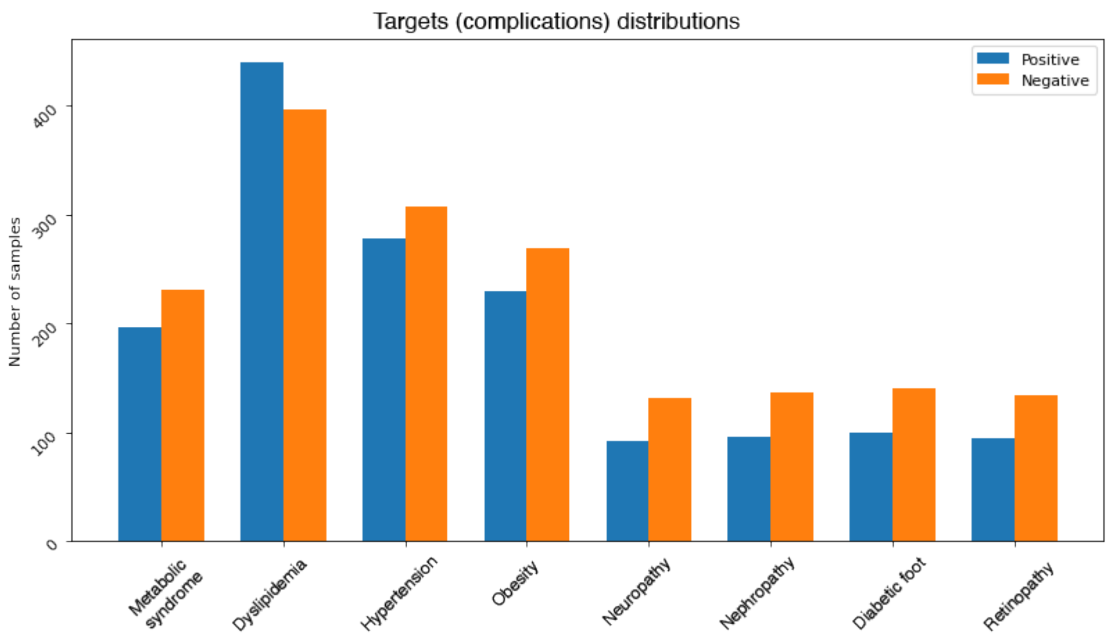


Figure 4-7: Classes distributions for each complication after handling imbalance problem.

Since the severity of the imbalance problem varies between the complications, we treated each complication independently. For instance, cluster centroids was applied alone on obesity without using any technique to oversample the minority class. However, since neuropathy, nephropathy, retinopathy and diabetic foot suffer from severe class imbalance issues, SMOTE as well as cluster centroids were applied to such complications. The selection of oversampling and undersampling percentages was carefully tested to reach the best possible performance as well as keeping and saving the original information in the dataset.

4.4.2 Data normalization

As described in section 4.1, the dataset in hand contains many numerical attributes, which were recorded with different measurement units and scales. Dealing with such attributes with the original scales could affect the performance of the model, since the variables that are measured at different scales do not contribute equally to the analysis and might end up creating a bias. For instance, an attribute that ranges between 0 and 1000 outweighs a variable that ranges between 0 and 10. Therefore, rescaling the features was a necessity for this research.

In this study, normalization was selected to rescale all numeric features. Normalization helps to scale all the attributes into a range between 0 and 1. Equation (5) describes the normalization formula. If a feature has a maximum value of 100 and a minimum value of 0, after normalizing using equation (5), the new values will be ranging between [0, 1]. The *Max* in the equation represents the maximum value in the attribute, whereas the *Min* is the minimum value in the attribute.

$$\frac{Value - Min}{Max} \tag{5}$$

4.4.3 Train and test splits

One required step to correctly evaluate the performance of a model is to split the dataset into training and testing sets. Figure 4-8 illustrates the splitting criteria. The purpose of the split is to train the model on the training set and then test its performance on a different, unseen data. The main goal of this split is to test if the model can generalize well to new data and its ability to avoid overfitting. Several percentage splits were evaluated in this research, the best split found to work properly is by using 80% of the total number of records for training and a 20% for testing. The results found using this method can be found in Appendix A.

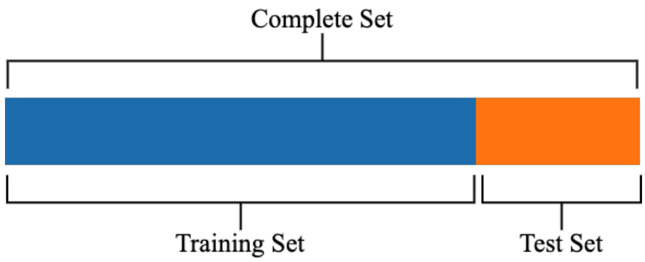


Figure 4-8: Train-Test Split

4.4.4 Cross validation and model training

Although using a simple train and test splits is useful for testing a ML model, it only evaluates the performance of the trained model on a small subset of the dataset which may produce biased metrics based on the specific test set used. A better technique that should be used is called K-fold cross validation.

The K-fold cross-validation (KCV) technique is one of the most widely used approaches to select a classifier and to evaluate its performance [52]. Figure 4-9 shows a detailed pictorial presentation of the data splitting using this technique with k=10. The dataset in hand was split into K folds. The $K-1$ folds were used to train and fine-tune the hyperparameters in the inner loop where the grid search algorithm [53] was employed. In the outer loop, the best hyperparameters and the test data were used to evaluate the model. Since the dataset contains imbalanced records, the stratified KCV [54] was used to preserve the percentage of samples for each class same as found in the dataset. Moreover, for a better evaluation, this process was repeated 10 times. The final performance metric was estimated using equation(6), where M is the final performance metric for the classifier, and $P_n \in \mathrm{R}$, n = 1,2,...,K is the performance metric for each fold.

$$M= \frac{1}{K} \times \sum_{n=1}^{K} P_n \tag{6}$$

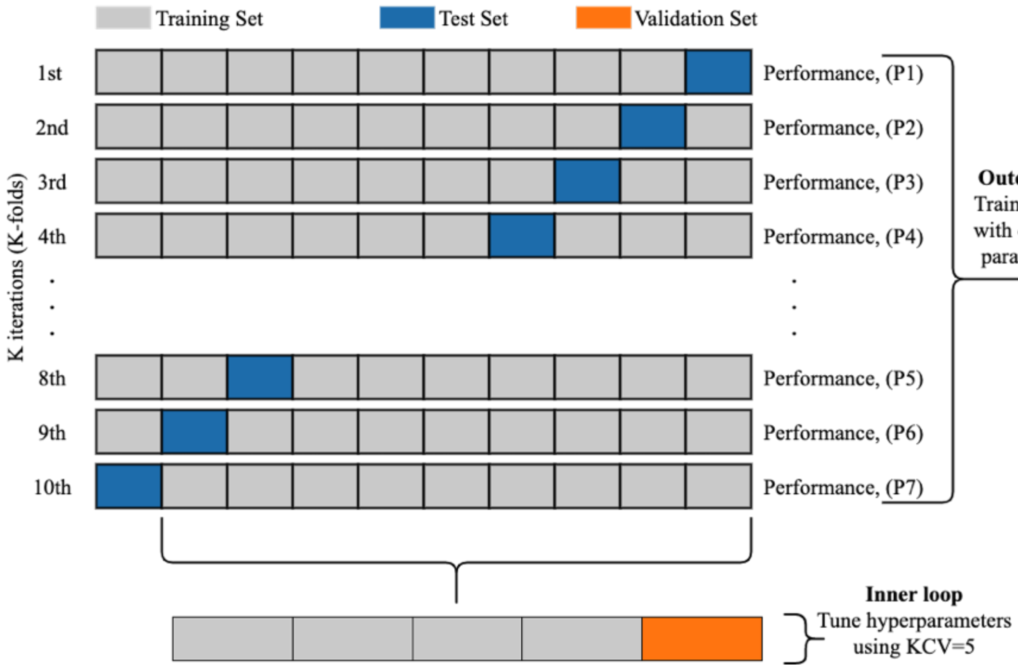


Figure 4-9: The use of KCV for both hyperparameters tuning and training [21].

As mentioned earlier, grid search [55] was employed in the inner loop to test a decent number of possible combinations for each estimator. For instance, if we have the following sets of parameters, a[1,2,3] and b[4,5,6], the grid search model works by testing the performance of each possible combination of the given sets. For example, it tests the model when a=1 and b=4, a=1 and b=5 or a=1 and b=6, etc. To better check the performance of each complication, a cross validation (with k=5) was used. Meaning for each possible combination, the data is divided into 5 folds, 4 for training and 1 for testing, making sure that all the 5 folds are used for testing purposes.

In this study, all the classifiers require a suitable tuning. For example, tuning a polynomial SVM model requires selecting a polynomial degree beforehand, such as 2 for quadratic polynomial. For that purpose, different polynomial degrees were tested and evaluated. Decision Tree (CART) also requires selecting and tuning different parameters such as the maximum depth of the tree, the minimum samples needed to perform a split and the minimum sample per leaf. Table 4-8 shows the hyperparameters that need to be tuned for the different models. Since repeated stratified KCV approach was used, it was not feasible to report the final selected hyperparameter values for all the built models.

Table 4-8: Different hyperparameters to be tuned using grid search technique

ML Model	Hyperparameters
LR	- The regularization parameter (C).
SVM	- The regularization parameter (C). - The kernel type to be used in the algorithm (Linear, Polynomial) - The degree of the polynomial kernel function (if used)
DT (CART)	- Maximum depth of the tree - The minimum number of samples per split - The minimum number of samples per leaf
RF	- The number of trees in the forest
AdaBoost	- The number of trees in the forest - The learning rate to decide the contribution of each classifier.
XGBoost	- The maximum depth of the tree - The number of trees - The booster: gbtree or dart

4.4.5 Feature selection

After training and building different models for each complication using all the algorithms explained in section 2.1.2, feature selection was applied on each classifier to select the top 5 and 10 features used to construct it. This step is important to improve our understanding of the built models especially since we are dealing with a dataset with more than 60 features. Understanding which features contribute most to the results, leads to establishing new models which can be built using only a small subset of the attributes available. It is also important to compare the features that are used to construct the top rules found in ARM with these selected features. The goal from this comparison is to find out if there is any shared knowledge between the dominant features found in each method, hence understanding the importance of such features and its contribution to the extracted knowledge.

There are many statistical methods to evaluate and select the dominant features in a dataset. For example, the variance can be calculated and utilized to remove features with low variance threshold. In this research, the selection of the top features relies on utilizing the parameters and equations built in each model. As mentioned before, this study uses two types of models. The first one is linear models, such as Logistic regression and Linear SVM, which rely on a linear equation to calculate the final output (or decision). For such estimators, the coefficients of the equations were used to determine the top features to select. Basically, the top features are given higher weights in the equation since they contribute more to the results. By considering the following example: $0.2x_1 + 0.01x_2 + 0x_3$ where three features were used to construct a LR model, it is noticed that attribute x_3 is not useful at all and does not affect the learning since its coefficient equals to 0. On the other hand, x_1 which has the maximum coefficient value contributes most to the learning.

The second type of models is tree-based models. Feature importance is calculated as the decrease in node impurity weighted by the probability of reaching that node. The node probability can be calculated by the number of samples that reach the node, divided by the total number of samples. The higher the value the more important the feature [56].

By utilizing the methods described above, the top 5 and 10 features were selected from the best model for each complication to further build reduced and specialized models. The selection of such number of features aims to reduce complexity and preserve the performance of the models. All the results are presented and explained in Chapter 5 in detail.

4.5. Conducted Experiments

To meet all the objectives of this study, an extensive number of experiments were conducted, which can be divided into two independent categories. The first group of experiments focuses on extracting different rules from the dataset using mainly ARM techniques. The second group of experiments, however, focuses on building supervised ML models to classify and predict different diabetes complications. The rest of this section describes these experiments in detail.

4.5.1 Association rule mining experiments

To understand the potential relations between attributes, several experiments should be carried out. The first conducted experiment aimed to study the possible relations between the diabetes complications themselves. Meaning that, ARM techniques were utilized to extract rules only from the 8 different complications.

The second conducted experiment utilized a larger subset of the data. For that, in the second experiment, the original categorical data which consists of Gender, Nationality and Diabetes type along with the complications were covered and studied. The aim of this experiment was mainly to enhance the understanding of the different relations, and to detect particularly any possible connections between the complications and the geographical attributes.

The last experiment carried out in this part utilized the whole number of attributes available which is 66 attributes. The purpose of this experiment was to check any possible relations in the complete dataset between the geographical attributes, medical tests and the eight complications. Table 4-9 summarizes all the experiments. It is worth mentioning that both Apriori as well as FP-Growth algorithms were utilized to build and extract the rules.

Table 4-9: ARM Experiments

Exp. #	Attributes Included	ARM Algorithms
1	8 Targets (Complications)	Apriori, FP-Growth
2	8 Targets + Gender, Nationality Name, Diabetes Type	Apriori, FP-Growth
3	All Attributes	FP-Growth

After investigating the possible relations between the available attributes, an analysis was carried out to select the top attributes used in extracting the rules. The top attributes can be defined as the most frequently used attributes to create the rules.

4.5.2 Classification and prediction experiments

Since this study deals with 8 independent targets, meaning that a patient could suffer from one or more complications at the same time, we decided to build binary classifiers. Such classifiers can predict the occurrence of a single specific complication. Moreover, and as explained in section 2.1.2, to reach the best possible performance, several algorithms were investigated and constructed such as Logistic Regression, Linear SVM, Polynomial SVM, CART DT, Random Forest, AdaBoost and XGBoost. Each algorithm was used to build 8 prediction models, one for each complication.

In this research, an extensive number of experiments were conducted. All of which use all the algorithms mentioned above. As described earlier in section 4.2.2, after calculating and comparing the RMSE value of different approaches for data imputation, which includes a simple mean substitution, the use of *k*-NN [47] algorithm as well as using MissForest [27], the later was used to fill any missing value for all the experiments to follow.

After handling the missing values problem, multiple experiments were conducted for testing different balancing techniques. The first set of experiments tested only one method at a time. For instance, the first experiment duplicated some random records for the minority class. By following this simple approach, it was noticed that the performance of the models did not enhance much. The second experiment was conducted by utilizing SMOTE method [51] to synthesize more data for the minority class. Generally speaking, a better performance was achieved. The third experiment was carried out by utilizing cluster centroids [50] approach. In this experiment we also noticed achieving better performance.

The second set of experiments focused on utilizing both oversampling and undersampling techniques together to boost the performance of the models. For that, both removing samples for the majority class from the dataset at random as well as utilizing SMOTE technique to increase the number of minority class were used. Another combination built was by using SMOTE followed by using cluster centroids. It is worth mentioning that the best performance reached was by using the later approach.

After selecting the balancing method which consists of both SMOTE and cluster centroids techniques, feature selection was also applied to select and build models using the top 5 and 10 features along with building models that utilize all the attributes available.

Table 4-10 summarizes all the experiments conducted. Due to the large number of experiments, the number of algorithms used as well as the number of complications, section 5.3 only shows the performance results of the last experiment in Table 4-10, covering both all the attributes as well as feature selection.

Table 4-10: Supervised ML Experiments

Exp #	Imbalance Method
1	Duplicate random records from the minority classes
2	Using SMOTE for oversampling
3	Under sampling using Cluster centroids (without oversampling)
4	Randomly remove samples from the majority class and use SMOTE for minority class
5	Cluster centroids with SMOTE

Chapter 5. Experimental Results and Discussion

In this chapter, the results of the various conducted experiments are discussed in detail. Before that, we introduce the performance metrics that were utilized to evaluate the models. Furthermore, this chapter presents and discusses the top extracted rules generated in section 4.5.1. Lastly, the performance of the supervised models in the experiments is presented and discussed.

5.1. Performance Measure

In this section, two metrics groups are discussed. This is due to the fact that ARM uses different metrics than the ones used in classification and prediction models. The rest of this section explains the two groups in detail by clarifying the equations used for each metric as well as providing several examples for illustration purposes.

5.1.1 ARM

As mentioned before, ARM extracts frequent patterns among sets of data. The general concept of ARM depends on three main measurements: Support, Confidence and Lift.

Before explaining the metrics, let's consider the example in Table 5-1. As shown in the example, different transactions are available (T1 to T5). Each transaction has its own items, for instance, T1 consists of 1, 3 and 4. This table is used in the following subsections as base for our discussion.

Table 5-1: ARM Example

TID	Items
T1	{1, 3, 4}
T2	{2, 3, 5}
T3	{1, 2, 3, 5}
T4	{2, 5}
T5	{1, 3, 5}

It is worth mentioning that an itemset simply means a group of items, for instance, $\{1\}$, $\{2\}$, $\{3\}$ …etc., can be considered as itemsets with a single value. Figure 5-1 represents an example of a rule format. It is worth mentioning that A stands for antecedent and B stands for consequent. Both A and B are lists of items that represent a rule.

Figure 5-1: Rule format in ARM

5.1.1.1. Support

"Support" simply is an indicator of the popularity of an itemset in the dataset [57]. Equation (7) can be used to calculate the Support for an item A.

$$\text{Support(A)} = \frac{\text{Records containing item A}}{\text{Total number of records}} \tag{7}$$

As mentioned in section 2.1.1 ARM identifies the frequent individual items in the dataset and extends them to larger and larger itemsets as long as those item sets appear sufficiently often in the dataset. For instance, the first iteration of Apriori algorithm starts by identifying item sets of size of 1 and calculating their support values. Table 5-2 represents the calculation of support for the first iteration of Apriori algorithm. A predefined support threshold should be provided. For this example, let's define a minimum support of 0.3. By using equation (7), itemset {1} has a support of 0.6, this is because if we check Table 5-1, we can notice that the number of transactions contains {1} is 3, and the total number of transaction is 5, hence 3/5=0.6.

Table 5-2: Iteration #1 in Apriori Algorithm

Itemset	**Items**
{1}	0.6
{2}	0.6
{3}	0.8
{4}	0.2
{5}	0.8

Since the predefined threshold for support is 0.3, this means any itemset that has a smaller support will not be considered for further analysis. In this example, since itemset {4} has support of 0.2, it is eliminated. After several iterations, we will end up with the final itemsets that all have a support value > 0.3.

5.1.1.2. Confidence

After identifying all the possible item sets, the "confidence" metric can be utilized to define the final rules. Confidence is an indication of how often the rule was found to be true [57]. To calculate the confidence, we can use equation (8).

$$Confidence\ (A \Longrightarrow B) = \frac{Support(A \cup B)}{Support(A)} \tag{8}$$

As noticed from equation (8), to calculate the confidence, the support metric should be utilized as well.

By using equation (8), the confidence of rule $\{1\} \rightarrow \{3\}$ is 1. This is because support ({1,3}) = 0.6 and support ({1}) = 0.6. This represents that there is a noticeable relation between {1} and {3}.

5.1.1.3. Lift

The third measurement, "lift", provides information about the degree to which two attributes in a dataset depend on each other [57]. For example, if Lift "$A \rightarrow B$" is >1 then this is an indication of a possible relation between A and B. In other words, lift can reflect the importance of a rule. Equation (9) represents the formula to calculate the lift metric.

$$Lift\ (A \Longrightarrow B) = \frac{Support(A \cup B)}{Support(A) \times Support(B)} \tag{9}$$

By using equation (9), the lift of rule $\{1\} \rightarrow \{3\}$ is 1.25. This is because support ({1,3}) = 0.6, support ({1}) = 0.6 and support ({3}) = 0.8. Since the lift value exceeds 1, this indicates the possibility of a relation between {1} and {3}.

5.1.2 Supervised ML

As mentioned earlier, since supervised ML models rely on a different set of evaluation metrics than the ones used in ARM, this section explains in detail this second set of metrics. The set includes: Accuracy score, Precision and Recall scores, F1-Score as well as AUC score.

Before discussing the evaluation metrics, let's explain some terms in the confusion matrix shown in Table 5-3. This confusion matrix summarizes the results in a way that easily reflects the performance of a model to classify or predict a target class. For instance, the True Positive term, represents the values that were classified to 'Yes' and they are actually 'Yes'. False Positive on the other hand means that the model predicts the target as 'Yes', although the actual value of it is 'No'. In the same way, False Negative means that the model predicts the class as 'No', however, the real class is 'Yes'. Finally, the True Negative represents that both the model and the actual value of the target are 'No'. By utilizing the confusion matrix, Table 5-4 represents an example that is going to be used to explain the evaluation metrics for the rest of this section.

Table 5-3: Confusion Matrix Template

Actual \ Predicted	Yes	No
Yes	True Positive (TP)	False Negative (FN)
No	False Positive (FP)	True Negative (TN)

Table 5-4: Classification Example

Actual \ Predicted	Yes	No
Yes	20000	0
No	1000	1

From Table 5-4, it is noticed that the model is biased towards class 'Yes' since it classifies almost all the data instances to that class. To evaluate the performance of such a model, several evaluation metrics should be used.

5.1.2.1. *Accuracy*

The first evaluation metric is "Accuracy". The classification accuracy is defined as the percentage of instances classified as their true class labels [26]. It is also known as the recognition rate since it resembles the percentage of test set instances that are correctly classified. Although it is one of the most used evaluation metrics, it does not accurately describe the model performance in case of unbalanced datasets. Hence, it is important to use other techniques as well in this case. The accuracy of a classifier can be computed using equation (10), where TP is the True Positive, TN is the True Negative, FP is the False Positive and finally FN is the False Negative values.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \times 100\% \tag{10}$$

By utilizing equation (10) to calculate the accuracy for the given example in Table 5-4, the accuracy score can be calculated as:

$$\text{Accuracy } = \frac{20000 + 1}{20000 + 1 + 1000 + 0} \times 100\% = 95.2383$$

By looking at the accuracy results, one can think it is a good model. In fact, by inspecting the confusion matrix a little bit more, it is noticed that this classifier predicts almost all the instances as 'Yes' regardless of whether they are actually 'Yes' or 'No'. Therefore, this model is biased towards one class ('Yes'). This bias issue can be severe in some applications such as the one in hand, where classifying ill patients as healthy may result in death over long terms. To avoid such a limitation, other metrics besides accuracy can be utilized to help assisting the performance of the models.

5.1.2.2. Precision

Another interesting metric to use is "Precision". Precision is defined as the percentage of instances that were classified as X and are actually X. Therefore, precision is sometimes referred to as the exactness of the classifier. Equation (11) can be utilized to calculate the Precision.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{11}$$

By utilizing the above equation, the precision for the example in Table 5-4 for class 'Yes' can be calculated as follows:

$$\text{Precision} = \frac{20000}{20000 + 1000} = 0.9524$$

whereas the precision for class 'No' can be calculated as follows:

$$\text{Precision} = \frac{1}{1 + 0} = 1$$

5.1.2.3. Recall

The third evaluation metric to be used in this study is "Recall". Recall is defined as the percentage of instances that are actually X and were predicted as X by the classifier. It is worth mentioning that recall score is an important metric in case of medical classification problems since it takes into consideration the number of false negative cases [26]. It also can help in reflecting the reality when the data is unbalanced.

To calculate the recall score, equation (12) can be utilized as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{12}$$

We can use the previous equation to calculate the recall score for 'Yes' class as follows:

$$\text{Recall} = \frac{20000}{20000 + 0} = 1$$

Following the same approach, the recall score for class 'No' can be calculated as follows:

$$\text{Recall} = \frac{1}{1 + 1000} = 0.000999$$

By looking at the results above, it is obvious that the model can classify class 'Yes' correctly, however, the model performs poorly when it comes to classifying class 'No'. From this observation we can notice the importance of utilizing such metric in this research.

5.1.2.4. F1-Score

The fourth metric to utilize is F-score. It is the harmonic mean of precision and recall; therefore, it incorporates both the precision and recall in a single value. Hence, F-score is maximum at the value of 1 and minimum at the value of 0 [58]. Equation (13) can be used to calculate F1-score.

$$\text{F1-Score} = \frac{2 \times \text{Precesion} \times \text{Recall}}{\text{Precesion} + \text{Recall}} \tag{13}$$

By using the values of Table 5-4, F1-score for class 'Yes' can be calculated as follows:

$$\text{F1-Score} = \frac{2 \times 0.9524 \times 1}{0.9524 + 1} = 0.9756$$

The F1-score for class 'No' however can be calculated as:

$$\text{F1-Score} = \frac{2 \times 1 \times 0.000999}{1 + 0.000999} = 0.0020$$

The final average value of F-Score can be computed as:

$$\text{Averaged F1-Score} = \frac{0.9756 + 0.0020}{2} = 0.4888$$

From the previous calculations, it can be found that the average F1-score can elaborate the problem as well. Furthermore, it can provide us with a good understanding of the model performance by including both precision and recall scores in one equation. As a result, F1-score will be used along with the accuracy score to measure and evaluate the performance of each model.

5.2. Generated Rules by ARM

As described earlier in section 4.5.1, there are mainly three major experiments conducted in this research to study different aspects of the dataset using ARM. The first experiment includes only the 8 complications. The second experiment expands to cover the categorical attributes namely Gender, Nationality Name and Diabetes Type. Finally, the third and last experiment includes all the available attributes in the dataset.

For the ARM algorithms, both Apriori algorithm as well as FP-Growth were utilized. Support, Confidence and Lift were all used to evaluate the extracted rules. Table 5-5 summarizes all the conducted experiments. Moreover, it shows the minimum support threshold which was selected for each experiment.

Table 5-5: ARM Experiments Summary

Experiment #	Algorithm Utilized	Dataset Attributes	Min. Support Threshold
1	Apriori	8 Complications	0.17
2	Apriori	8 Complications + Categorical	0.4
3	FP-Growth	8 Complications	0.17
4	FP-Growth	8 Complications + Categorical	0.4
5	FP-Growth	All Available attributes	0.3
6	FP-Growth	All Except ratios attributes	0.3

To select the best rules, we started with a minimum support threshold of 0.05. Afterwards, the threshold value was increased gradually. For all the experiments, the amount of increase in the support value depends mainly on two factors, the first one is to make sure to generate the best rules possible, and the second factor is to avoid memory limitations. Memory limitations can be produced due to the huge number of the possible item sets, hence the increase of the memory needed to store them while processing. To avoid such a problem, the minimum support threshold found to work properly using the complete attributes set was 0.3. It is worth mentioning that a smaller support value was needed when studying the complications set, that is because the number of occurrences (the support) of the itemsets that represent the complications was limited, hence, it was necessary to reduce the thresholds for such experiments. It is important to mention that the lift metric with a minimum value of 1 was selected to generate all the possible rules. As mentioned in section 5.1.1.3, lift provides information about the degree to which two attributes in a dataset depend on each other, meaning that it reflects the importance of the generated rule. The higher the value (>1) the stronger the extracted rule. Moreover, the confidence was used to order the extracted rule. According to section 5.1.1.2, the confidence is an indication of how often the rule was found to be true, in other words, it reflects the accuracy of that rule.

From applying all the experiments, we can observe that both Apriori algorithm and FP-Growth can find the exact same results when using the same dataset and tuned in the same way. We believe this is because both algorithms rely on the same mathematical equations to calculate the ‘strength’ of a rule. In fact, the only difference between the two algorithms is the way of building the item sets table. Another observation is that FP-Growth is much faster than Apriori algorithm and can be used with larger datasets that have a larger number of attributes. This is the same reason behind using only the FP-Growth algorithm with all the available attributes. Since the results of the Apriori algorithm are identical to what was found in FP-Growth experiments, they will not be presented. Table 5-6, Table 5-7, Table 5-8, and Table 5-9 represent the top 10 rules resulting from experiments number 3, 4, 5 and 6 respectively.

From Table 5-6, we can notice that the strongest relations are between metabolic syndrome, hypertension, dyslipidemia, and obesity. This is due to the large number of rules that include these complications together. For instance, Metabolic syndrome and hypertension are strongly connected with each other, meaning that if a patient develops hypertension, it is more likely to develop metabolic syndrome as well. Moreover, if a patient suffers from both metabolic syndrome and hypertension, they are more likely to suffer from dyslipidemia. In the same way, it can be noticed that obesity is connected to dyslipidemia and hypertension.

Table 5-6: Generated Rules by FP-Growth for the complications' subset only

Antecedents	Consequents	Support	Confidence	Lift
{'MetabolicSyndrome', 'Hypertension'}	{'Dyslipidemia'}	0.1717	0.9058	1.4589
{'Hypertension'}	{'Dyslipidemia'}	0.3434	0.8741	1.4079
{'MetabolicSyndrome'}	{'Dyslipidemia'}	0.2418	0.8585	1.3828
{'Obesity'}	{'Dyslipidemia'}	0.2788	0.8388	1.3511
{'MetabolicSyndrome', 'Dyslipidemia'}	{'Hypertension'}	0.1717	0.7102	1.8079
{'MetabolicSyndrome'}	{'Hypertension'}	0.1896	0.6732	1.7135
{'MetabolicSyndrome'}	{'Dyslipidemia', 'Hypertension'}	0.1717	0.6098	1.7756
{'Dyslipidemia'}	{'Hypertension'}	0.3434	0.5531	1.4079
{'Obesity'}	{'Hypertension'}	0.1799	0.5413	1.3779
{'Dyslipidemia', 'Hypertension'}	{'MetabolicSyndrome}	0.1717	0.5000	1.7756

In Table 5-7, the most connected attributes can be found when a patient is from the UAE, has Type 2 Adult Onset and suffers from dyslipidemia. From the table we can notice that patients from UAE and having Diabetes Type 2 are probably more likely to be at risk of having or developing dyslipidemia. The opposite rule also exists which emphasizes the importance of the previous rule.

Table 5-7: Generated Rules by FP-Growth for the complications and categorical attributes

Antecedents	Consequents	Support	Confidence	Lift
{'Dyslipidemia'}	{'DiabetesType_Type 2 Adult Onset'}	0.5824	0.9381	1.1104
{'NationalityName_United Arab Emirate', 'Dyslipidemia'}	{'DiabetesType_Type 2 Adult Onset'}	0.5330	0.9372	1.1094
{' Gender _Female'}	{'NationalityName_United Arab Emirate'}	0.4657	0.9187	1.0027
{' Gender _Male'}	{'DiabetesType_Type 2 Adult Onset'}	0.4341	0.8802	1.0420
{'Dyslipidemia'}	{'DiabetesType_Type 2 Adult Onset', 'NationalityName_United Arab Emirate'}	0.5330	0.8584	1.1139
{'DiabetesType_Type 2 Adult Onset', 'NationalityName_United Arab Emirate'}	{'Dyslipidemia'}	0.5330	0.6916	1.1139
{'DiabetesType_Type 2 Adult Onset'}	{'Dyslipidemia'}	0.5824	0.6894	1.1104
{'DiabetesType_Type 2 Adult Onset'}	{'NationalityName_United Arab Emirate', 'Dyslipidemia'}	0.5330	0.6309	1.1094

{'DiabetesType_Type 2 Adult Onset'}	{'Gender_Male'}	0.4341	0.5138	1.0420
{'NationalityName_United Arab Emirate'}	{'Gender_Female'}	0.4657	0.5082	1.0027

From Table 5-8, it is observed that EAG, HbA1c (%) and HbA1c (mmol) are the most related attributes. Furthermore, there are many rules that describe the bidirectional relation between these attributes. For instance, if the value of EAG ranges between (9.0, 12.0] and the percentage of HbA1c ranges between (7.0, 8.9], the expected value of HbA1c (in mmol) is (53.0, 74.0]. Such rules can be considered a good starting point. However, in most cases such rules are not informative enough and can be considered as a given, especially for HbA1c (%) and HbA1c (mmol). To avoid any unnecessary calculations, and to extract more interesting results, experiment 6 has removed any ratio attributes.Table 5-9 represents the top 10 rules found in experiment 6. HbA1c (mmol), EAG, Nationality (UAE), Age and others are all found in the top rules. As it is observed from the table, different relations can be found in the generated rules, such as between the following itemsets: {Nationality (UAE), TestHbA1c (mmol) (68.0,141.0]} and {TestEAG (11.0,21.0]}.

Table 5-8: Generated Rules by FP-Growth for the complete dataset

Antecedents	**Consequents**	**Support**	**Confidence**	**Lift**
{TestEAG(9.0,12.0], TestHbA1c (%)(7.0,8.9]}	{TestHbA1c (mmol)(53.0,74.0]}	0.3187	1	3.1245
{TestHbA1c (mmol)(53.0,74.0]}	{TestHbA1c (%)(7.0,8.9]}	0.3201	1	3.0083
{TestEAG(9.0,12.0], TestHbA1c (mmol)(53.0,74.0]}	{TestHbA1c (%)(7.0,8.9]}	0.3187	1	3.0083
{TestHbA1c (%)<=7.0}	{TestHbA1c (mmol)<=53.0}	0.3324	1	3.0083
{TestHbA1c (mmol)<=53.0}	{TestHbA1c (%)<=7.0}	0.3324	1	3.0083
{TestHbA1c (%)<=7.0, NationalityName_United Arab Emirate}	{TestHbA1c (mmol)<=53.0}	0.3146	1	3.0083
{NationalityName_United Arab Emirate, TestHbA1c (mmol)<=53.0}	{TestHbA1c (%)<=7.0}	0.3146	1	3.0083
{TestEAG<=9.0}	{TestHbA1c (mmol)<=53.0}	0.3063	1	3.0083
{TestEAG<=9.0}	{TestHbA1c (%)<=7.0}	0.3063	1	3.0083
{TestHbA1c (%)<=7.0, TestEAG<=9.0}	{TestHbA1c (mmol)<=53.0}	0.3063	1	3.0083

Table 5-9: Generated Rules by FP-Growth for the complete dataset. (Except Ratio Attributes)

Antecedents	Consequents	Support	Confidence	Lift
{TestHbA1c (mmol) (68.0,141.0]}	{TestEAG(11.0,21.0]}	0.3365	1	2.9004
{NationalityName_United Arab Emirate , TestHbA1c (mmol) (68.0,141.0]}	{TestEAG(11.0,21.0]}	0.3036	1	2.9004
{TestEAG(11.0,21.0]}	{TestHbA1c (mmol) (68.0,141.0]}	0.3365	0.9761	2.9004
{TestHbA1c (mmol) (57.0,68.0]}	{TestEAG(9.0,11.0]}	0.3242	0.9752	2.4651
{NationalityName_United Arab Emirate , TestEAG(11.0,21.0]}	{TestHbA1c (mmol) (68.0,141.0]}	0.3036	0.9736	2.8929
{TestHbA1c (mmol)<=57.0}	{NationalityName_United Arab Emirate}	0.3173	0.9585	1.0462
{Age(57.0,91.0]}	{DiabetesType_Type 2 Adult Onset}	0.3187	0.9508	1.1255
{TestInsulin(10.1,18.0]}	{NationalityName_United Arab Emirate}	0.3173	0.9506	1.0376
{Dyslipidemia , Hypertension}	{DiabetesType_Type 2 Adult Onset}	0.3255	0.948	1.1222
{Test25-OH Vitamin D Total(56.6,226.0]}	{NationalityName_United Arab Emirate}	0.3173	0.9467	1.0333

5.3. Prediction Performance

As discussed in section 4.5.2, many experiments were conducted to build supervised machine learning models for each complication. Moreover, the models were used to select the top features that contributed most to the results to build more specialized models. The rest of this section provides the evaluation metrics (Accuracy and F1-Score) for each complication resulted from using repeated stratified K-fold cross validation (RSCV). Section 5.4 explains the findings and discusses some observations.

5.3.1 Baseline classifiers

To understand the actual performance and the effect of training all the algorithms, some baselines need to be constructed first. For that, we established some simple classifiers to compare them with the trained models. The job for each classifier (ZeroR) is simply to predict all the instances as the majority class. After accomplishing this step, the accuracy and F1-Score were calculated for all these basic estimators. The performance of the baseline classifiers is reported in Table 5-10.

Table 5-10: Base-Line Performance

Algorithm	Accuracy	F1-Score
Metabolic syndrome	0.5397	0.3505
Dyslipidemia	0.5263	0.3448
Hypertension	0.5256	0.3445
Obesity	0.5402	0.3507
Neuropathy	0.5874	0.3701
Nephropathy	0.588	0.3703
Diabetic Foot	0.5875	0.3701
Retinopathy	0.5877	0.3702

The main reason behind establishing the baseline predictors is that the dataset in hand is used for the first time in this research, as a result, there are no previous performance scores to compare with for this particular dataset. A comparison between the baseline and the trained classifiers is conducted in section 5.4.

5.3.2 Metabolic syndrome

As it can be observed from Figure 5-2 below, LR achieved the best results. Both SVM Linear and RF achieved comparable results as well.

Table 5-11 below provides an accurate representation of the model performance. We can notice that the top three models for predicting the complication are LR, SVM Linear and RF, with a maximum difference of 2% when using all the attributes. For a better understanding of such close results, we calculated the confidence interval for all the three algorithms. The confidence interval (CI) displays the probability that a parameter will fall between a pair of values around the mean [59]. With a confidence of 95%, we found that the CI of accuracy for LR, SVM Linear, and RF are [0.765682, 0.777197], [0.758188, 0.768777], and [0.74712, 0.75806] respectively. It is noticed from the calculations that the performance difference between the three algorithms is insignificant. Although this can be considered as a minor difference, the gap between these algorithms is expanded when using selected features. This observation may indicate that LR better identifies the best features.

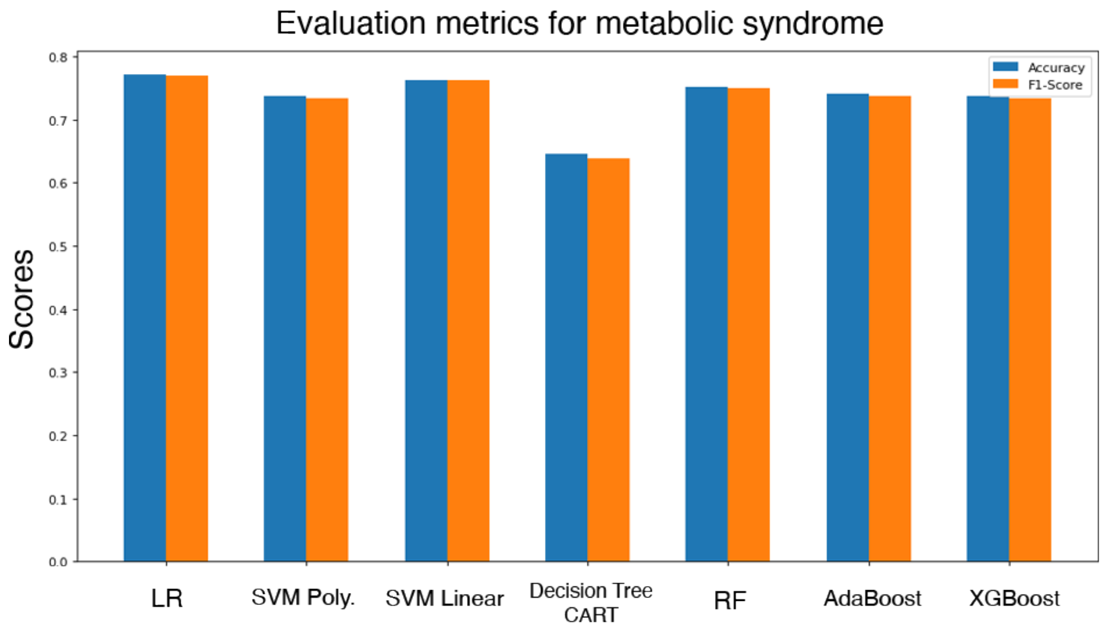


Figure 5-2: Metabolic Syndrome - Evaluation Performance using all attributes

Table 5-11: Metabolic Syndrome - Evaluation Performance

Algorithms	All Attributes		Top 10		Top 5	
	Accuracy	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
LR	**0.771**	**0.77**	**0.735**	**0.734**	**0.756**	**0.754**
SVM Poly.	0.737	0.734	-	-	-	-
SVM Linear	0.763	0.762	0.746	0.744	0.69	0.684
CART (DT)	0.646	0.639	0.649	0.643	0.651	0.646
RF	0.753	0.75	0.703	0.7	0.682	0.679
AdaBoost	0.74	0.738	0.698	0.694	0.673	0.67
XGBoost	0.738	0.735	0.703	0.7	0.707	0.704

Figure 5-3 represents a visual comparison between all the attribute sets used to build the best performing model which is LR for this complication. As noticed, applying feature selection on the data resulted in a slight degradation in the performance. Another interesting observation is that using only 5 features results in a 2% better performance than using the top 10 features, which indicates that reducing the model's complexity overweighs the the added value from extra features.

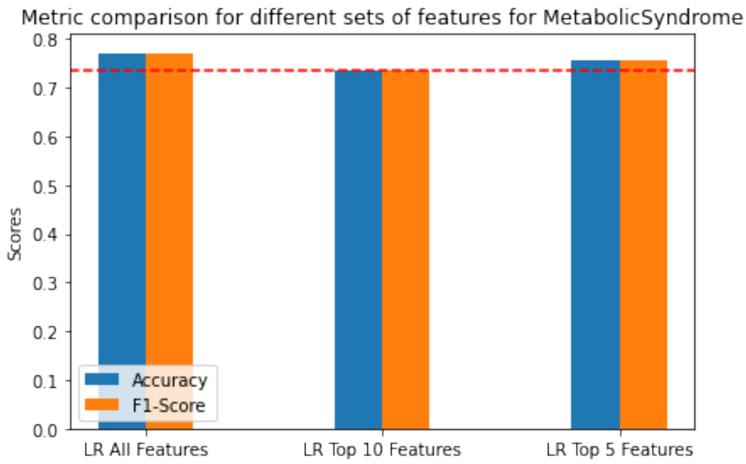


Figure 5-3: Metabolic Syndrome - Comparison between all feature sets

5.3.3 Dyslipidemia

For dyslipidemia, both RF and XGBoost achieved nearly the same results. Figure 5-4 shows a pictorial presentation of the evaluation metrics using all the attributes available to build the models.

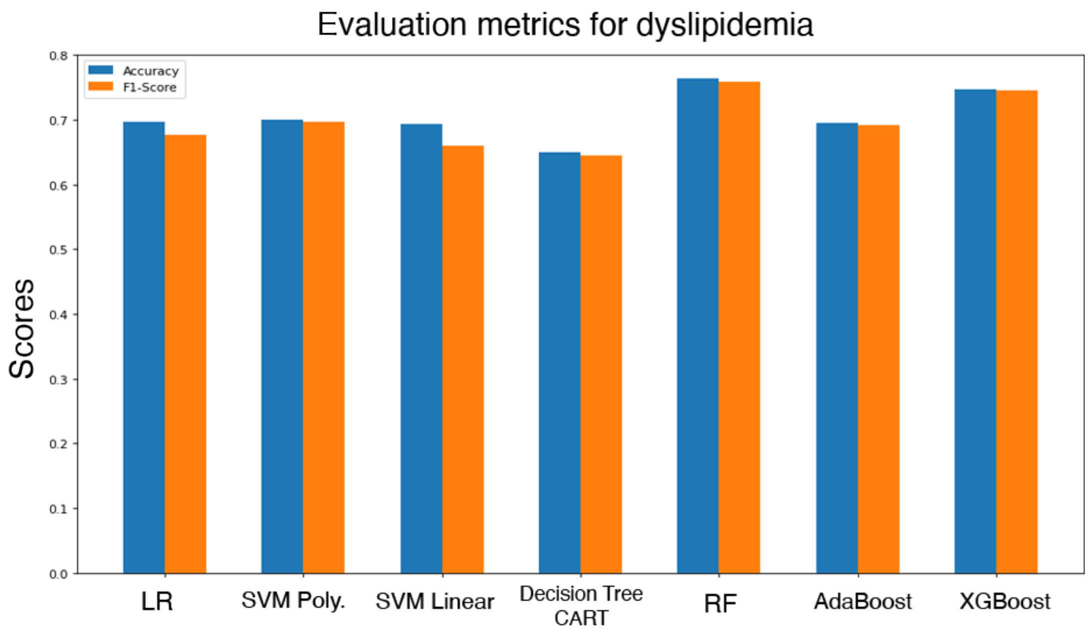


Figure 5-4: Dyslipidemia - Evaluation Performance using all attributes

Table 5-12 below shows the performance evaluation of using different attribute sets to train several models to classify dyslipidemia cases. Random forest achieved the best accuracy and F1-scores with 76.3% and 75.9%, respectively. However, XGBoost was also able to achieve comparable results with a difference of only 1.7% when using all the attributes. With 95% confidence, the confidence intervals of the accuracy score for

RF and XGBoost are [0.760182, 0.766158] and [0.744074, 0.750438], respectively. According to these results, and with a confidence of 95%, the difference in accuracy scores between these two algorithms will not exceed 2%. Moreover, both models achieved almost identical results when applying feature selection. This indicates the possibility of using either model for such a problem.

Table 5-12: Dyslipidemia - Evaluation Performance

Algorithms	**All Attributes**		**Top 10**		**Top 5**	
	Accuracy	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
LR	0.697	0.677	0.694	0.666	0.695	0.659
SVM Poly.	0.699	0.696	-	-	-	-
SVM Linear	0.693	0.66	0.685	0.65	0.691	0.654
CART (DT)	0.649	0.646	0.649	0.646	0.637	0.634
RF	**0.763**	**0.759**	**0.713**	**0.71**	**0.677**	**0.675**
AdaBoost	0.695	0.692	0.66	0.658	0.621	0.618
XGBoost	0.747	0.745	0.709	0.706	0.679	0.676

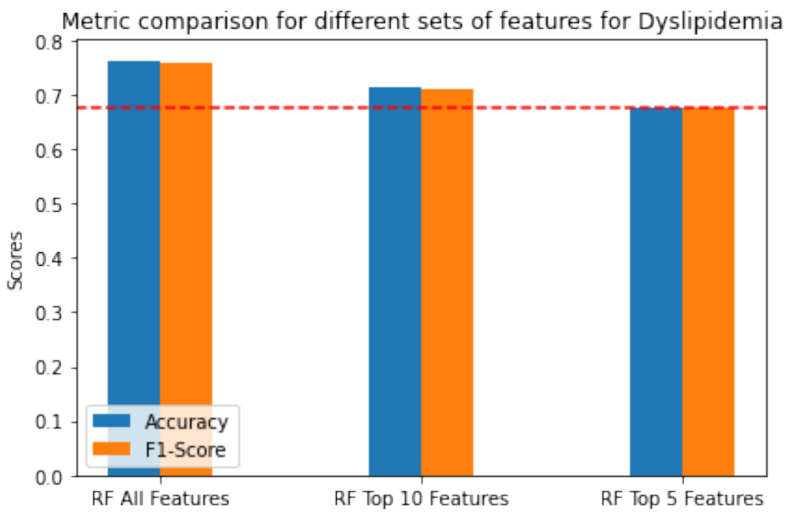


Figure 5-5: Dyslipidemia - Comparison between all feature sets

Since RF achieves the best performance, it was used to compare the metrics when using all the attributes compared to using only 10 or 5 features for training as shown in Figure 5-5. A clear degradation in performance is noticed, especially when using the top 5 features with a difference of 8.6% in performance.

5.3.4 Hypertension

The best performing model for hypertension is RF. However, the results in Figure 5-6 shows a very competitive performance for LR, SVM, RF, and XGBoost with differences not exceeding 2%.

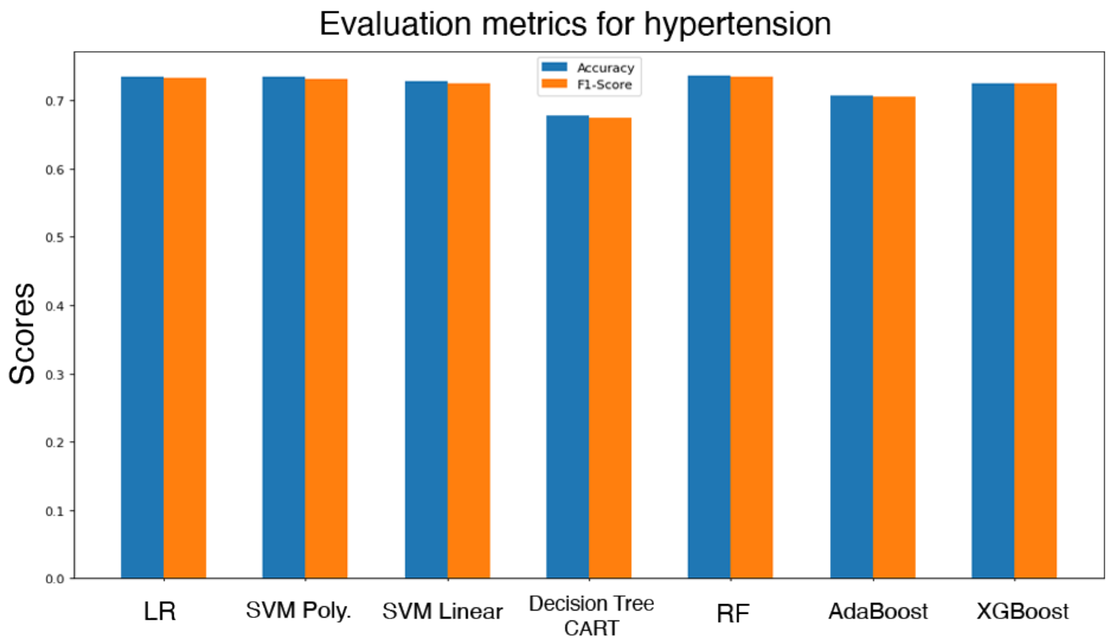


Figure 5-6: Hypertension - Evaluation Performance using all attributes

Table 5-13 below provides an exact representation of the performance for each model. Although the RF model achieved the best results, LR, SVM (Poly. & Linear), and XGBoost also achieved comparable results. Hence, for a better analysis, we calculated the confidence interval (with 95% confidence) of the accuracy scores for all these models as follows: LR [0.730736, 0.738873], SVM Poly. [0.730313, 0.738185], SVM Linear [0.723368, 0.73188], RF [0.731093, 0.740035], XGBoost [0.721191, 0.729627]. The CI results show insignificant differences in the performance of all the algorithms with a maximum accuracy difference of 1.8% between RF and XGBoost.

Table 5-13: Hypertension - Evaluation Performance

Algorithms	All Attributes		Top 10		Top 5	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
LR	0.735	0.732	0.726	0.723	0.702	0.698
SVM Poly.	0.734	0.732	-	-	-	-
SVM Linear	0.728	0.725	0.725	0.723	0.703	0.699

CART (DT)	0.678	0.675	0.676	0.673	0.687	0.685
RF	**0.736**	**0.734**	**0.716**	**0.715**	**0.7**	**0.699**
AdaBoost	0.707	0.705	0.673	0.67	0.607	0.604
XGBoost	0.725	0.724	0.701	0.699	0.689	0.688

According to Figure 5-7, the difference in accuracy metric when using the top 5 features compared to using all the attributes is only 3.6%. In this case, the reduction of the model's complexity overweighs the the added value from extra features.

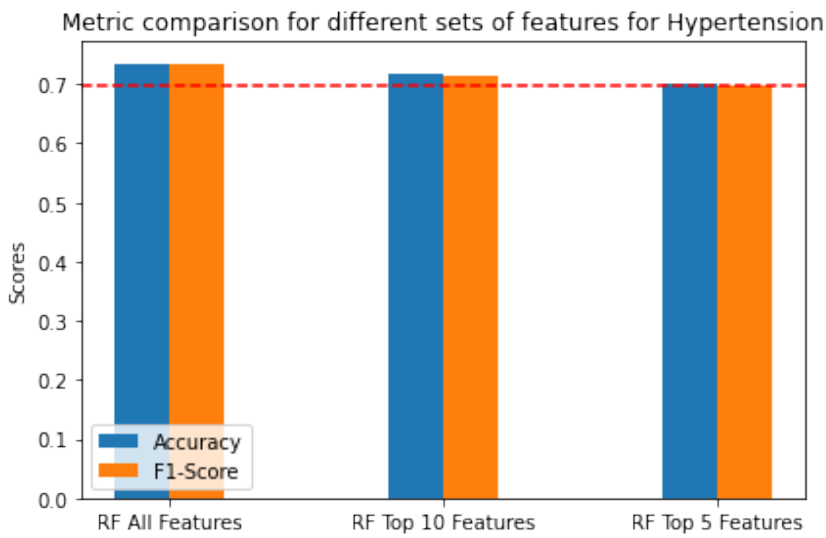


Figure 5-7: Hypertension - Comparison between all feature sets

5.3.5 Obesity

Training the models to predict obesity achieved better results than the ones mentioned before. For example, the minimum accuracy reported was 75.2% using the AdaBoost model. On the other hand, the maximum accuracy score was reached by RF with a value of 80%. The F1-score was 79.9% using the same algorithm. Figure 5-8 shows the accuracy as well as F1-Score for all algorithms.

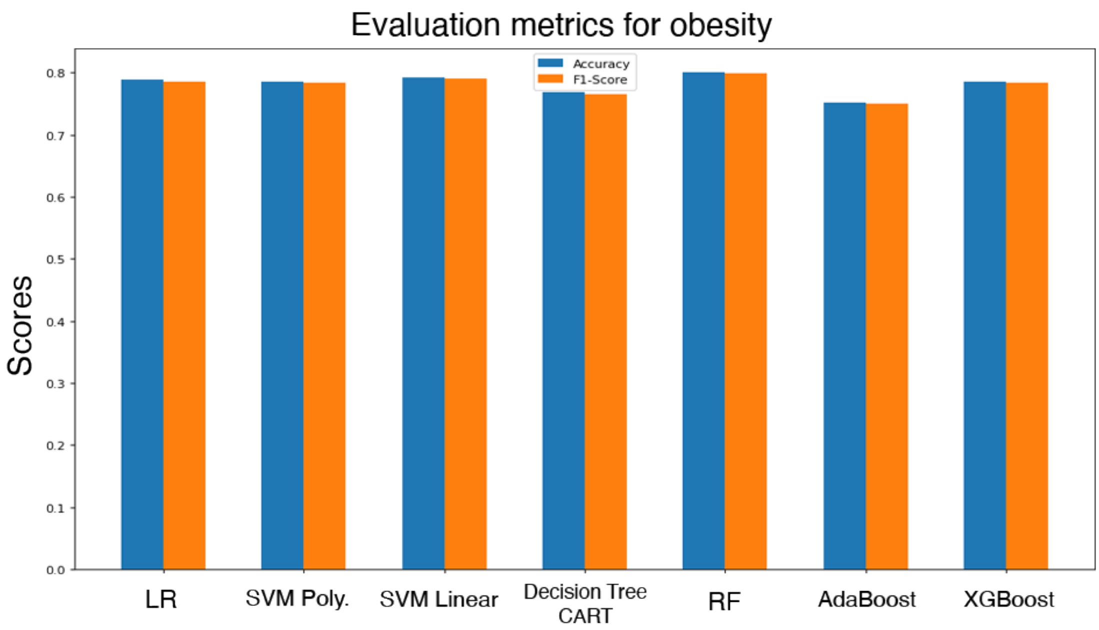


Figure 5-8: Obesity - Evaluation Performance using all attributes

Table 5-14 below provides an exact representation of the performance for each model. Despite that the RF model achieved the best results, LR, SVM (Poly. & Linear), and XGBoost also achieved comparable results. Hence, for a better analysis, we calculated the confidence interval (with 95% confidence) of the accuracy scores for all these models as follows: LR [0.783583, 0.792792], SVM Poly. [0.780825, 0.790309], SVM Linear [0.788138, 0.797389], RF [0.795353, 0.804402], XGBoost [0.779882, 0.789286]. The CI results show insignificant differences in the performance of all the algorithms with a maximum accuracy difference of 2.45% between RF and XGBoost.

Table 5-14: Obesity - Evaluation Performance

Algorithms	**All Attributes**		**Top 10**		**Top 5**	
	Accuracy	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
LR	0.788	0.786	0.775	0.773	0.79	0.788
SVM Poly.	0.786	0.784	-	-	-	-
SVM Linear	0.793	0.791	0.79	0.788	0.774	0.77
CART (DT)	0.768	0.765	0.767	0.764	0.768	0.765
RF	**0.8**	**0.799**	**0.803**	**0.802**	**0.78**	**0.779**
AdaBoost	0.752	0.75	0.739	0.737	0.738	0.736
XGBoost	0.785	0.784	0.772	0.771	0.767	0.765

Figure 5-9 shows an interesting comparison between the three different attribute sets that were used to create the RF model with a small improvement reached when using the top 10 features compared to using all the attributes for training the classifier. Such observation encourages the selection of the reduced model (with 10 features) over the other models, especially because it preserves the performance with reduction in model complexity.

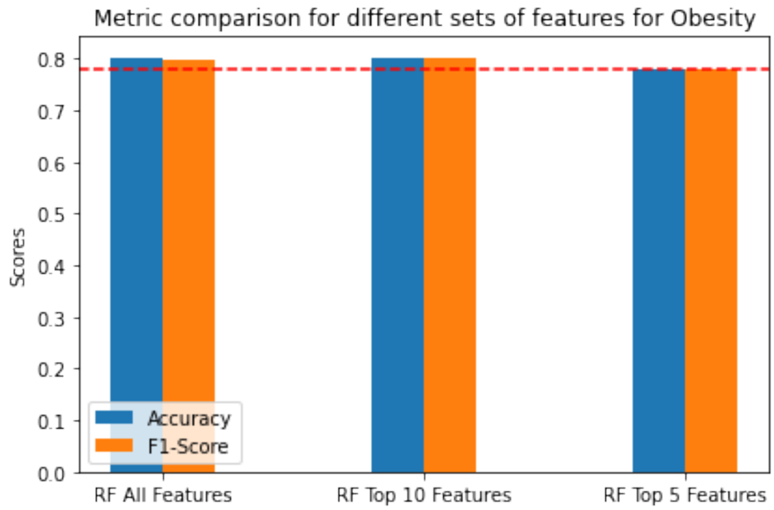


Figure 5-9: Obesity - Comparison between all feature sets

5.3.6 Neuropathy

The fifth complication in this study is neuropathy. The boosting methods achieved almost similar performance results. XGBoost reached an 82.9% accuracy as well as 81.8% for F1-score. SVM also achieved good results for this problem. Figure 5-10 shows the accuracy as well as F1-Score for all algorithms.

Table 5-15 provides an exact representation of the performance for each model. The confidence interval (with a confidence of 95%) was calculated for all the algorithms which achieved a performance over 80% as follows: SVM Linear [0.795669, 0.812908], RF [0.81047, 0.831427], AdaBoost [0.800702, 0.82167] and XGBoost [0.819454, 0.839083]. The calculated intervals confirm the insignificant differences in accuracy scores between the algorithms, especially for RF and XGBoost.

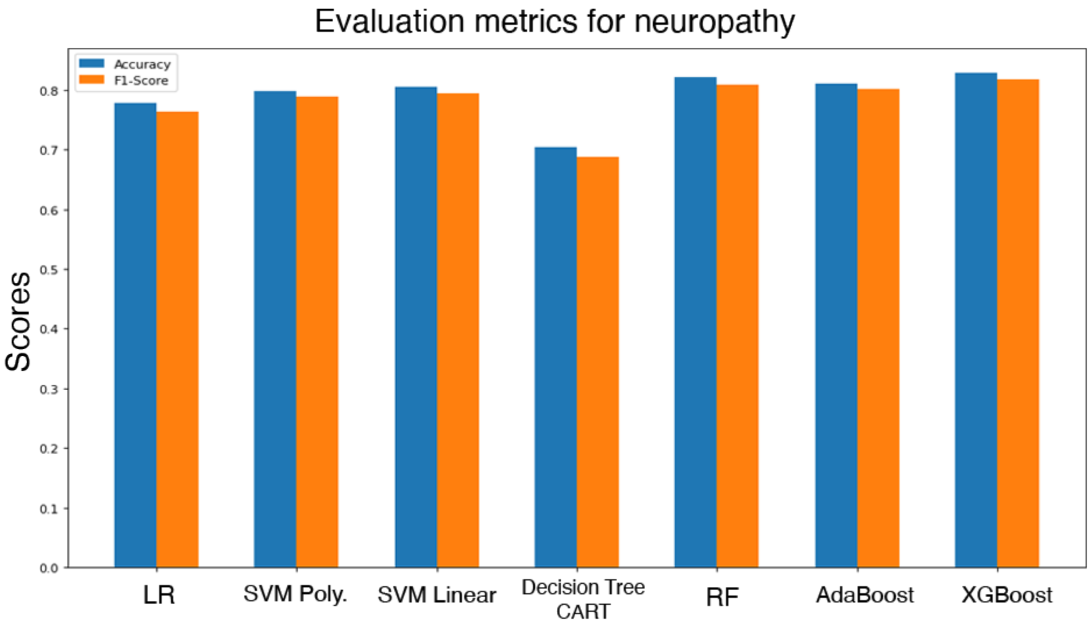


Figure 5-10: Neuropathy - Evaluation Performance using all attributes

Table 5-15: Neuropathy - Evaluation Performance

Algorithms	**All Attributes**		**Top 10**		**Top 5**	
	Accuracy	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
LR	0.778	0.764	0.757	0.744	0.708	0.688
SVM Poly.	0.798	0.789	-	-	-	-
SVM Linear	0.804	0.795	0.786	0.778	0.757	0.744
CART (DT)	0.704	0.688	0.712	0.697	0.68	0.661
RF	0.821	0.809	0.783	0.77	0.717	0.701
AdaBoost	0.811	0.802	0.779	0.769	0.708	0.693
XGBoost	**0.829**	**0.818**	**0.804**	**0.794**	**0.739**	**0.726**

As it is noticed in Figure 5-11, we were able to achieve comparable results when using the top 10 features for model training and evaluation. However, the model lost around 10% of its accuracy when using only the top 5 features. Such observation encourages the selection of the reduced model (with 10 features) over the other models, especially since it preserves the performance with reduction in model complexity.

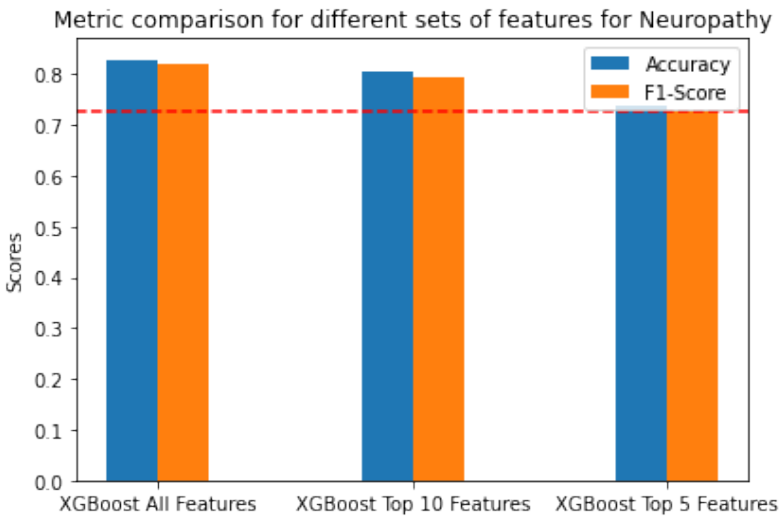


Figure 5-11: Neuropathy - Comparison between all feature sets

5.3.7 Nephropathy

For nephropathy, the heights evaluation scores were achieved by using AdaBoost. The best scores reached were nearly 92% and 91.4% for the accuracy and F1-scores, respectively. The second-best performance was achieved by using XGBoost. Figure 5-12 shows the accuracy as well as F1-Score for all algorithms.

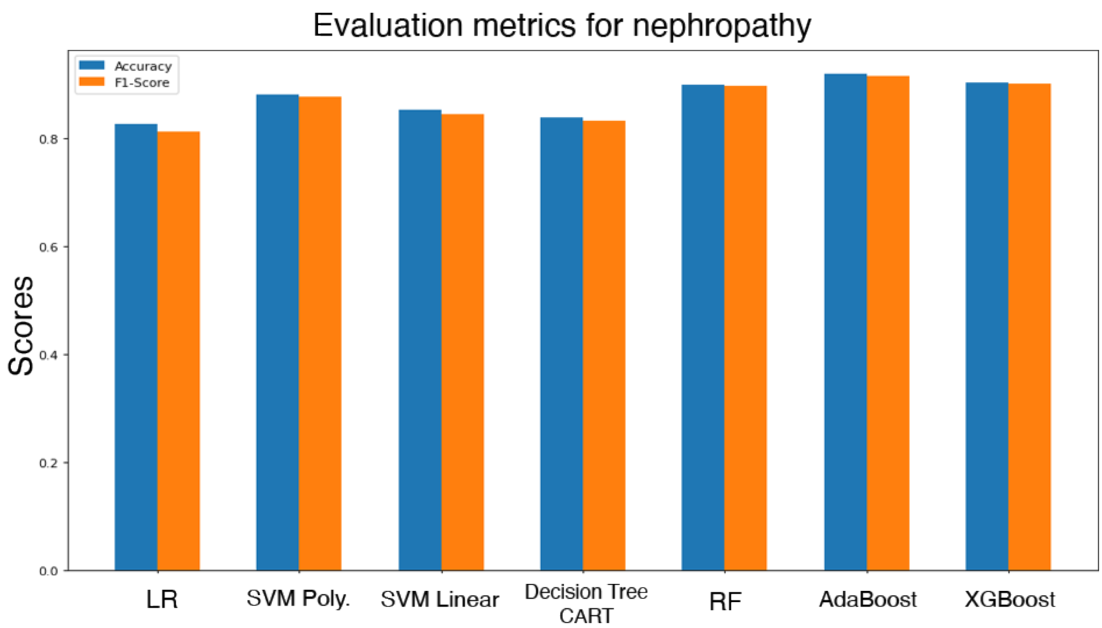


Figure 5-12: Nephropathy - Evaluation Performance using all attributes

Table 5-16 below provides an exact representation of the performance for each model. For a better analysis, the confidence interval (with a confidence of 95%) of the accuracy was calculated for the ensemble algorithms which achieved the top performance as follows: RF [0.890769, 0.905934], AdaBoost [0.911102, 0.9235] and XGBoost [0.894516, 0.909107]. As it is noticed from the CIs, both RF and XGBoost can achieve nearly similar results. Moreover, with a confidence of 95%, the differences between these models and AdaBoost can range between a minimum of 0.52% and a maximum of 3.27% which indicates the insignificance in the performance results between these models.

Table 5-16: Nephropathy - Evaluation Performance

Algorithms	**All Attributes**		**Top 10**		**Top 5**	
	Accuracy	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
LR	0.825	0.811	0.8	0.784	0.772	0.753
SVM Poly.	0.88	0.876	-	-	-	-
SVM Linear	0.852	0.844	0.819	0.805	0.822	0.807
CART (DT)	0.838	0.831	0.838	0.831	0.839	0.831
RF	0.898	0.896	0.892	0.889	0.891	0.886
AdaBoost	**0.917**	**0.914**	**0.887**	**0.883**	**0.85**	**0.844**
XGBoost	0.902	0.899	0.885	0.881	0.867	0.861

As shown in Figure 5-13, a minimum accuracy of 85% was reached by training the AdaBoost model with only the top 5 features.

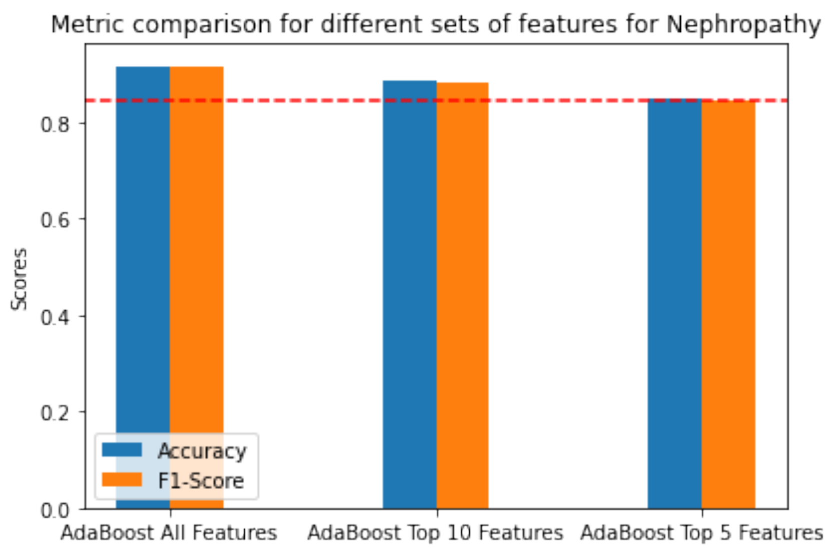


Figure 5-13: Nephropathy - Comparison between all feature sets

5.3.8 Diabetic foot

For diabetic foot, the best evaluation scores were achieved by using XGBoost, with 97.8% and 97.7% for accuracy and F1-scores, respectively. It is considered one of the highest scores achieved for a complication in this study indeed. Figure 5-14 compares the performance using all the algorithms.

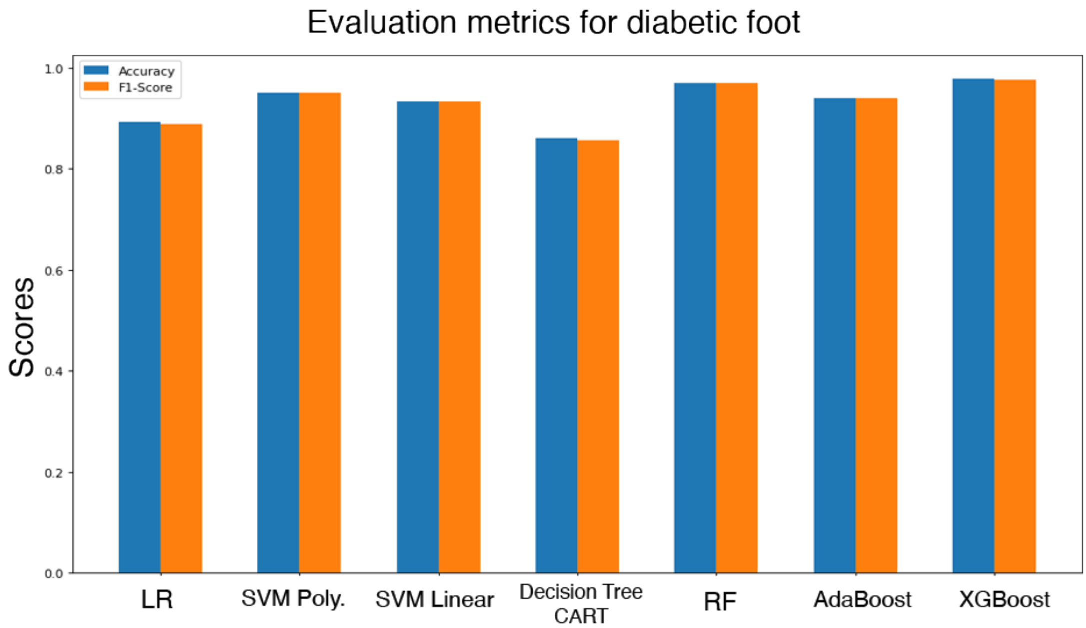


Figure 5-14: Diabetic Foot - Evaluation Performance using all attributes

Table 5-17 below provides the exact representation of the performance for each model. Moreover, the confidence interval (with a confidence of 95%) of the accuracy score was calculated for the top three algorithms which achieved a performance of more than 0.95 as follows: SVM Poly. [0.944898, 0.956768], RF [0.966089, 0.975578], and XGBoost [0.973892, 0.981108]. As it is observed from the previous intervals, both RF and XGBoost can achieve a comparable performance, where the minimum difference between both algorithms is 0.168% and the maximum is 1.5%.

Table 5-17: Diabetic Foot - Evaluation Performance

Algorithms	All Attributes		Top 10		Top 5	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
LR	0.893	0.888	0.868	0.862	0.843	0.837
SVM Poly.	0.951	0.95	-	-	-	-

SVM Linear	0.935	0.933	0.908	0.904	0.907	0.903
CART (DT)	0.86	0.856	0.865	0.86	0.86	0.855
RF	0.971	0.97	0.944	0.942	0.92	0.916
AdaBoost	0.941	0.939	0.935	0.933	0.918	0.915
XGBoost	**0.978**	**0.977**	**0.938**	**0.936**	**0.91**	**0.907**

As shown in Figure 5-15, there is a trade-off between using a complicated model to reach the best performance possible and the use of a much simplified model with a degradation in performance. For the XGBoost algorithm, the accuracy difference between using all the attributes and the top 10 features is 4%.

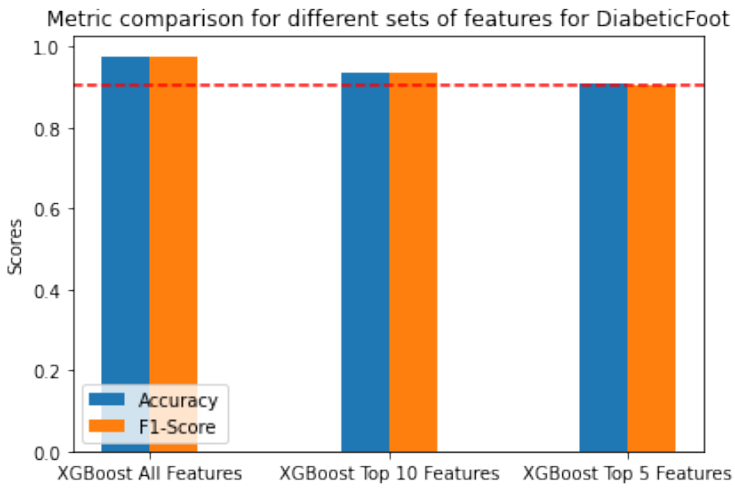


Figure 5-15: Diabetic Foot - Comparison between all feature sets

5.3.9 Retinopathy

The last complication to study in this research is retinopathy. In comparison with nephropathy, retinopathy achieved slightly the same results for the accuracy as well as F1-Socres with an 87.2% and 86.7%, respectively. Figure 5-16, shows the accuracy as well as F1-Scores for all algorithms.

Table 5-18 below provides the exact representation of the performance for each model. The confidence interval (with a confidence of 95%) of the accuracy was calculated for the best performing algorithms as follows: SVM Poly. [0.839055, 0.857664], RF [0.837813, 0.858274], AdaBoost [0.843004, 0.861147] and XGBoost [0.863146, 0.881044]. The CIs indicates that all the algorithms can indeed achieve comparable performance, hence, the performance differences between these algorithms can be considered insignificant in this case.

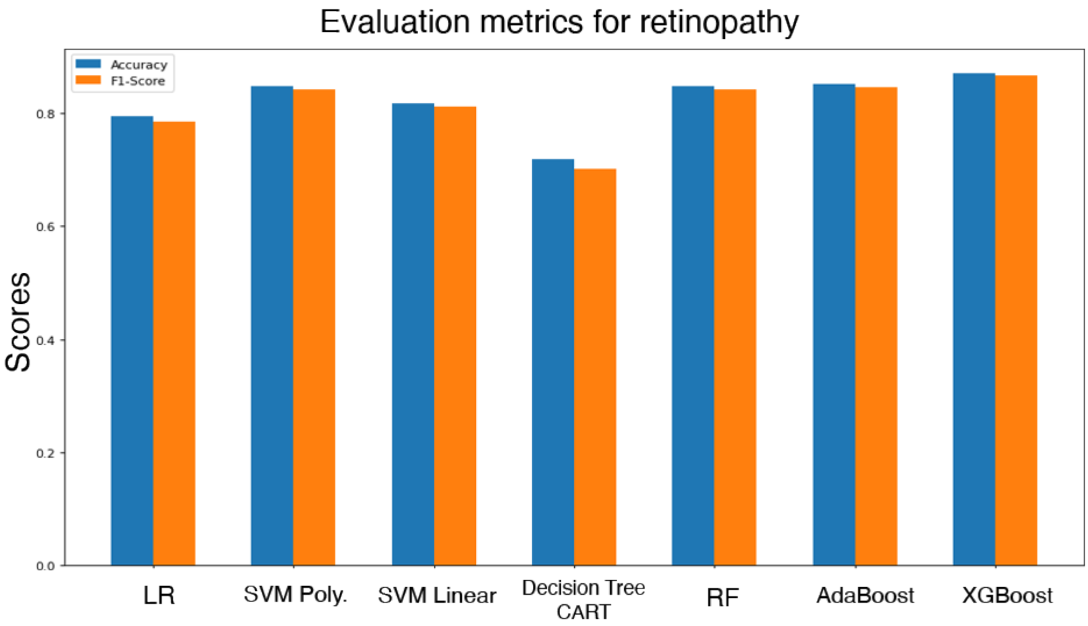


Figure 5-16: Retinopathy - Evaluation Performance using all attributes

Table 5-18: Retinopathy - Evaluation Performance

Algorithms	All Attributes		Top 10		Top 5	
	Accuracy	**F1-Score**	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
LR	0.796	0.786	0.792	0.779	0.784	0.771
SVM Poly.	0.848	0.843	-	-	-	-
SVM Linear	0.818	0.812	0.801	0.789	0.792	0.774
CART (DT)	0.719	0.703	0.722	0.705	0.731	0.717
RF	0.848	0.842	0.832	0.825	0.801	0.793
AdaBoost	0.852	0.846	0.828	0.821	0.759	0.748
XGBoost	**0.872**	**0.867**	**0.804**	**0.796**	**0.771**	**0.762**

Figure 5-17 shows a comparison between using different feature sets to build the XGBoost model. A maximum difference of almost 10% in accuracy is present when using the full dataset attributes and top 5 features. However, the difference between using the full dataset attributes and the top 10 features is less significant with only 2.4% in favour of the full set.

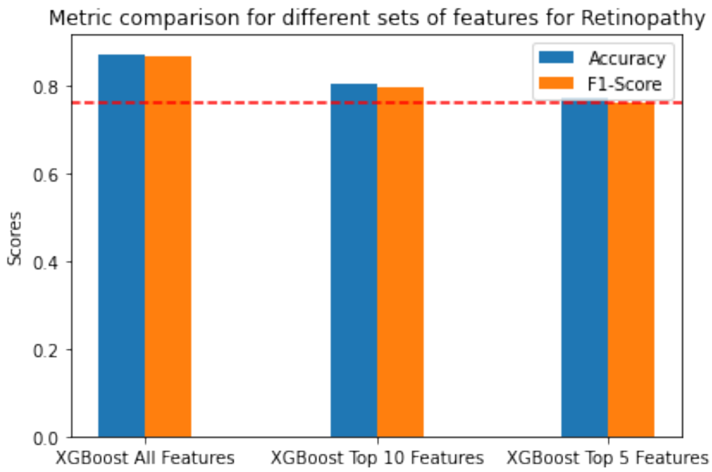


Figure 5-17: Retinopathy - Comparison between all feature sets

5.3.10 Best features

As shown in the previous section, besides utilizing the full attribute set for building the models, a feature selection was performed on the dataset to select the top 5 and 10 features. Moreover, these selected features were utilized to build specialized models.

This step has many beneficial uses. For instance, the first benefit is to enhance our understanding of the data by understanding which attributes contributed to the results the most. This step is especially essential since we are dealing with more than 60 attributes. The second benefit of selecting the features is to compare the knowledge extracted from using ARM models and compare it with the knowledge constructed while using supervised ML. For that, a detailed discussion can be found in section 5.4.

As discussed before, since this research studies 8 different complications independently, we utilized the top 8 models, one for each complication, to select the top 10 features. Table 5-19 describes the selected features.

Table 5-19: Top 10 features selected from the best supervised models

Complication	Best ML Algorithm	Top 10 Features
Metabolic Syndrome	LR	'Gender_Female', 'Gender_Male', 'DiabetesType_Type 2 Adult Onset', 'WeightKg', 'BMI', 'BPDiastolic', 'TestLDL', 'TestUrine creatinine', 'TestHCT', 'TestMicroalbumin creatinine ratio'
Dyslipidemia	RF	'TestLDL', 'TestTotal cholesterol', 'Age', 'BMI', 'WeightKg', 'TestC- peptide', 'BPSystolic', 'TestTriglycerides', 'TestHDL', 'TestNeutrophils'

Hypertension	RF	TestCystatin C', 'TestLDL', 'Age', 'BPSystolic', 'BPDiastolic', 'TestTotal cholesterol', 'BMI', 'Test25-OH Vitamin D Total', 'DiabetesAge', 'TestBICARB'
Obesity	RF	'WeightKg', 'BMI', 'TestInsulin', 'TestK', 'HeightMetres', 'TestCRP-HS', 'TestMXD', 'TestC-peptide', 'TestRBC', 'TestBilirubin DIR'
Neuropathy	XGBoost	Gender_Male', 'TestMicroalbumin', 'DiabetesAge', 'BPSystolic', 'TestTotal cholesterol','TestMicroalbumin creatinine ratio', 'TestAlkaline Phosphatase', 'DiabetesType_Type 2 Adult Onset', 'TestAST', 'TestHDL'
Nephropathy	AdaBoost	'Gender_Male', 'TestMicroalbumin', 'DiabetesAge', 'BPSystolic', 'TestTotal cholesterol', 'TestMicroalbumin creatinine ratio', 'TestAlkaline Phosphatase', 'DiabetesType_Type 2 Adult Onset', 'TestAST', 'TestHDL'
Diabetic Foot	XGBoost	'TestLymphocyte', 'HeightMetres', 'DiabetesAge', 'Gender_Male', 'TestTotal cholesterol', 'TestNa', 'TestNeutrophils', 'TestBICARB', 'TestBilirubin Total', 'Test25-OH Vitamin D Total'
Retinopathy	XGBoost	'DiabetesAge', 'TestUrea', 'TestAST', 'NationalityName_United Arab Emirate', 'TestLympacyte Absolute', 'TestC- peptide', 'TestCreatinin', 'TestHb', 'TestALT', 'TestCRP-HS'

For easier analysis of all the features mentioned in Table 5-19, Table 5-20 provides further analysis showing the frequency of occurrence of the features above for all the complications.

Table 5-20: The total number of occurrences of the top-10 features for each complication

Number of Occurrence	Features
5	'TestTotal cholesterol', 'Diabetes Age'
4	'Gender_Male', 'BMI', 'BPSystolic'
3	'DiabetesType_Type 2 Adult Onset', 'WeightKg', 'TestLDL', 'TestMicroalbumin creatinine ratio', 'TestC- peptide', 'TestHDL', 'TestAST'
2	'BPDiastolic', 'Age', 'Test25-OH Vitamin D Total', 'TestBICARB', 'HeightMetres', 'TestCRP-HS', 'TestMicroalbumin', 'TestAlkaline Phosphatase'
1	'Gender_Female', 'TestUrine creatinine', 'TestHCT', 'TestTriglycerides', 'TestNeutrophils, TestCystatin C', 'TestInsulin', 'TestK', 'TestMXD', 'TestRBC', 'TestBilirubin DIR', 'TestLymphocyte', 'TestNa', 'TestNeutrophils', 'TestBilirubin Total', 'TestUrea',

	'NationalityName_United Arab Emirate', 'TestLympacyte Absolute', 'TestCreatinin', 'TestHb', 'TestALT'

As it is observed from the previous table, there are some expected results that were found to be useful for classification such as Diabetes Age and Gender. However, the experiments also show some other useful features. For instance, BMI, Albumin, Creatinine and other medical tests were all observed to have a main impact on the models' performance.

5.4. Discussion of Results

As shown in the previous section, many algorithms were utilized to study eight different diabetes complications. By comparing the results found by the baseline classifiers in Table 5-10 with the best results achieved for complications' models, it is observed that the final trained models overperformed the simple classifiers. Moreover, by comparing our results with the reported accuracy scores in [26], we can notice that our models achieved more than 10% improvement for predicting retinopathy, nephropathy as well as neuropathy. 21 shows further comparisons between our proposed method and other available studies. The accuracy score was used for the comparison since it is the mostly utilized evaluation metric in the literature. According to the table, our model achieved remarkable results for predicting diabetic foot. Furthermore, and based on the previously calculated confidence interval for using XGBoost to predict diabetic foot which is [0.973892, 0.981108], it is noticed that with a confidence of 95%, the minimum accuracy score can be reached to predict the complication is 97.38% which is higher than other models in 21.

Table 5-21: A comparison of recent works developed for predicting diabetes complications using ML.

Source	Dataset Size	Best Model	Complication	Accuracy
Our Study	884	XGBoost	Diabetic Foot	97.8%
[15]	455	ID3	Eye, Kidney Heart and Diabetic Hyperlipidemia	92.35%
[26]	943	LR	retinopathy	77.7%
[13]	779	RF	nephropathy	89%

From the reported results in section 5.3, it can be observed that LR, RF, AdaBoost and XGBoost mainly achieved the best performance. We believe this relates to the problem in hand. For instance, reaching a good performance by using LR reinforces the importance of testing simple algorithms first. The simplicity plays a significant role in understanding the problem and the results as well as reducing the training time needed to build a classifier. We also observed that utilizing tree-based ensemble algorithms is essential in such problems. For instance, XGBoost, RF and AdaBoost rely on building several sub-trees to come up with the final output. Another observation that can be extracted from the reported results is that in most cases the best results whether for all attributes or top 10 or top 5 sets are produced by the same algorithm which confirms the importance of that algorithm for predicting a specific complication. Although different algorithms resulted in different performance, we observe that by using RF alone, we can predict all the complications and still achieve acceptable results.

It is also important to study and compare the performance reached for each complication. For instance, one observation is related to the distribution of the output class. The distribution itself plays a significant role and can affect the overall performance. This can be noticed since we used the same independent features to predict all the complications, in fact, the only thing changed is the output class (the complications in this case).

By looking at the performance of the best models, we observed that by using only a small subset of all the attributes available we can still achieve acceptable results. The performance achieved by using the selected features' sets and the total number of features was compared by calculating the mean and standard deviation of the difference between the accuracy scores. For example, the difference between the accuracy score achieved by using all the attributes and by using only the top 10 attributes is 0.0332 ± 0.021, whereas using the top 5 attributes resulted in a difference of 0.06 ± 0.032. From such results we observe that the degradation of performance resulted by using selected features in most cases is very limited. This observation emphasizes the positive effects of applying features selection on the dataset. Furthermore, reducing the number of attributes by more than 60 features has positive effects of reducing the training and prediction time needed.

Moreover, the top 10 selected features were compared with attributes found in the best ARM rules. Since we have 8 complications, we end up with 8 different sets of features. As shown previously in Table 5-20, the features ordered based on the number of occurrences in all the 8 sets, including Test Total cholesterol, Diabetes Age, Gender Male, BMI, BPSystolicas, Diabetes Type 2 Adult Onset, Insulin, Nationality, 25-OH Vitamin D Total and others. For ARM, the attributes found in the best rules contains: EAG, Nationality, Age, HbA1c (mmol), Diabetes Type 2 Adult Onset, 25-OH Vitamin D Total and Insulin. By comparing the reported features from both experiments (supervised ML algorithms and ARM), we can indeed observe that there are some strong relations between the features resulting from both parts. For instance, Insulin as well as 25-OH Vitamin D Total were found in both experiments. Nationality and diabetes type 2 and some other variables also exist in both sets. This observation can help us building more sophisticated models by giving more attention and weight to such features. Moreover, physicians can also benefit from such information by also investigating possible relations between these features in the lab.

Chapter 6. Conclusion and Future Work

In this research, data mining and machine learning algorithms were used to diagnose eight different diabetes complications. The complications' set consists of metabolic syndrome, dyslipidemia, hypertension, obesity, diabetic foot, neuropathy, nephropathy and retinopathy. All these complications are available in a dataset provided by the Rashid Centre for Diabetes and Research (RCDR). The dataset consists of 884 records and 79 attributes. After cleaning the dataset, multiple experiments were conducted to solve the missing value problem. For that, simple mean imputation, *k*-NN as well as MissForest were all tested and evaluated. It was found that MissForest achieved the minimum RMSE score. As a result, it was utilized throughout the rest of this research. After handling all the missing values, one-hot encoding was applied on the categorical attributes such as Nationality Name, Gender and Diabetes Type.

Since this study aims to investigate any possible relations between the attributes in the dataset, Association Rule Mining was utilized. After the general preprocessing steps were applied, a simple data discretization method was used to transform all numerical features into categorical ones. Apriori algorithm and FP-Growth were both applied on the prepared dataset. It was noticed that both algorithms resulted in the same set of rules. However, FP-Growth required less computational resources, hence it was utilized to extract rules using the total number of attributes.

The second part of this thesis focused on building several supervised ML models to predict and classify all the diabetes complications available. Since the dataset in hand suffers from data imbalance issue, different balancing methods were examined. A combination of SMOTE for oversampling the minority class and cluster centroids for under sampling the majority class was used. The algorithms constructed for this study contains Logistic Regression, SVM, Decision Tree (CART), Random Forest, AdaBoost and XGBoost. Extensive experiments were carried out for model tuning and training. Grid search with cross validation was employed to select the best hyperparameters for each model. Moreover, *k*-fold cross (KCV) validation with k=10 was utilized to split the data into training and testing sets. Since the data has imbalanced classes, stratified cross validation was applied. Moreover, to assure getting reliable results, the process of CV was repeated 10 times.

Along with using all the attributes to build the models, feature selection was applied on the dataset to select the top 10 and 5 features. Moreover, the models built using the reduced datasets achieved a comparable performance with the models utilized all the attribute set.

The feature selection step also helps in enhancing the understanding of the top features affecting the classifiers. A comparison between the selected features and the features appeared in the top ARM rules was conducted. We observed that there are indeed some interesting relations between both feature sets. This observation can help us build more sophisticated models by giving more attention and weight to such features.

In the future, we plan to extend this research to cover other multi-modal approaches trying to enhance the overall performance. Moreover, we are planning to utilize some unsupervised methods such as clustering, which can also help reinforcing our observations in this study and may lead to new hypotheses.

References

[1] CDC, “About Chronic Diseases,” Apr. 28, 2021. https://www.cdc.gov/chronicdisease/about/index.htm (accessed May 05, 2021).

[2] World Health Organization, “Diabetes,” Nov. 10, 2020. https://www.who.int/news-room/fact-sheets/detail/diabetes (accessed Nov. 30, 2020).

[3] MedlinePlus, “How to Prevent Diabetes,” Jun. 15, 2020. https://medlineplus.gov/howtopreventdiabetes.html (accessed Nov. 30, 2020).

[4] Centers for Disease Control and Prevention, “Diabetes,” 2019. https://www.cdc.gov/diabetes/managing/problems.html (accessed Nov. 30, 2020).

[5] M. H. Tekieh and B. Raahemi, “Importance of Data Mining in Healthcare,” in *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015 - ASONAM ’15*, Aug. 2015, pp. 1057–1062. doi: 10.1145/2808797.2809367.

[6] IBM, “What is Machine Learning?” https://www.ibm.com/ae-en/cloud/learn/machine-learning (accessed Sep. 26, 2021).

[7] R. Sharma, S. N. Singh, and S. Khatri, “Medical Data Mining Using Different Classification and Clustering Techniques: A Critical Survey,” in *2016 Second International Conference on Computational Intelligence & Communication Technology (CICT)*, Feb. 2016, pp. 687–691. doi: 10.1109/CICT.2016.142.

[8] V. Dominic, D. Gupta, S. Khare, and A. Aggarwal, “Investigation of chronic disease correlation using data mining techniques,” in *2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS)*, Dec. 2015, pp. 1–6. doi: 10.1109/RAECS.2015.7453329.

[9] A. Subasi, “Machine learning techniques,” in *Practical Machine Learning for Data Analysis Using Python*, A. Subasi, Ed. Academic Press, 2020, pp. 91–202. doi: 10.1016/B978-0-12-821379-7.00003-5.

[10] V. Kotu and B. Deshpande, "Association Analysis," in *Data Science*, 2nd ed., Morgan Kaufmann, 2019, pp. 199–220. doi: 10.1016/B978-0-12-814761-0.00006-X.

[11] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," in *Proceedings of the 20th International Conference on Very Large Data Bases*, 1994, pp. 487–499.

[12] R. Veroneze *et al.*, "Using association rule mining to jointly detect clinical features and differentially expressed genes related to chronic inflammatory diseases," *PLOS ONE*, vol. 15, no. 10, Oct. 2020, doi: 10.1371/journal.pone.0240269.

[13] M. Ashraful Alam Assistant Professor and A. Zaida Khanom, "Prediction of Diabetes Induced Complications Using Different Machine Learning Algorithms," BRAC University, 2018. Accessed: Nov. 30, 2020. [Online]. Available: http://dspace.bracu.ac.bd/xmlui/handle/10361/10945

[14] Scikit-Learn, "Support Vector Machines — scikit-learn 0.24.1 documentation." https://scikit-learn.org/stable/modules/svm.html (accessed Dec. 05, 2020).

[15] K. Kantawong, S. Tongphet, P. Bhrommalee, N. Rachata, and S. Pravesjit, "The Methodology for Diabetes Complications Prediction Model," in *2020 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)*, Mar. 2020, pp. 110–113. doi: 10.1109/ECTIDAMTNCON48261.2020.9090700.

[16] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, Mar. 1986.

[17] Scikit-Learn, "Ensemble methods — scikit-learn 0.24.1 documentation." https://scikit-learn.org/stable/modules/ensemble.html (accessed Mar. 27, 2021).

[18] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: https://doi.org/10.1023/A:1010933404324.

[19] Scikit-Learn, “Decision Trees — scikit-learn 1.0 documentation.” https://scikit-learn.org/stable/modules/tree.html (accessed Sep. 30, 2021).

[20] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[21] Md. K. Hasan, Md. A. Alam, D. Das, E. Hossain, and M. Hasan, “Diabetes Prediction Using Ensembling of Different Machine Learning Classifiers,” *IEEE Access*, vol. 8, pp. 76516–76531, 2020, doi: 10.1109/ACCESS.2020.2989857.

[22] D. Sisodia and D. S. Sisodia, “Prediction of Diabetes using Classification Algorithms,” *Procedia Computer Science*, vol. 132, pp. 1578–1585, 2018, doi: 10.1016/j.procs.2018.05.122.

[23] X.-H. Meng, Y.-X. Huang, D.-P. Rao, Q. Zhang, and Q. Liu, “Comparison of three data mining models for predicting diabetes or prediabetes by risk factors,” *The Kaohsiung Journal of Medical Sciences*, vol. 29, no. 2, pp. 93–99, Feb. 2013, doi: 10.1016/j.kjms.2012.08.016.

[24] E. W. Gregg, N. Sattar, and M. K. Ali, “The changing face of diabetes complications,” *The Lancet Diabetes and Endocrinology*, vol. 4, no. 6. Lancet Publishing Group, pp. 537–547, Jun. 01, 2016. doi: 10.1016/S2213-8587(16)30010-9.

[25] J. R. Quinlan, “Induction of decision trees,” *Machine Learning*, vol. 1, no. 1, Mar. 1986, doi: 10.1007/BF00116251.

[26] A. Dagliati *et al.*, “Machine Learning Methods to Predict Diabetes Complications,” *Journal of Diabetes Science and Technology*, vol. 12, no. 2, pp. 295–302, Mar. 2018, doi: 10.1177/1932296817706375.

[27] D. J. Stekhoven and P. Buhlmann, “MissForest--non-parametric missing value imputation for mixed-type data,” *Bioinformatics*, vol. 28, no. 1, pp. 112–118, Jan. 2012, doi: 10.1093/bioinformatics/btr597.

[28] M. Cui *et al.*, “Risk Assessment of Sarcopenia in Patients With Type 2 Diabetes Mellitus Using Data Mining Methods,” *Frontiers in Endocrinology*, vol. 11, p. 123, Mar. 2020, doi: 10.3389/fendo.2020.00123.

[29] I. Kavakiotis, O. Tsave, A. Salifoglou, N. Maglaveras, I. Vlahavas, and I. Chouvarda, “Machine Learning and Data Mining Methods in Diabetes Research,” *Computational and Structural Biotechnology Journal*, vol. 15, pp. 104–116, 2017, doi: 10.1016/j.csbj.2016.12.005.

[30] K. S. Lakshmi and G. Vadivu, “Extracting Association Rules from Medical Health Records Using Multi-Criteria Decision Analysis,” *Procedia Computer Science*, vol. 115, pp. 290–295, 2017, doi: 10.1016/j.procs.2017.09.137.

[31] M. D. Kamalesh, K. H. Prasanna, B. Bharathi, R. Dhanalakshmi, and R. Aroul Canessane, “Predicting the Risk of Diabetes Mellitus to Subpopulations Using Association Rule Mining,” in *Proceedings of the International Conference on Soft Computing Systems*, vol. 397, L. P. and Suresh and B. K. Panigrahi, Eds. Springer, New Delhi, 2016, pp. 59–65. doi: 10.1007/978-81-322-2671-0_6.

[32] P. Kasemthaweesab and W. Kurutach, “Association analysis of diabetes mellitus (DM) with complication states based on association rules,” in *2012 7th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, Jul. 2012, pp. 1453–1457. doi: 10.1109/ICIEA.2012.6360952.

[33] I. H. Witten and E. Frank, “Data mining: practical machine learning tools and techniques with Java implementations,” *Acm Sigmod Record*, vol. 31, pp. 76–77, 2002.

[34] SKMCA, “Rashid Centre for Diabetes & Research.” https://www.skmca.ae/rashid-centre-for-diabetes-research/ (accessed Jan. 21, 2021).

[35] Centers for Disease Control and Prevention, “About Adult BMI | Healthy Weight, Nutrition, and Physical Activity.” https://www.cdc.gov/healthyweight/assessing/bmi/adult_bmi/index.html (accessed Jan. 21, 2021).

[36] WebMed, “HbA1c (Hemoglobin A1c): A1c Chart, Test, Levels, & Normal Range.” https://www.webmd.com/diabetes/guide/glycated-hemoglobin-test-hba1c (accessed Jan. 21, 2021).

[37] J. R. Chalcraft *et al.*, “Vitamin D : National Institutes of Health (NIH),” *Nutrients*, vol. 12, no. 8, pp. 1–15, Aug. 2020, doi: 10.3390/nu12082237.

[38] WebMed, “Blood Pressure Chart & Numbers (Normal Range, Systolic, Diastolic).” https://www.webmd.com/hypertension-high-blood-pressure/guide/diastolic-and-systolic-blood-pressure-know-your-numbers#1 (accessed Jan. 21, 2021).

[39] World Health Organization, “Hypertension.” https://www.who.int/news-room/fact-sheets/detail/hypertension (accessed Jan. 21, 2021).

[40] World Health Organization, “Obesity and overweight.” https://www.who.int/news-room/fact-sheets/detail/obesity-and-overweight (accessed Jan. 21, 2021).

[41] ScienceDirect, “Dyslipidemia - an overview.” https://www.sciencedirect.com/topics/neuroscience/dyslipidemia (accessed Jan. 21, 2021).

[42] American Heart Association, “About Metabolic Syndrome.” https://www.heart.org/en/health-topics/metabolic-syndrome/about-metabolic-syndrome (accessed Jan. 21, 2021).

[43] C. D. Liapis, K. Balzer, F. Benedetti-Valentini, and J. Fernandes e Fernandes, Eds., “Diabetic Foot,” in *Vascular Surgery*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 501–521. doi: 10.1007/978-3-540-30956-7_45.

[44] Centers for Disease Control and Prevention, “Diabetes and Nerve Damage | Features | Library.” https://www.cdc.gov/diabetes/library/features/diabetes-nerve-damage.html (accessed Jan. 21, 2021).

[45] WebMed, “What is nephropathy?” https://www.webmd.com/diabetes/qa/what-is-nephropathy (accessed Jan. 21, 2021).

[46] Centers for Disease Control and Prevention, "Diabetic Retinopathy | Examination-Based Studies | Information on Data Sources | Vision and Eye Health Surveillance System | Vision Health Initiative (VHI)." https://www.cdc.gov/visionhealth/vehss/data/studies/diabetic-retinopathy.html (accessed Jan. 21, 2021).

[47] Scikit-Learn, "Imputation of missing values — scikit-learn 1.0 documentation." https://scikit-learn.org/stable/modules/impute.html#knnimpute (accessed Oct. 04, 2021).

[48] Centers for Disease Control and Prevention, "Defining Adult Overweight and Obesity | Overweight & Obesity." https://www.cdc.gov/obesity/adult/defining.html (accessed Mar. 06, 2021).

[49] S. M. Shamshirgaran, A. Mamaghanian, A. Aliasgarzadeh, N. Aiminisani, M. Iranparvar-Alamdari, and J. Ataie, "Age differences in diabetes-related complications and glycemic control," *BMC Endocrine Disorders*, vol. 17, no. 1, pp. 1–7, Dec. 2017, doi: 10.1186/s12902-017-0175-5.

[50] The imbalanced-learn developers, "ClusterCentroids — Version 0.8.1." https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.ClusterCentroids.html (accessed Oct. 10, 2021).

[51] N. v. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.

[52] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, pp. 40–79, Jan. 2010, doi: 10.1214/09-SS054.

[53] D. Krstajic, L. J. Buturovic, D. E. Leahy, and S. Thomas, "Cross-validation pitfalls when selecting and assessing regression and classification models," *Journal of Cheminformatics*, vol. 6, no. 1, pp. 1–15, Dec. 2014, doi: 10.1186/1758-2946-6-10.

[54] M. Ojala and G. C. Garriga, “Permutation Tests for Studying Classifier Performance,” in *2009 Ninth IEEE International Conference on Data Mining*, Dec. 2009, pp. 908–913. doi: 10.1109/ICDM.2009.108.

[55] Scikit-Learn, “GridSearchCV — scikit-learn 1.0 documentation.” https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (accessed Oct. 11, 2021).

[56] Scikit-Learn, “Feature importances — scikit-learn 1.0 documentation.” https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html (accessed Oct. 11, 2021).

[57] M. Hahsler, B. Grün, and K. Hornik, “A Computational Environment for Mining Association Rules and Frequent Item Sets,” *Journal of Statistical Software*, vol. 14, no. 15, pp. 1–25, 2005, doi: 10.18637/jss.v014.i15.

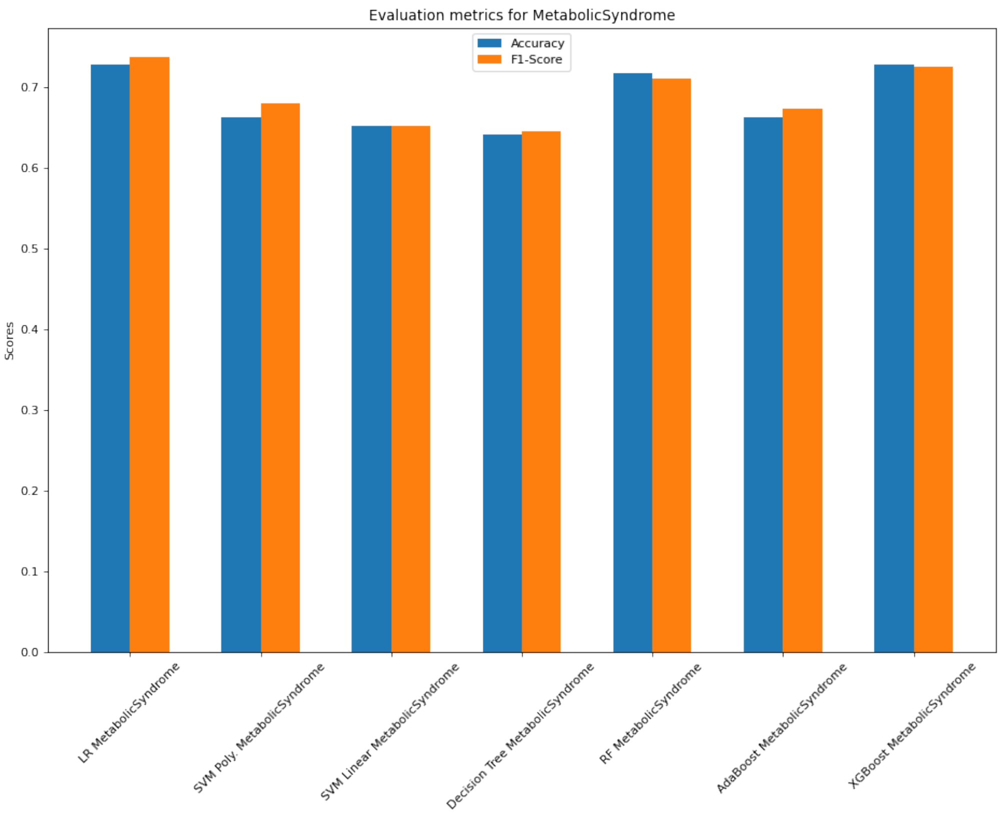
[58] S. Minaee, “20 Popular Machine Learning Metrics. Part 1: Classification & Regression Evaluation Metrics | by Shervin Minaee | Towards Data Science,” Oct. 28, 2019. https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce (accessed Mar. 25, 2021).

[59] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, Third Edition. Elsevier, 2011. doi: 10.1016/C2009-0-19715-5.

Appendix A

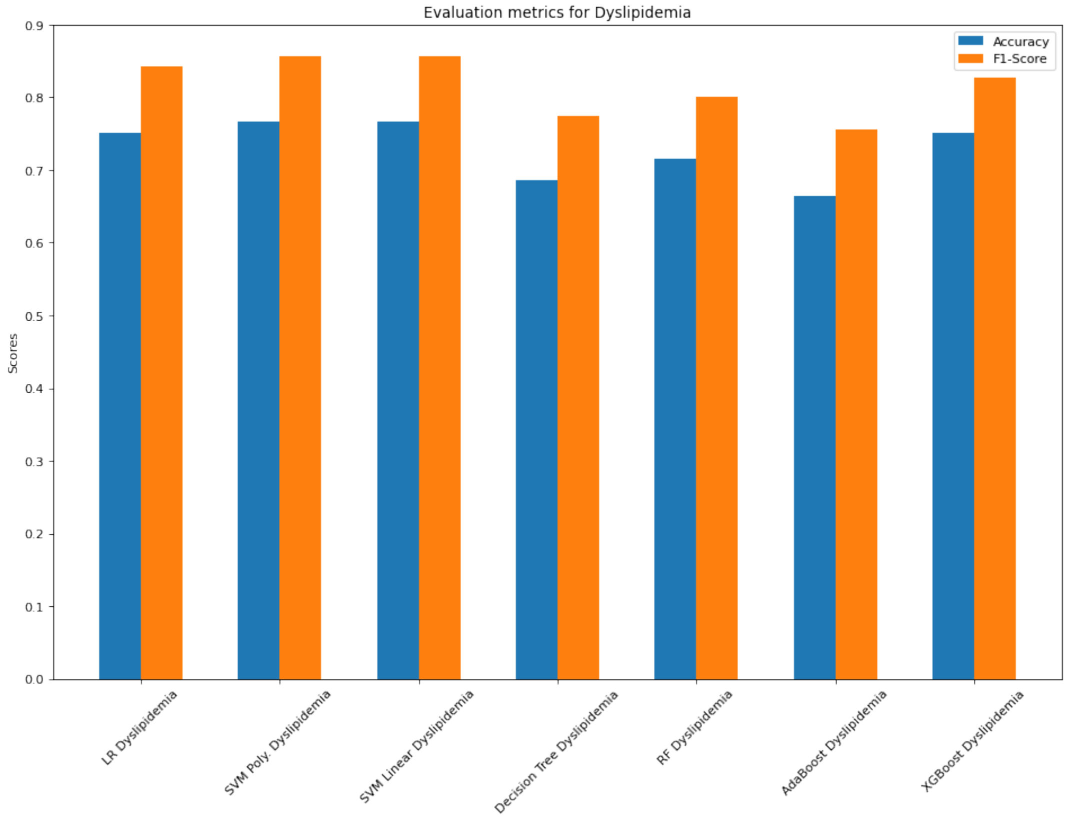
Model evaluation using Train Test Splits

Metabolic Syndrome - Evaluation Performance using all attributes



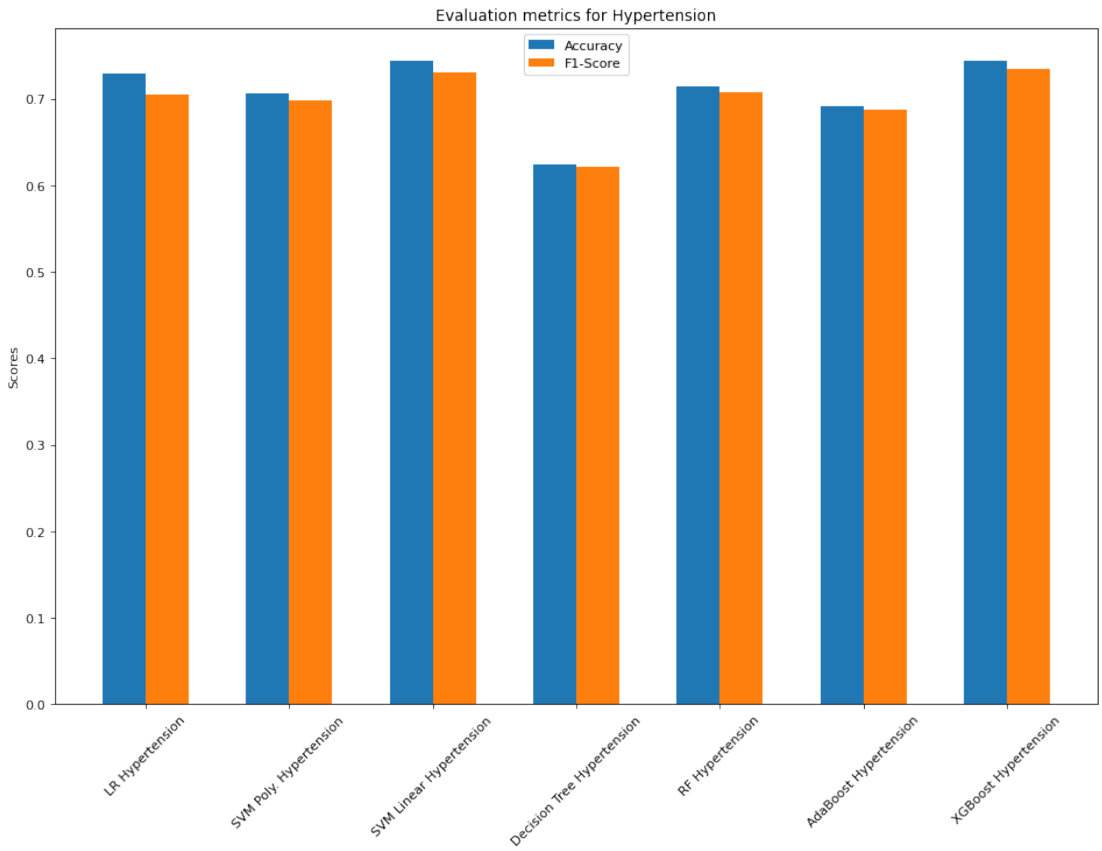
Algorithm	Accuracy	F1-Score
LR	0.728261	0.736842
SVM Poly.	0.663043	0.680412
SVM Linear	0.652174	0.652174
CART (DT)	0.641304	0.645161
RF	0.717391	0.711111
AdaBoost	0.663043	0.673684
XGBoost	0.728261	0.725275

Dyslipidemia - Evaluation Performance using all attributes



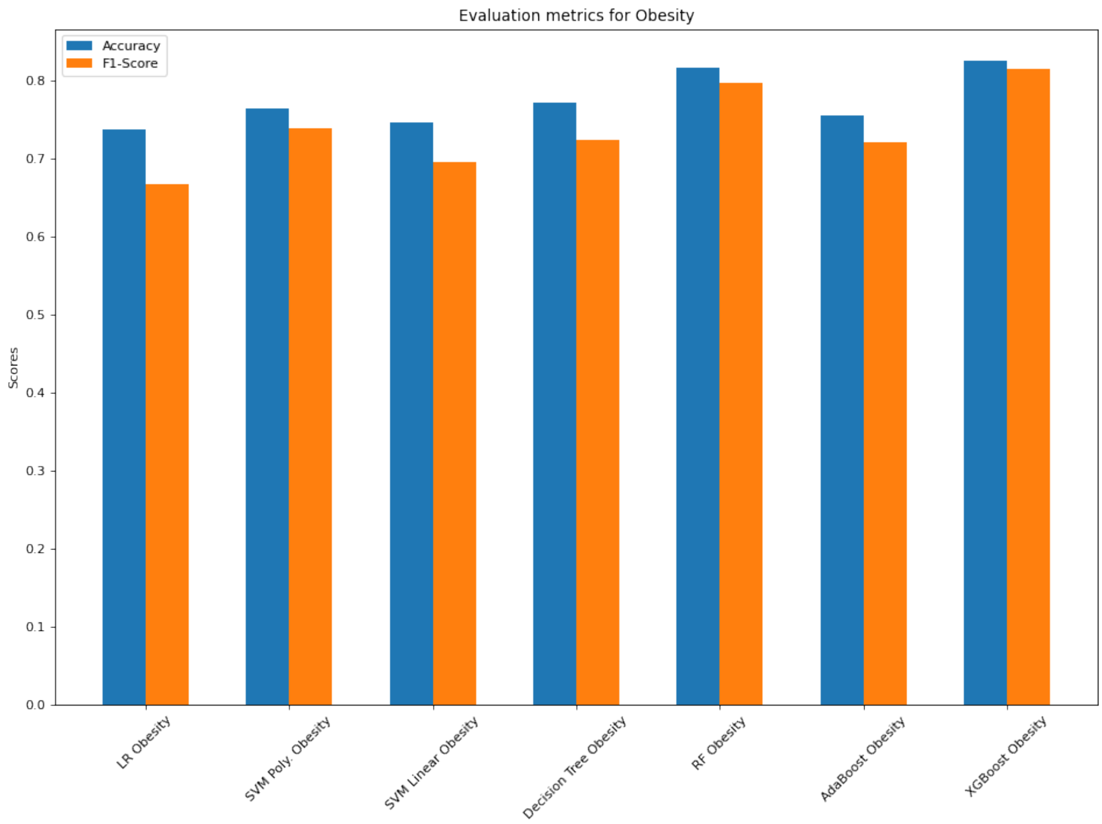
Algorithm	Accuracy	F1-Score
LR	0.751825	0.842593
SVM Poly.	0.766423	0.857143
SVM Linear	0.766423	0.857143
CART (DT)	0.686131	0.774869
RF	0.715328	0.8
AdaBoost	0.664234	0.755319
XGBoost	0.751825	0.826531

Hypertension - Evaluation Performance using all attributes



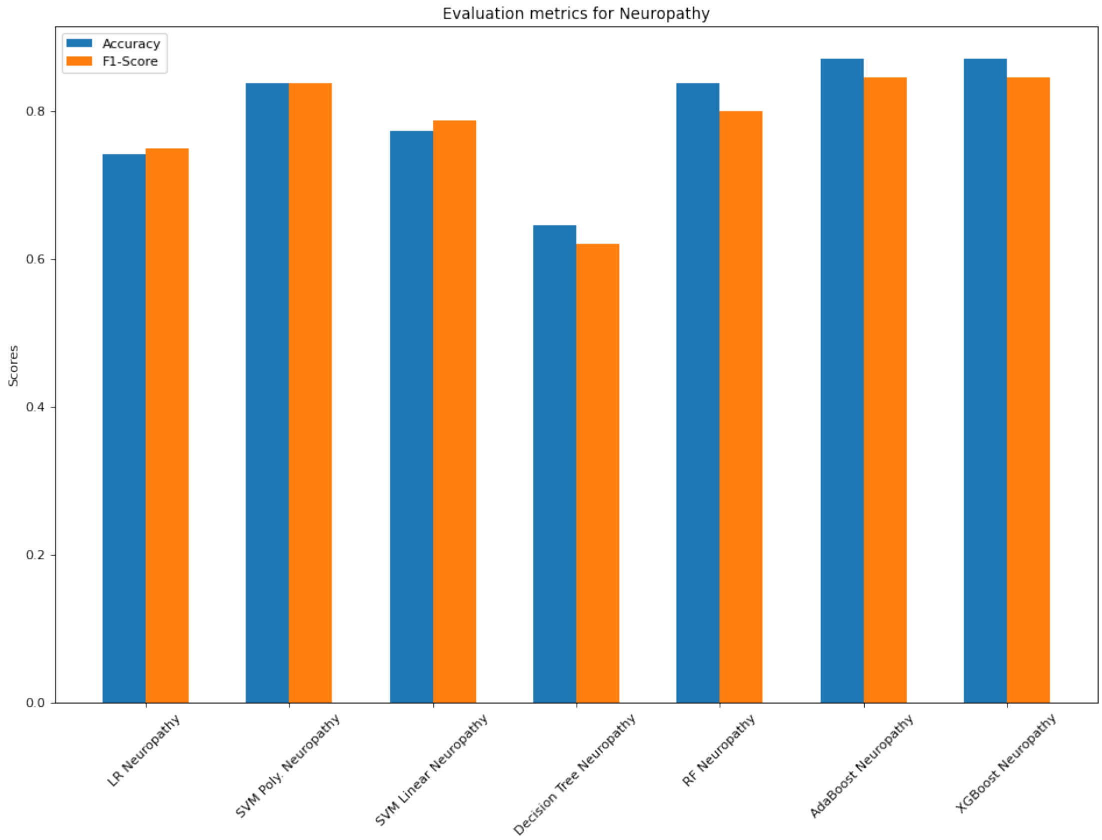
Algorithm	Accuracy	F1-Score
LR	0.729323	0.704918
SVM Poly.	0.706767	0.697674
SVM Linear	0.744361	0.730159
CART (DT)	0.62406	0.621212
RF	0.714286	0.707692
AdaBoost	0.691729	0.687023
XGBoost	0.744361	0.734375

Obesity - Evaluation Performance using all attributes



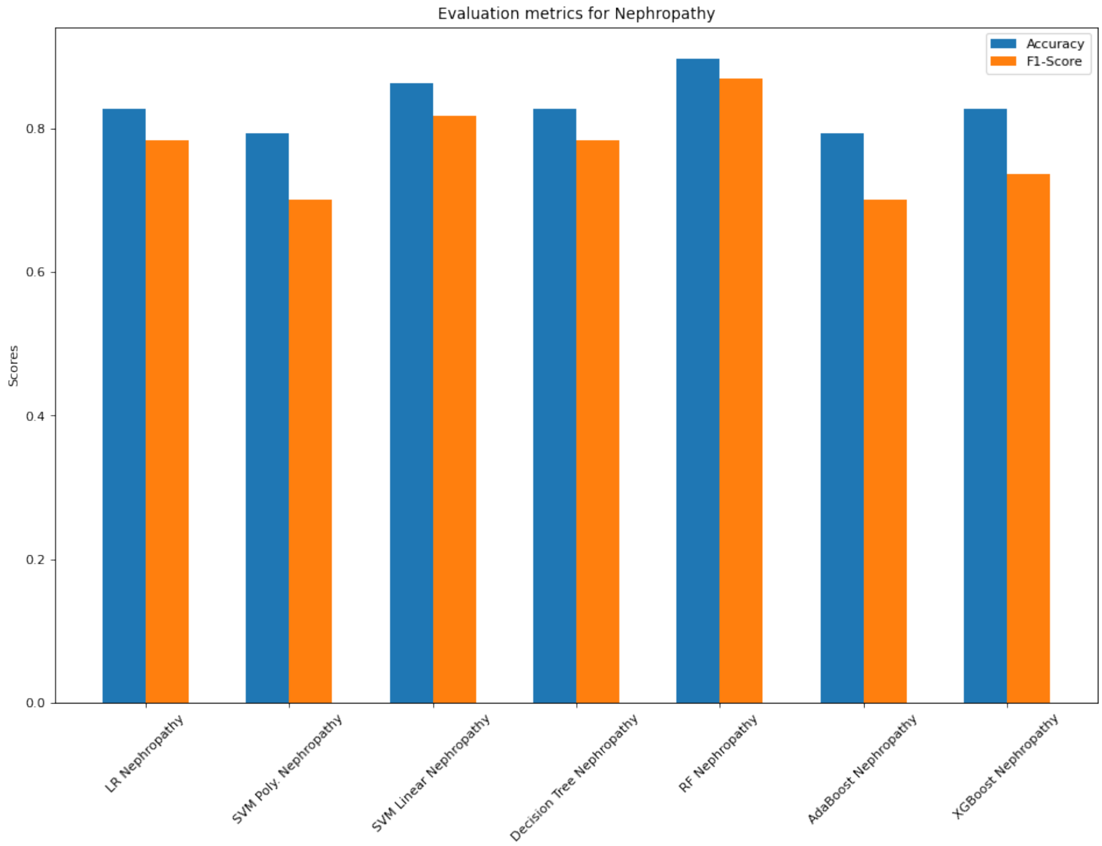
Algorithm	Accuracy	F1-Score
LR	0.736842	0.666667
SVM Poly.	0.763158	0.737864
SVM Linear	0.745614	0.694737
CART (DT)	0.77193	0.723404
RF	0.815789	0.796117
AdaBoost	0.754386	0.72
XGBoost	0.824561	0.814815

Neuropathy - Evaluation Performance using all attributes



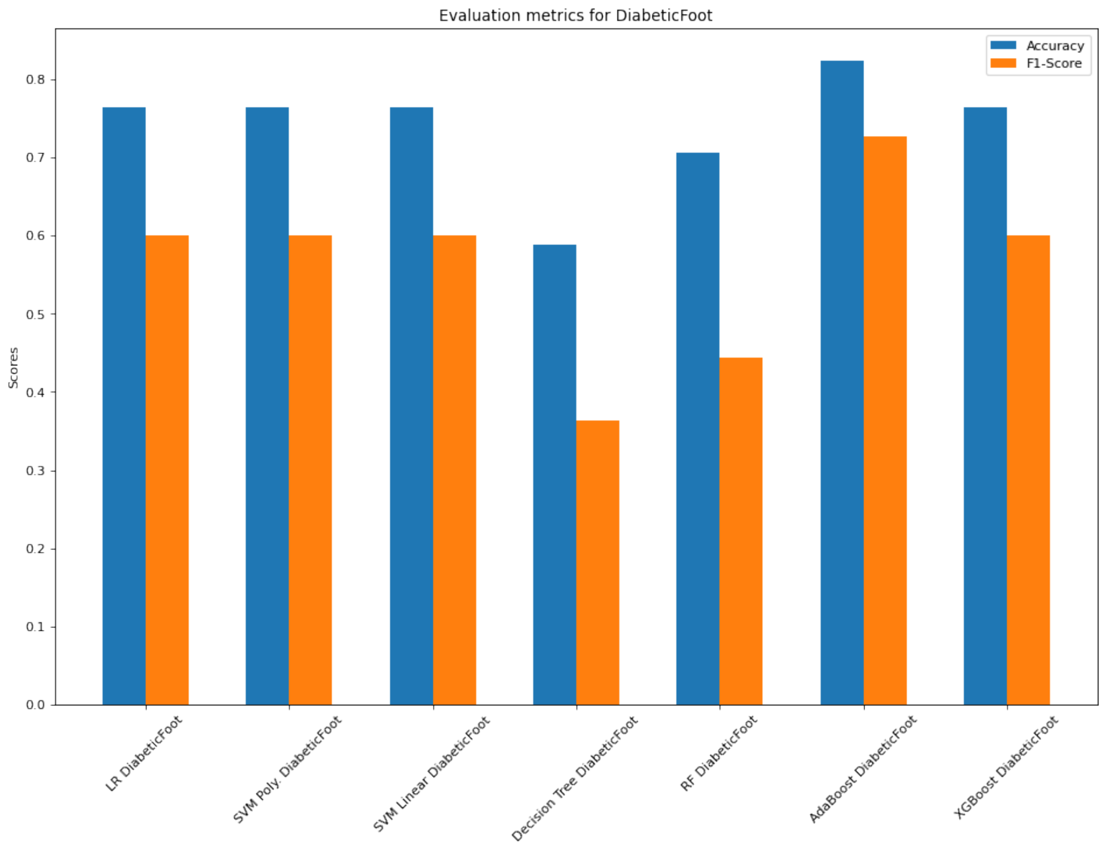
Algorithm	Accuracy	F1-Score
LR	0.741935	0.75
SVM Poly.	0.83871	0.83871
SVM Linear	0.774194	0.787879
CART (DT)	0.645161	0.62069
RF	0.83871	0.8
AdaBoost	0.870968	0.846154
XGBoost	0.870968	0.846154

Nephropathy - Evaluation Performance using all attributes



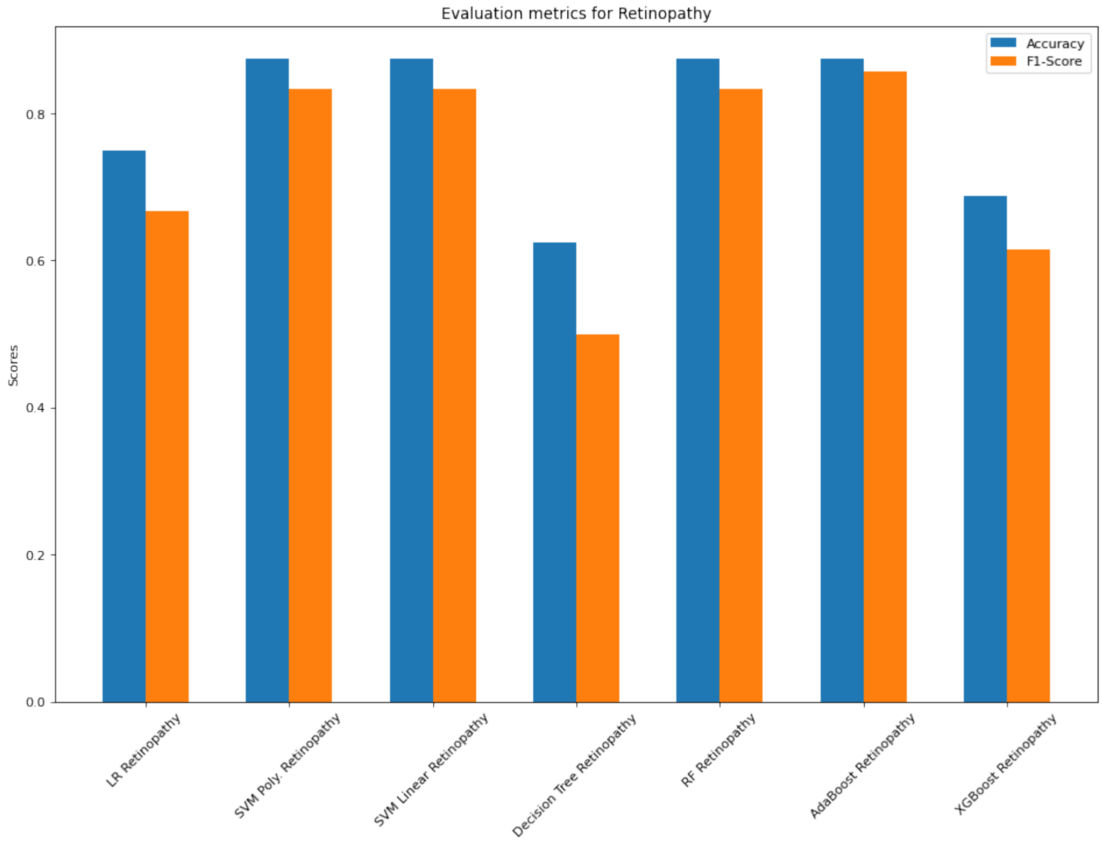
Algorithm	Accuracy	F1-Score
LR	0.827586	0.782609
SVM Poly.	0.793103	0.7
SVM Linear	0.862069	0.818182
CART (DT)	0.827586	0.782609
RF	0.896552	0.869565
AdaBoost	0.793103	0.7
XGBoost	0.827586	0.736842

Diabetic Foot - Evaluation Performance using all attributes



Algorithm	Accuracy	F1-Score
LR	0.764706	0.6
SVM Poly.	0.764706	0.6
SVM Linear	0.764706	0.6
CART (DT)	0.588235	0.363636
RF	0.705882	0.444444
AdaBoost	0.823529	0.727273
XGBoost	0.764706	0.6

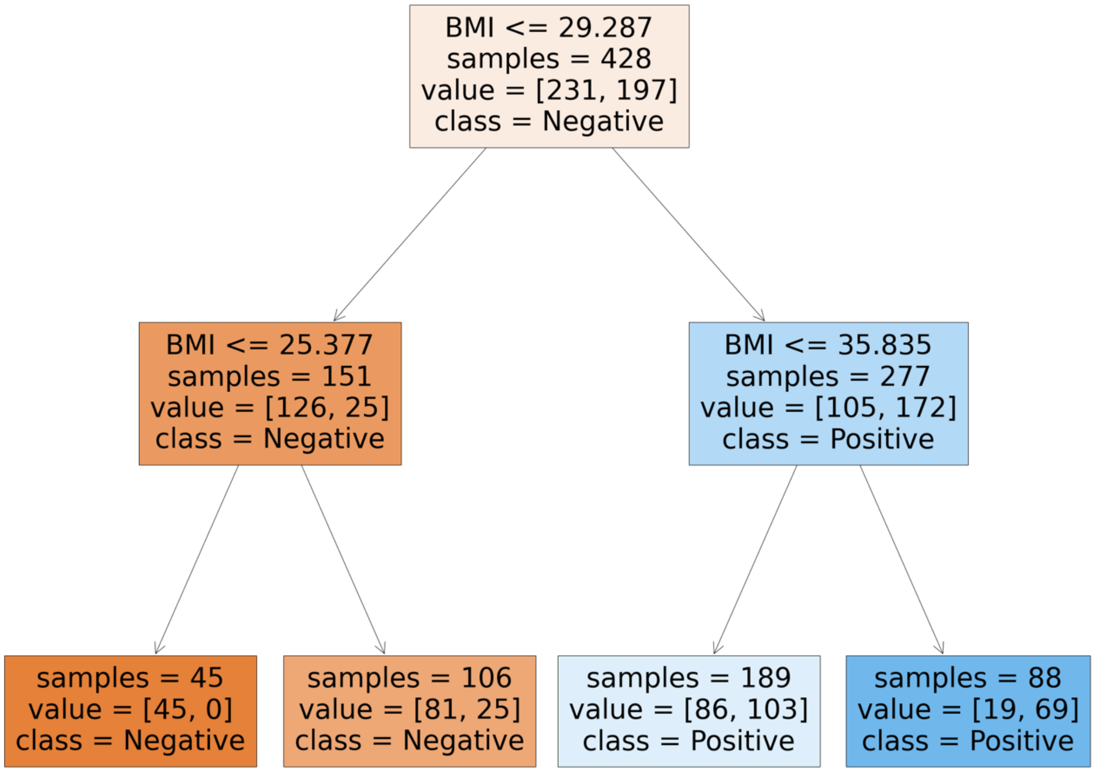
Retinopathy - Evaluation Performance using all attributes



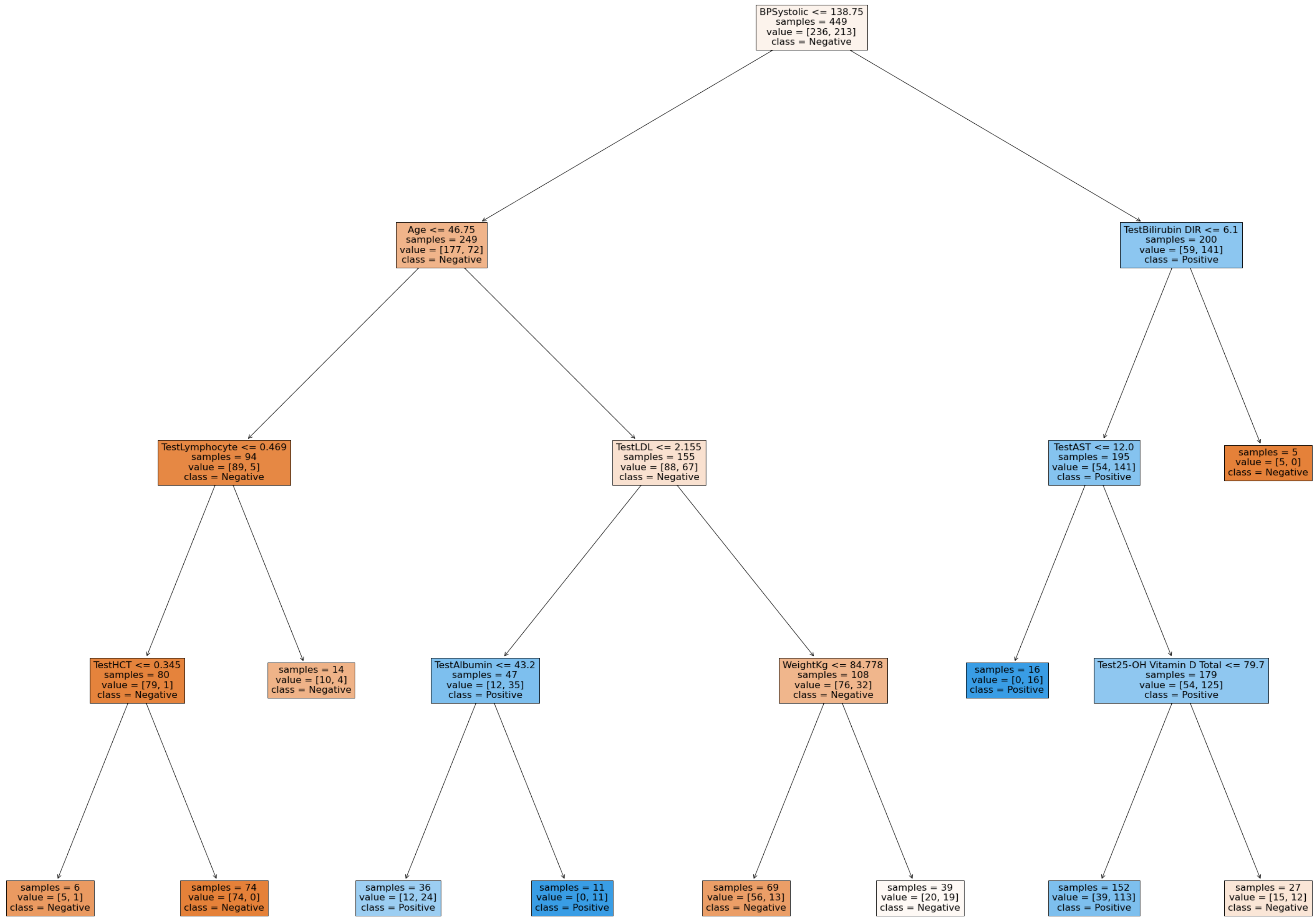
Algorithm	Accuracy	F1-Score
LR	0.75	0.666667
SVM Poly.	0.875	0.833333
SVM Linear	0.875	0.833333
CART (DT)	0.625	0.5
RF	0.875	0.833333
AdaBoost	0.875	0.857143
XGBoost	0.6875	0.615385

Appendix B

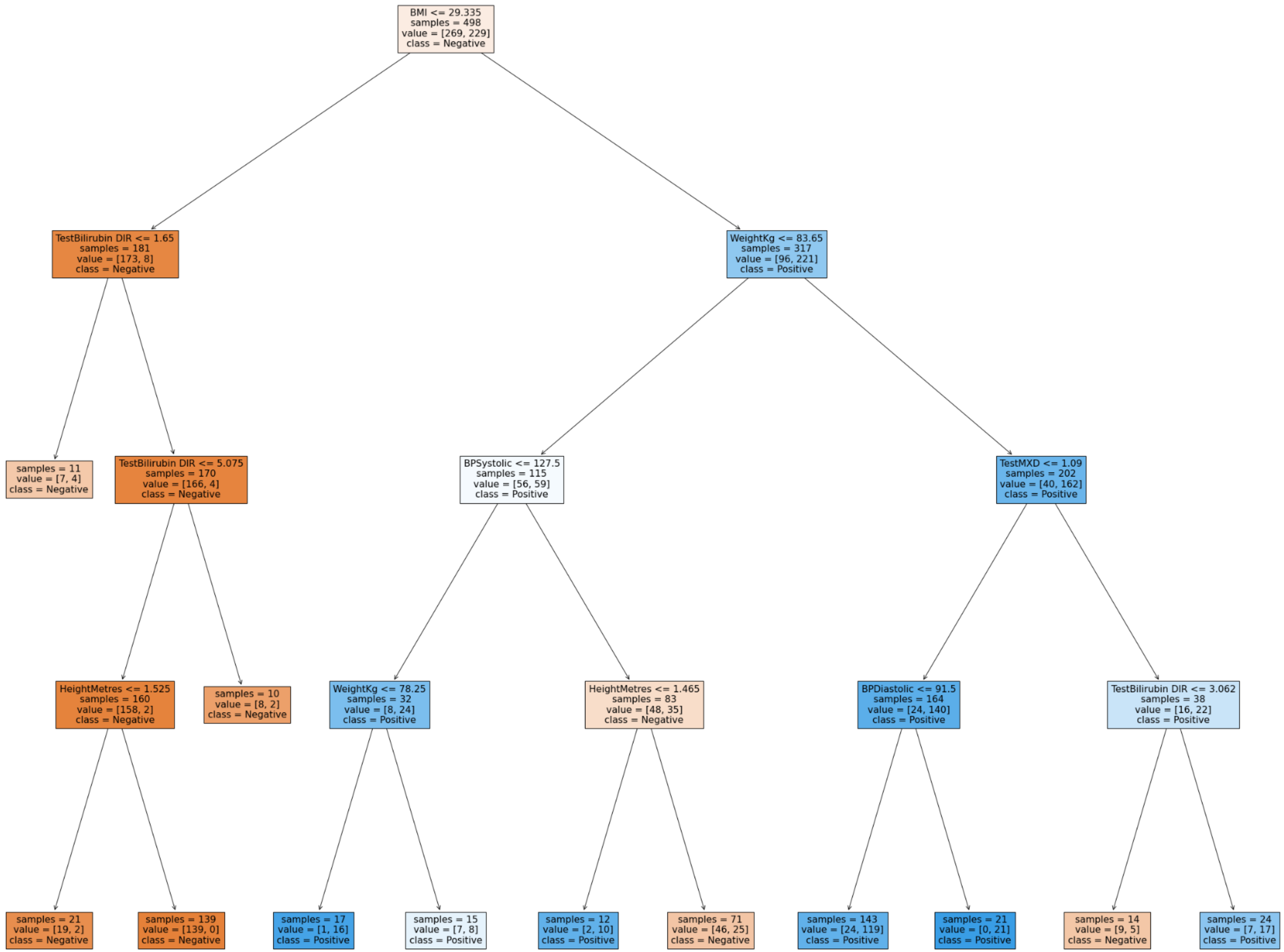
Decision Tree per complication using top 10 features. The blue color represents the positive class whereas the orange color represents the negative class.



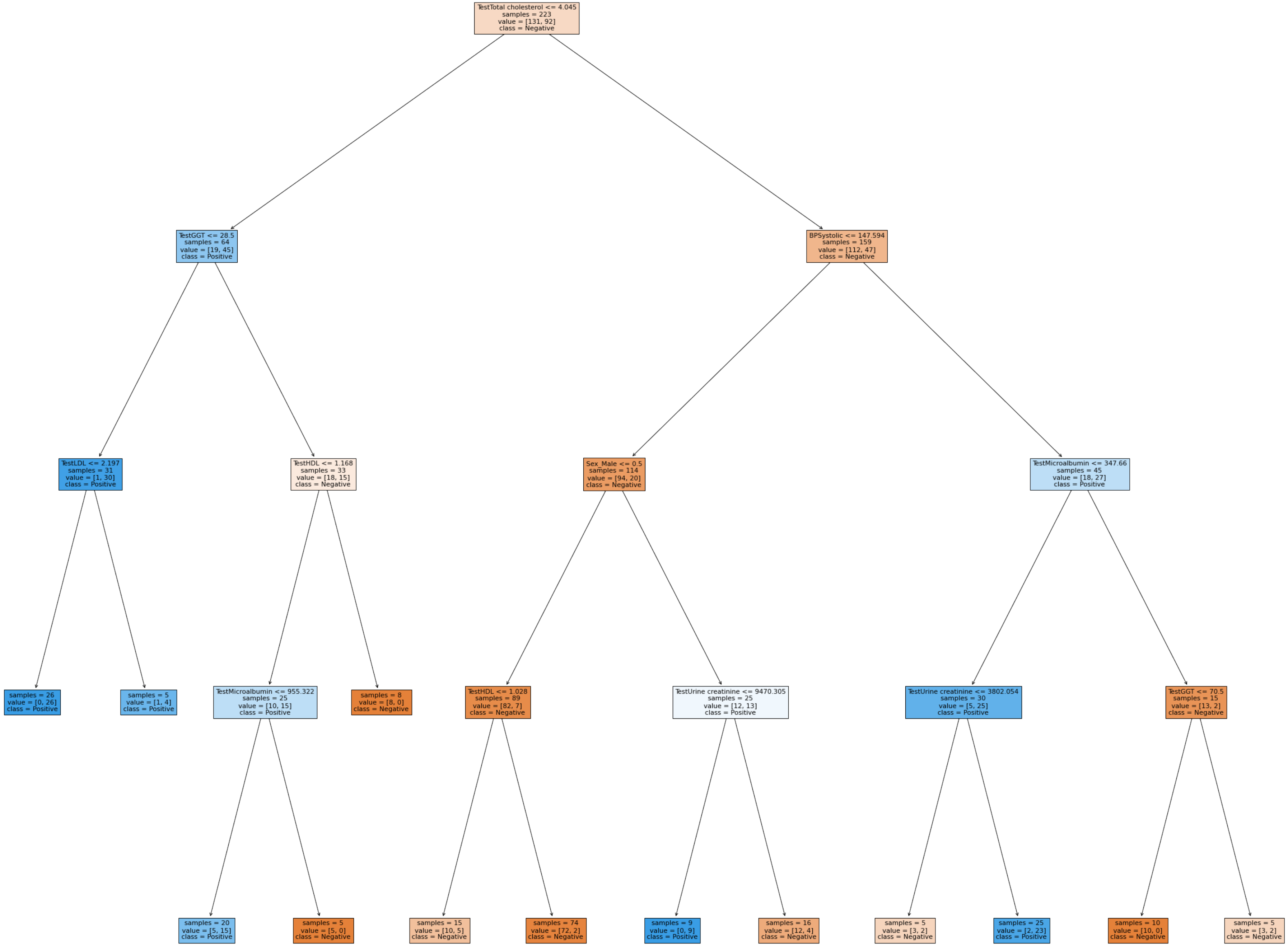
Decision Tree - Metabolic Syndrome



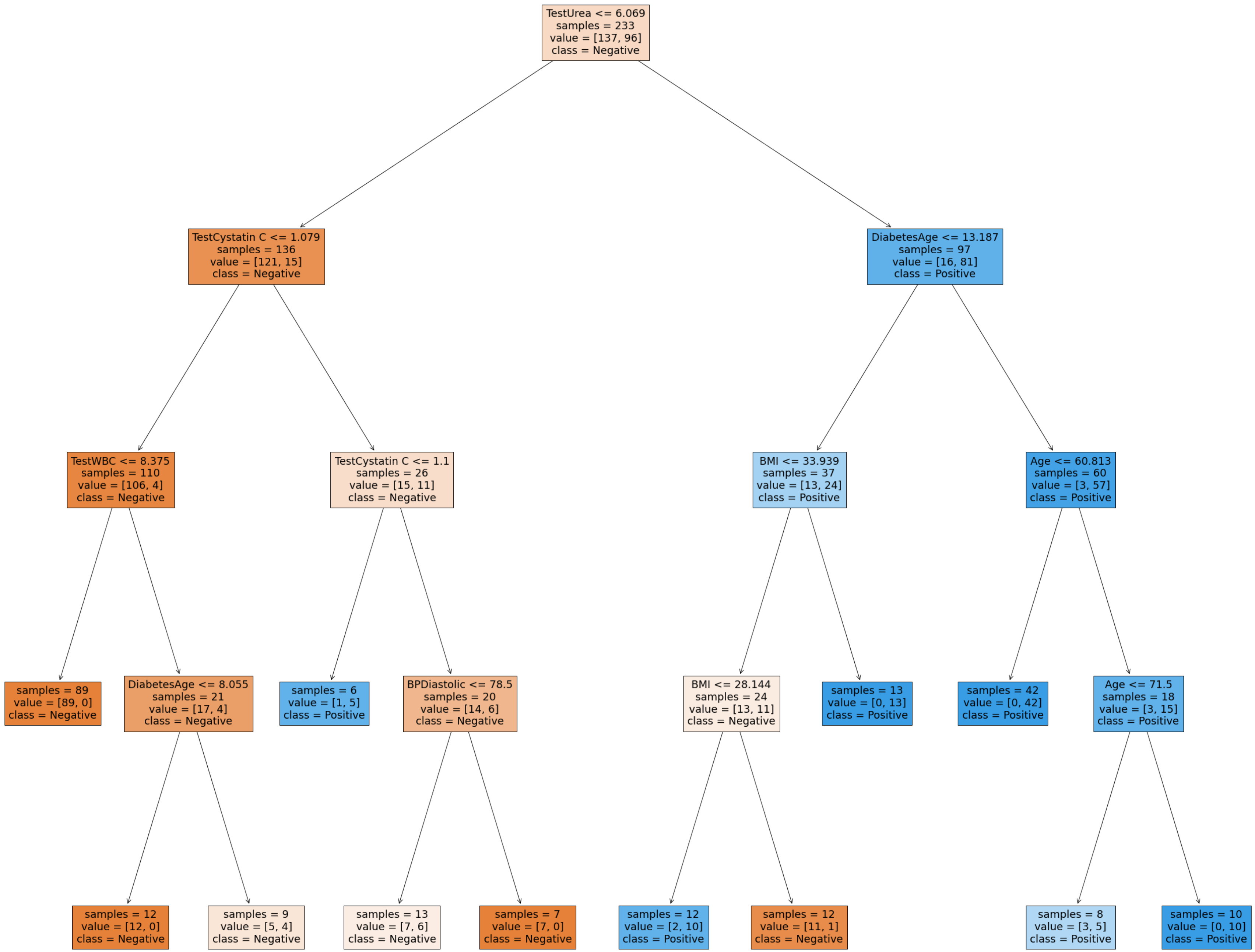
Decision Tree - Hypertension



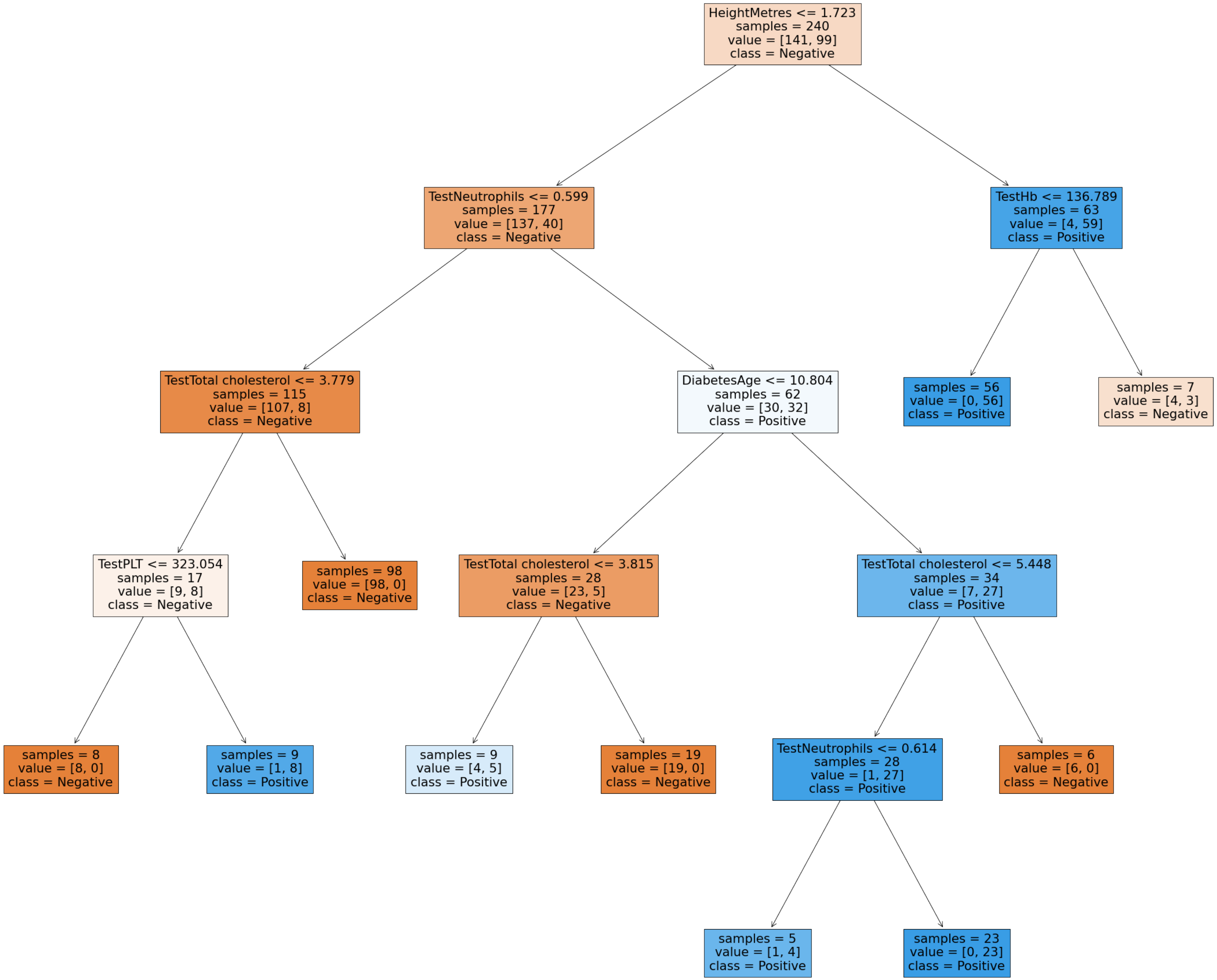
Decision Tree - Obesity



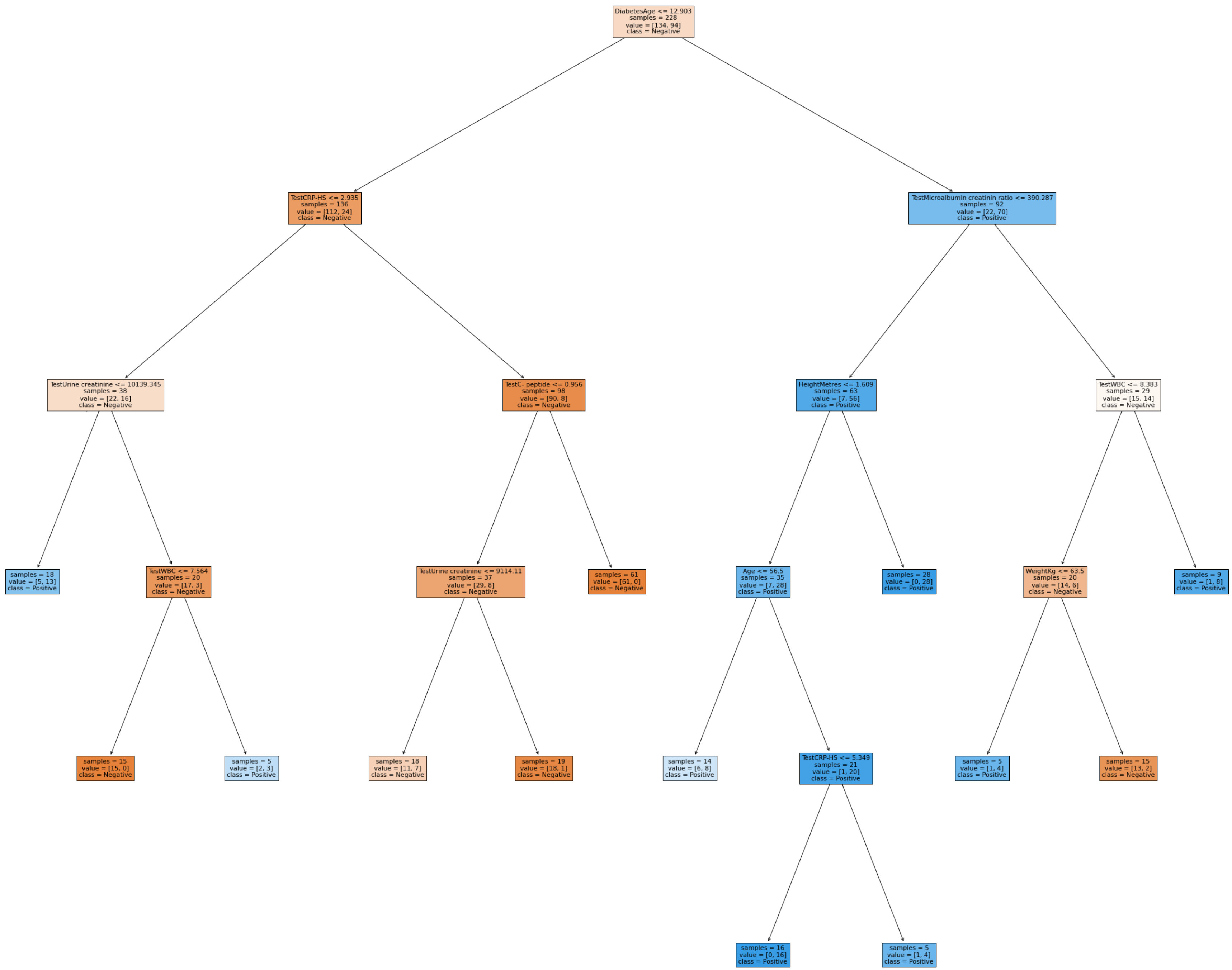
Decision Tree - Neuropathy



Decision Tree – Nephropathy



Decision Tree – Diabetic Foot



Decision Tree - Retinopathy

Vita

Yazan Jian was born in 1997, in Nablus, Palestine. He received his primary and secondary education in Nablus, Palestine. He received his B.Sc. degree in Computer Engineering from the An-Najah National University in 2019. From 2019 to 2020, he worked as a Software Engineer at Exalt Technologies Ltd.

In January 2020, he joined the Computer Engineering master's program in the American University of Sharjah as a graduate teaching assistant. During his master's study, he co-authored 1 conference paper which was represented in reputable international conference, in addition to 1 journal paper. His research interests are in Machine Learning and data mining.