Developing A Machine Learning Based Prognostic Model and A Supporting Web-based Application for Predicting The Possibility of Early Diabetes and Diabetic Kidney Disease

by

M.M. Shahriar Amin
17101048
Partho Mark Gomes
20241067
Jui Philomina Gomes
17301041
Faiza Tasneem
17141011

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science &
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
June 2021

© 2021. Brac University
All rights reserved.

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.

2. The thesis does not contain any material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

[illegible]	Partho
M.M. Shahriar Amin 17101048	Partho Mark Gomes 20241067
Gomes	Tasneem
Jui Philomina Gomes 17301041	Faiza Tasneem 17141011

Approval

The thesis/project titled ”Developing A Machine Learning Based Prognostic Model and A Supporting Web-based Application for Predicting The Possibility of Early Diabetes and Diabetic Kidney Disease” submitted by

1. M M Shahriar Amin (17101048)
2. Partho Mark Gomes (20241067)
3. Jui Philomina Gomes (17301041)
4. Faiza Tasneem (17141011)

Of Spring, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science & B.Sc. in Computer Science and Engineering on June 10, 2021.

Examining Committee:

Supervisor:
(Member)

Saiful

Md. Saiful Islam
Lecturer
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

[illegible]

Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

[illegible]

Sadia Hamid Kazi
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Ethics Statement

Our goal is to detect patients by providing them with an online assessment powered by a machine learning model. We have worked with extensive healthcare data to predict accurate results by processing patients' symptoms. The machine learning model used to score an individual based on a heuristic does not dive into any further analysis which provokes ethical dilemmas. We have assured complete transparency of our evaluation process in addition to providing visual interpretation of the output generated by the model.

Abstract

Machine Learning has gotten attention in the healthcare industry for the competences to ameliorate disease prediction. Machine learning has already been used in the health sector. Diabetes can also trigger the permanent loss of kidney function. Diabetic kidney disease (DKD) is one of the most recurrent diabetic micro vascular issues and has become the dominant cause of chronic kidney disease (CKD). It causes steady and permanent loss of kidney function. Kidney damage has been caused by poorly controlled diabetes that can damage the blood vessel clusters in the kidneys. Diabetic kidney damage normally develops over a long period of time. Therefore, there is a need for a machine learning model and application that can effectively predict and track the level of diabetes along with Diabetic kidney disease. In present studies, different classification algorithms such as Logistics Regression, Random Forest, Decision Tree, XGBoost show a notable accuracy to predict the early stage of diabetes. In this paper, our key motive is to find an efficient machine learning model to predict diabetes and diabetic kidney disease (DKD). Since, Disease Prognosis is a sensitive issue, it is not ethical to provide a result without extensive testing. Therefore, we have assessed our model using Recall, F-1 Score, Precision, AUC and also followed some robust evaluation metrics such as ROC, Sensitivity and Specificity to appraise performance of the models from the medical perspective. We are able to obtain an optimized prediction models using LightGBM with an accuracy of 98.75 % on diabetic kidney disease prediction and CatBoost with accuracy of 96.15% on diabetes prediction. We have also proposed a web application using our prognostic machine learning model to predict the result based on user input. This application can be used to predict the initial stage of the diabetes mellitus and diabetic kidney disease which may help to expedite the existing disease medication process.

Keywords: Early Diabetes Prediction, Diabetes Kidney Disease Prediction, Machine Learning, Diabetes, GBDT, Bioinformatics

Dedication

We would like to dedicate this thesis work to our loving parents. As well as, all the amazing faculties we came across and learnt from in the course of pursuing our Bachelors degree. It's been a worthwhile experience...

Acknowledgement

We would like to acknowledge that this report has been done under the supervision of Mr. Md. Saiful Islam, Lecturer at BRAC University. We express sincere gratitude to our supervisor Mr. Md. Saiful Islam and our co-supervisor Mr. Mirza Farhan Bin Tarek, Lecturer at BRAC University, for assisting and guiding us patiently throughout the year. Their valuable suggestions, ideas, support and effort have significantly helped us in our research work.

Keywords: Early Diabetes, DKD, Diabetes Kidney Disease, CatBoost,
Early Diabetes Prediction,Machine Learning,Diabetes,Bioinformatics

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols and abbreviation that will be later used within the body of the document

ANN	Artificial Neural Networks
DKD	Diabetic Kidney Disease
DT	Decision Tree
$GBDT$	Gradient-Boosted Decision Tree
SVM	Support Vector Machine
CKD	Chornic Kidney Disease

Chapter 1

Introduction

1.1 Diabetes and Diabetic Kidney Disease

Diabetes is a long-term illness that occurs when the body's ability to utilize insulin it generates is impaired. Hence, the body has difficulty in properly regulating the amount of dissolved sugar (glucose) in the bloodstream. Diabetes raises the chance of organ failure, particularly in the eyes, kidneys, heart, and blood vessels.
In addition, Diabetic Kidney Kidney disease (DKD) is the reduction of kidney function over time. This occurs as a result of excessive blood sugar levels causing kidney damage. The kidneys filter the blood, removing fluid, pollutants, and waste. When DKD develops due to Diabetes, kidneys do not function properly.
Around 30%-40% of people with diabetes may eventually develop diabetic kidney disease. Diabetic nephropathy is another name for diabetic kidney disease. In 2019, approximately 463 million adults (20-79 years) were living with diabetes, by 2045 which will rise to 700 million [29]. Diabetes caused about 4.2 million deaths in 2019. Diabetes is the primary cause of kidney failure, accounting for more than 44% of new cases [17].

Machine Learning is used to apply different techniques, methods and algorithm for developing any specific models from stored data. The use of classification-based techniques in disease diagnosis and treatment can drastically minimize medical errors and human costs. When compared to other data classification algorithms, Machine learning classification algorithms have shown to be effective in terms of prediction accuracy.
The accuracy of data classification may vary depending on the approach used. Classifications models deal with medical data collected from the patient to predict the early stage of diabetes and diabetic kidney disease (DKD). Classifiers such as Naïve Bayes, Logistics Regression, Support Vector Machine (SVM), Decision Tree and Artificial Neural Network (ANN) have been used earlier to predict diabetes and diabetic kidney disease separately.

1.2 Problem Statement

Diabetes is a deadly and chronic condition that results in an increase in the blood glucose levels in the human body. The complications due to diabetes may arise in all age groups of people if diabetes remains unidentified and untreated. According to the International Diabetes Federation, Bangladesh has 7.1 million diabetics and nearly as many undiagnosed diabetics. By 2025, this number is expected to double [31]. It is a costly condition and can lead to stroke, heart attack, severe kidney diseases, nephropathy, visual impairment and amputations.

Our vision is to predict to early diabetes as well as diabetic kidney disease (DKD) in order to treat it on time. With the rise in machine learning approaches, the initial identifying process is improving remarkably. Big data analytics may be used to examine large datasets and uncover hidden information and patterns, allowing users to gain knowledge from the data and anticipate outcomes accordingly. The categorization and prediction accuracy of the present approach is not very good. [16]. Due to the fact of COVID-19, it was quite impossible to collect the raw data from the hospitals. We have used a existing dataset which was found quite unpopular in case of usage. We have formatted the dataset to make it consistent.

In this paper, our motive is to develop a model and prediction system which can forecast the chances of diabetes and DKD in patients with utmost accuracy while ensuring to fit the best model possible upon the existing models. Therefore, we have used machine learning algorithms namely Logistic Regression, SVM, ANN, CatBoost in this study to detect diabetes in an primary stage and Diabetes Kidney Disease (DKD). Precision, Accuracy, F1-Score, ROC, and AUC have been used to evaluate the performance of all of these methods. Finally, we have further optimized two selected models CatBoost and LightGBM and evaluated the result using ROC, AUC, K-FOLD Cross Validation to determine the performance of the models.

1.3 Research Objective

Our goal is to design a system that can predict primary stage of “Early Diabetes” and “Diabetic Kidney Disease” also known as DKD, based on patients’ medical record/history, with the help of classification algorithms that utilize machine learning. The research we are conducting is to find the most efficient and accurate machine learning model to predict the possibility of Diabetes and Diabetic Kidney Disease (DKD) in a patient. For example, our system will take the input of the patient’s medical record and then tell him/her the possibility of developing Diabetes and Diabetic Kidney Disease using the developed model.

Input features consist of both numerical and nominal values in the data set. According to the literature we reviewed, most of them were conducted on diabetes prediction. That is why, in our research, we hope to develop an efficient and highly accurate system to predict the early stages of Diabetes and Diabetic Kidney Disease in a patient. Therefore, our system will be trained on multiple machine learning classifiers as well as utilize ensemble methods, to increase the accuracy of the models. The models will then be evaluated using multiple classifier evaluation techniques. In addition to using these evaluation results, we will develop a prediction system based on our machine learning model to find the possibility of Diabetes and Diabetic kidney disease in a patient.

1.4 Challenges Faced

It is worth noting that, to conduct the whole thesis work was quite challenging for us. As the world is now facing the most challenging time due to the pandemic outbreak of the COVID-19 disease, it was quite difficult to analyze research work smoothly. We had targeted initially to collect patients’ data from the hospitals in Bangladesh as our aim was to implement our model in the perspective of Bangladeshi people. However, we were unable to collect raw data from patients as it was quite impossible to be present in the hospitals physically for the COVID-19 situation and, we had gone through quite several lockdown phases with harsher measures. To conduct our research work, however, we have used a secondary dataset retrieved from Cloud Machine Learning Repositories which was found quite unpopular in case of usage and we had to format the dataset to make it consistent. Moreover, in the perspective of Bangladesh, it is difficult to find any reliable dataset related to Diabetes and its consequent diseases. Even if it can be found, the collected data might not be prevalent. Many of the hospitals do not want to disclose their data regarding any disease due to the privacy and other valid reasonings. Though we tried to collect dataset from some of the hospitals via virtual communication, they were not up to that state for co-operating in this regard due to the restrictions as well as the COVID-19 situation. Besides, we, the team members had to prepare our work while communicating virtually since it was impossible to meet outside physically.

1.5 Thesis Outline

The focus of this research was to create a model that could predict the early stages of Diabetes and Diabetic Kidney Disease based on a person's medical data. Also, utilizing our machine learning model, create an application that can provide a prediction result based on user input. The authors' goal was to come up with the best model for accomplishing this task and then optimize it for accuracy. Additionally, provide visual rationale for the outcome.

To begin with, in the first chapter (Chapter 1), we have provided the overview of Machine Learning in Health Care Services, the Problem Statement to address the recent issues related to the Diabetes and Diabetic kidney disease and the Research Objective to clarify the goals and the proposed works of the authors.
In the second chapter (Chapter 2), the work that has been done in the area of diabetes and dkd prediction using machine learning is discussed. Researchers' major findings are summarized. The inadequacies of previous approaches are also highlighted.
In the third chapter (Chapter 3), the fundamental assessment of several supervised algorithms, as well as their implementation specifics are explored. Later on, these related algorithms will be used in the research pipeline.
In the fourth chapter(Chapter 4), the Research Methodology and Workflow are introduced. The research is described in detail, including data collection, processing, and feature selection.
In the fifth chapter (Chapter 5), Hyper-parameter tweaking and Cross-validation are used to increase the accuracy metric of selected models. The various algorithms were subjected to a comparative analysis.

In the sixth chapter (Chapter 6), Experimental results and analysis are conducted, In the seventh chapter (Chapter 7), the design and workflow of the proposed disease prognosis application are introduced and in the last chapter (Chapter 8), we have concluded our paper and talked about our future research on this subject.

Chapter 2

Literature Review

2.1 Literature Review

Deepti Sisodia et al. (2018) analyzed three machine learning classification algorithms for early diabetes prediction namely Decision Tree, SVM, and Naive Bayes [27].
Diabetes Mellitus is one of the most widespread diseases in the world, and there is no cure for it. In most cases, there are few visible symptoms at the early stage of diabetes. Diabetes can be controlled if it can be predicted at an early stage.The dataset utilized in this experiment is called Pima Indians Diabetes Database (PIDD) and it is available at the UCI machine learning repository. [18]. Precision, Accuracy, F-Measure, and Recall were among the evaluation methodologies used to evaluate the performance of the three algorithms. In addition, the experiment uses the WEKA tool, a software developed at the University of Waikato in New Zealand [11] [2].
For data clustering, classification, regression, and visualization, the software contains a set of machine learning methods. The dataset (PIDD) has eight attributes for a total of 768 instances, all of whom are female patients. The value '0' is addressed as a resulted negative for diabetes in the dataset, while value '1' is treated as a tested positive for diabetes. The data from the PIDD dataset is being classified using the naïve Bayes, SVM and Decision tree algorithms. A naïve Bayes algorithm is a classification method that incorporates probabilistic machine learning. The Bayes Theorem is at the heart of the classifier [23]. SVM is a supervised algorithm that may be used for both classification and regression problems. In real life, a tree has numerous analogues, and it turns out that it has inspired a wide number of machine learning techniques, including classification and regression [23].

Internal cross-validation 10-folds were used in the experiments. This work was evaluated using accuracy, F-measure, recall, precision, and ROC (Receiver Operating Curve) measurements. Accuracy was measured over correct and incorrectly classified instances.

$$Accuracy = \frac{Number of correct predictions}{Total Number of predictions}$$

To calculate accuracy in terms of positives and negatives, the following formula was used:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision refers to how precise/accurate the model is in terms of how many of the anticipated positives are actually positive. When the costs of False Positive are high, precision is a good statistic to use [26].

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

Recall calculates the number of Actual Positives in the model capture by labelling it as Positive [26]

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

To compute a balance between precision and recall, we need the F1-Score [26].

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

After conducting the classification algorithm, the experiments came with 76.39% accuracy in Naïve Bayes, 65.10% accuracy in SVM and 73.82% accuracy with Decision Tree classifier [27]. Then using ROC, the classification algorithms were compared against each other. In the ROC, Naïve Bayes outperformed the other two classifiers.
The most accurate method was Nave Bayes. As a result, the naive Bayes machine learning classifier was able to predict the likelihood of diabetes with more precision in this trial. According to these classified instances, accuracy was calculated and analyzed. Performance of the individual algorithm was evaluated based on Correctly Classified Instances and Incorrectly Classified Instances out of a total number of instances. When compared to other algorithms, the Naive Bayes classification algorithm outperformed them with 76.30% accuracy in the prediction of early diabetes [27].

Makino et al.(2019) discussed in their research [30] that DKD is one of the most severe diabetes consequences, and if left untreated, it could lead to end-stage renal disease requiring hemodialysis (ESRD) [16].The virtual branch of AI includes informatics, which is intended to help doctors make better clinical diagnoses and treatment decisions. Machine learning has made significant strides recently, with

big data analysis, has been contributing greatly, especially in the field of clinical imaging [28] [19], pharmacokinetics [24], genetics [22] and oncology [21].
Here population-based analysis was used for assessment. In this paper, they used big data machine learning to create a new predictive model of DKD in diabetes patients, which was based on electronic medical information (EMR).

They have extracted instances with pertinent data from the EMR. Then, determination of patients with Type 2 Diabetes Mellitus (T2DM) was done. Three methodologies were used to extract clinical features from these patients: structural data, text data, and longitudinal data from the EMR.

Laboratory test, diagnosis, prescription, and ICD 10 codes were among the structural features recovered by AI. By using Natural Language Processing (NLP), AI was able to extract prior histories, current ailments, and prescriptions from EMR information. Then, in stage-1 DKD, 180-day event pairings were built between the reference point and the prediction target point. These enabled to find stable group and aggravation group while taking same number of samples by "Under-sampling" procedure.

At first, they sorted out the amount of longitudinal data of EMR information that affected DKD prediction. Using a Convolutional autoencoder, they extracted raw features for people in the stable and aggravation groups for the previous 6 months to the reference point of prediction for 24 selected parameters (whose longitudinal information would affect DKD). From here, a typical time series pattern was structured for both stable and aggravation group and thus found increase of CPK (creatine phosphokinase) and BMI (body mass index) for aggravation group.

Secondly, using Logistic regression analysis, AI built predictive model including longitudinal variables that were obtained by summarizing past 180-days' EMR records. They performed 5-fold cross validation to get a prediction assessment result for each fold and discovered an average accuracy of about 71% and an AUC of 0.743.. Adding of the feature categories (like- urinary protein observation to predict DKD stage) in the model had improved the performance.

Thirdly,through examining long term relationship with 180 days prediction using same two labels (stable and aggravation group) for patients, The DKD aggravation group had a greater rate of hemodialysis (HD) than the stable group.

Yu et al. (2010) proposed that the Support Vector Machine (SVM) modeling, a supervised machine learning method, was a viable classification approach for recognizing persons with common diseases like diabetes and pre-diabetes in the community. [9].

It is well known that logistic regression uses a pre-determined model to estimate the likelihood of a binary event occurring by fitting data to a logistic curve. The SVM model, on the other hand, proved to be a data-driven and model-free technique. When datasets are modest and a high number of variables are included, it may have effective discriminative power for classification. This technique has recently been used to improve disease detection procedures in the clinical setting and to develop an automated illness categorization system.

They used diabetes as an example to test the effectiveness of SVM in categorizing people into illness groups. In the United States, around 23.6 million people have diabetes, with nearly a third of them being unaware of it. Prediabetes is a condition in which blood glucose levels rise, increasing the risk of diabetes, heart disease, and stroke. It affects approximately 57 million individuals. Diabetes can be prevented with a change in lifestyle or medicine, according to recent research. As a result, early detection and diagnosis is an important step in preventing diabetes in those with prediabetes.

Their goal was to develop an SVM-based method for distinguishing those with prediabetes from those who did not. Simple clinical parameters that did not require laboratory tests, such as body mass index (BMI), family history of diabetes, and so on, were employed to create the SVM models. The accuracy of this method's predictions was then compared to that of logistic regression models using the same set of variables [9].

Leung et al. (2013) compared the results of seven machine learning algorithms in identifying optimal combinations of clinical and/or genetic markers predictive of DKD based on their dataset. Two decision trees (the classification and regression tree and the c 5.0 decision trees), a random forest, a Nave Bayes classifier, a neural network, partial least square regression, and a support vector machine were among the techniques used [13].

Due to complicated interactions between various risk factors such as hypertension, hyperglycemia, and other genetic variations, While efforts to find genetic structural and regulatory variations to explain the heritability of complex characteristics are ongoing, some researchers claim that simple variables could explain up to 50% of the population's achievable risk for common diseases caused by common polymorphisms. It's difficult to figure out what these gene-gene interactions are and how they affect clinical outcomes [5].

Machine learning methods, as opposed to traditional statistical methods like the chi-square test or logistic regression, can be used to detect hidden correlations between genetic changes and disease vulnerability. These machine learning technologies allow researchers to discover hidden patterns, redirect and reclassify data, and show their inter-relationships for decision-making in a clear and understandable manner. However, these computational applications and their utility in treating common diseases like type 2 diabetes have yet to be thoroughly investigated and tested.

Comparison of the performances of different machine learning methods was done by using different sub-groups of attributes:

1. Clinical and genetic.
2. Genetic only.
3. Clinical only.

The least optimal performance was attained in the training stage using Nave Bayes classification (nb) and partial least squares regression (pls). For all sub-groups of traits, support vector machines (svmRadial) and random forest (cforest), followed by neural network (nnet), had the best performance. SvmRadial was only slightly affected by parameter adjustment and maintained good performance even after data cross-validation.

Chapter 3

Working Plan

3.1 Background Analysis

3.1.1 General Supervised Algorithms

Logistic Regression:

Logistic regression is a statistical tool that machine learning has borrowed. It is a method for estimating discrete values from a group of independent variables. Logistic regression is the method of choice for binary classification issues (problems with two class values).

The logistic regression algorithm's main method is the logistic function. The logistic function, also known as the sigmoid function, was developed by statisticians to represent the characteristics of population growth in ecology, such as how it grows exponentially and eventually approaches the carrying capacity of the ecosystem. It's an S-shaped curve that can change the value of any real-valued integer between 0 and 1, but never exactly between those two places.

$$\frac{1}{(1 + e^{-}value)}$$

Where e is the natural logarithms' base, Euler's number or the EXP() function in the spreadsheet and value is the numerical value to be transformed. The values between -5 and 5 have been changed into the range 0 and 1 using the logistic function, as shown below.

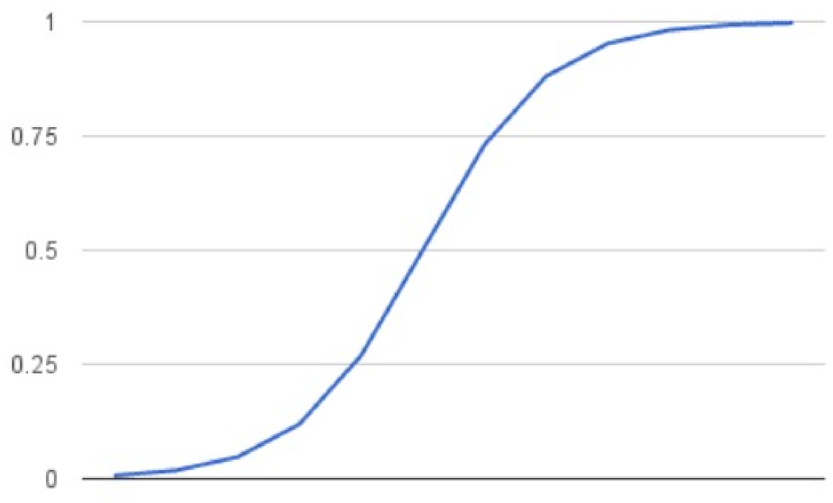


Figure 3.1: Logistic Function

Naive Bayes :

The Bayes theorem and the concept of predictor independence are the foundations of the Naive Bayes classification technique. In simple terms, a Naive Bayes classifier assumes that the presence of one feature in a class has no bearing on the presence of any other feature. The Naive Bayesian model is easy to build and is especially beneficial when dealing with large data sets.

Naive Bayes is renowned for outperforming the advanced classification systems due to its simplicity.

Bayes Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Support Vector Machine (SVM):

It is a discriminative classifier with a separating hyperplane as its formal definition. In addition, the system produces an ideal hyperplane that categorizes fresh samples based on labelled training data (supervised learning). This hyperplane is a line in two-dimensional space that divides a plane into two sections, with each class on either side [21]. This approach creates a model that assigns new samples to one of two categories, making it a binary linear classifier that is non-probabilistic.

There are many hyperplanes from which to pick when splitting the two types of data points. The objective is to find the plane with the biggest margin, or distance between data points from both classes.

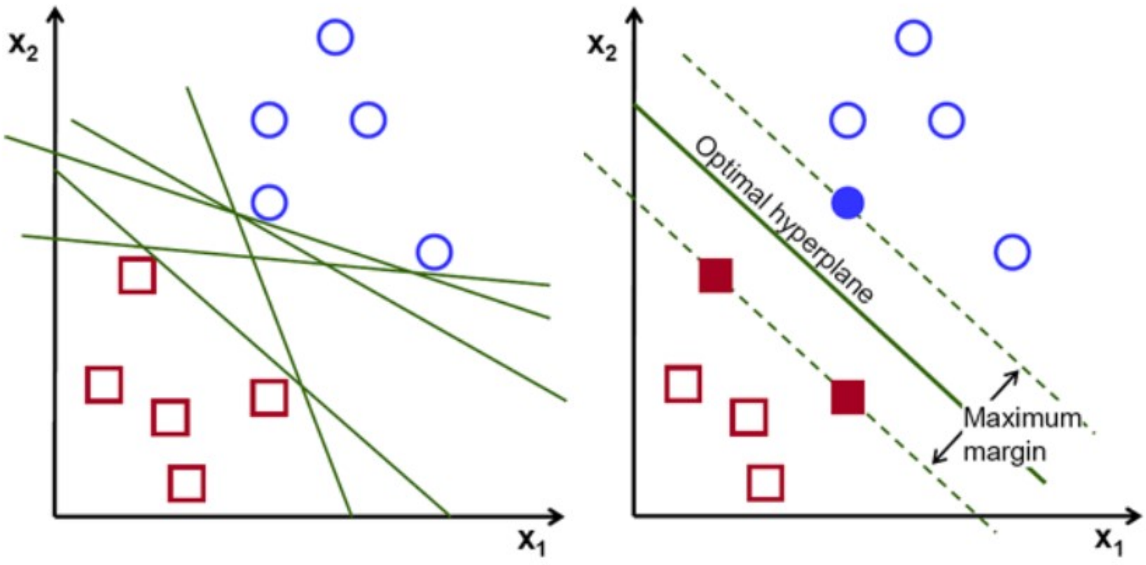


Figure 3.2: Possible hyperplanes

Maximizing the margin distance gives some reinforcement, making it easier to classify subsequent data points.

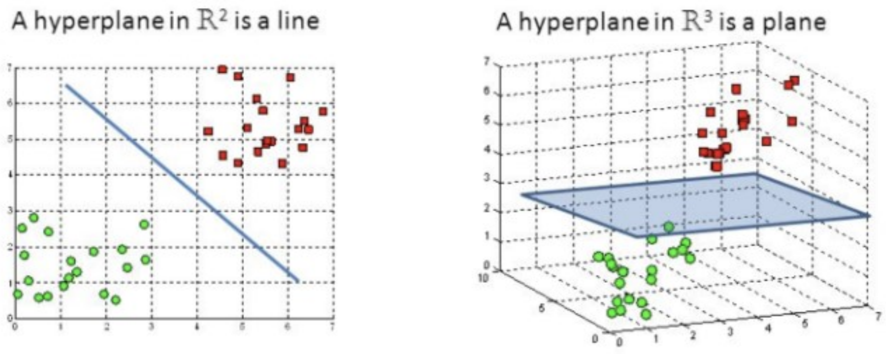


Figure 3.3: Hyperplanes in 2D and 3D feature space

The goal of the SVM method is to maximize the distance between the data points and the hyperplane. Hinge loss is a loss function that aids in margin maximization.

$$c(x, y, (f(x)) = (1 - y * f(x))_+$$

3.1.2 Ensemble Models

Decision Trees:

The decision-tree method is a supervised learning methodology that can handle both continuous and categorical output variables (regression and classification). it is constructed from the top down, segmenting the data into subsets containing samples with comparable values (homogenous). The homogeneity of a sample instance is calculated using the standard deviation. When the standard deviation of a sample is zero, it is considered entirely homogeneous.

$$(Standard Deviation).sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}((x_{i-\mu})^2, where(Mean)\mu = \frac{1}{N}\sum_{i=1}^{N} x_i}$$

$$Coefficient\ Of\ Variation(CV) = (\frac{\delta}{\mu}) \times 100$$

For two variables (Target, Feature), the standard deviation is:

$$S(T,X) = \sum_{c\epsilon X} P(c)S(c)$$

For the decision node, the attribute with the greatest standard deviation reduction is selected. The standard deviation reduction technique involves lowering the standard deviation after a data set has been separated by an attribute. To build a decision tree, identifying the feature that reduces standard deviation the most (SDR) is needed. For the decision node, the feature with the largest standard deviation reduction (SDR) is chosen.

$$SDR(T,X) = S(T) - S(T,X),$$
$$where\ T = Target,\ X = Feature\ and\ S = \ StandardDeviation$$

The data set is divided into two parts based on the values of the chosen characteristic. Until all of the samples in the dataset have been processed, the method is repeated for non-leaf branches. Typically, the coefficient of variation (CV) is used as a criterion for terminating recursion. if the CV for a specific branch goes below a specific level, such as 10%, the splitting process is ended and the average value is assigned at the leaf node for that subset.

3.1.3 Gradient Boosted Decision Tree

Gradient-boosted decision trees are a machine learning strategy for optimizing the predictive value of a model through progressive steps within the learning process. Each iteration of the decision tree includes altering the values of the coefficients, weights, or biases connected to each of the input factors being utilized to anticipate the target value, with the objective of minimizing the loss function. The loss function is a function that measures loss and error and represents the model's reliability. The higher the model's accuracy, the smaller the loss function is. GBDT now has a variety of loss functions [1][6][8].

The gradient is the incremental alteration made in each step of the process; boosting is a strategy of stepping up the change in predictive accuracy to adequately optimum value. In a nutshell, gradient boosting depending on regression trees is referred to as gradient boosting decision tree [12][3]. Gradient boosting is a gradient-based approach to learn a boosting classifier gradually, and estimates a function $f : R^n!R$ depending on a linear combination of weak learners $h : R^n!R$ as [12][3] :

$$f(x) = \sum_{k=1}^{M} \beta_k h_k(x : \theta_k)$$

GBDT algorithm utilizes a regression tree model of Classification and Regression Trees (CART) [4].

$$f(x) = \sum_{j=1} Jc_j : I(xIR_j)]$$

This equation indicates that the observed space is divided into J units (J leaf nodes), each with its own output value c_j. GBDT can solve practically any regression model using this equation and the gradient boosting approach. GBDT may, however, encounter the overfitting problem, which could be mitigated by displaying the learning rate n at each iteration. In GBDT, the learning rate is a regularization method. Freidman proposed an arbitrary sub sample to further improve the generalization of GBDT. [1]. The subsample fraction is offered as a way to analyze new training data sets arbitrarily without having to return to each step of the boosting process. Meanwhile, a fresh feature vector set is obtained and arbitrarily investigated in each phase without returning. The fresh training data sets and feature vector sets are used to fit the regression tree. The second method for regularizing GBDT is to use subsample division.

3.1.4 Neural Network

Artificial Neural Networks (ANN):

It's a well-known Neural Network model for classification and regression. The Artificial Neural Network (ANN) is a data processing method based on how the biological nervous system, such as the brain, processes information. It is made up of a huge number of densely connected processing elements (neurons) that work together to solve an issue.
An artificial neural network is based on the idea of joining numerous combinations of artificial neurons to produce more potent outputs. As a result, the conceptual structure of a typical artificial neural network looks somewhat like this:

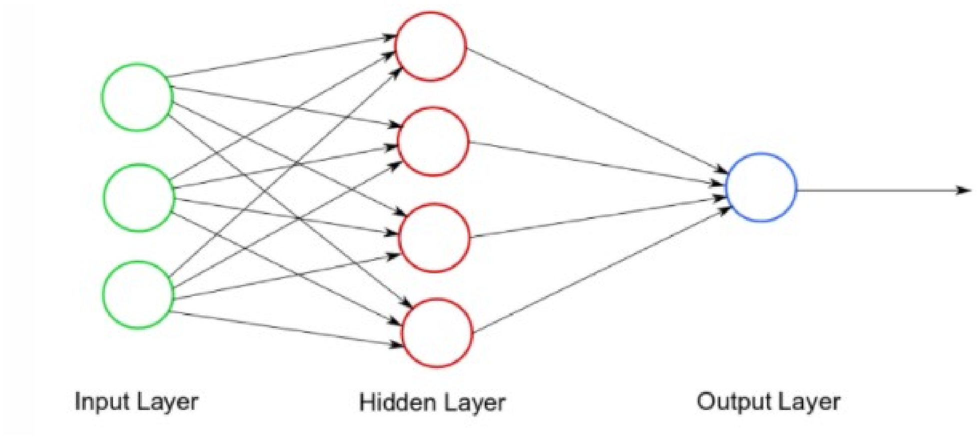


Figure 3.4: Artificial Neural Network Structure

The net input for the above general model of artificial neural network can be calculated as follows:

$$y_{in} = x_1.w_1 + x_2.w_2 + x_3.w_3...x_m.w_m$$

$$i.e.\ NetInput\ \ y_{in} = \sum_{i}^{m} x_i.w_i$$

The activation function can be applied to the net input to calculate the output.

$$Y = F(y_{in})$$

3.2 Workflow Diagram

The Figure 3.5 below show the workflow of our research.

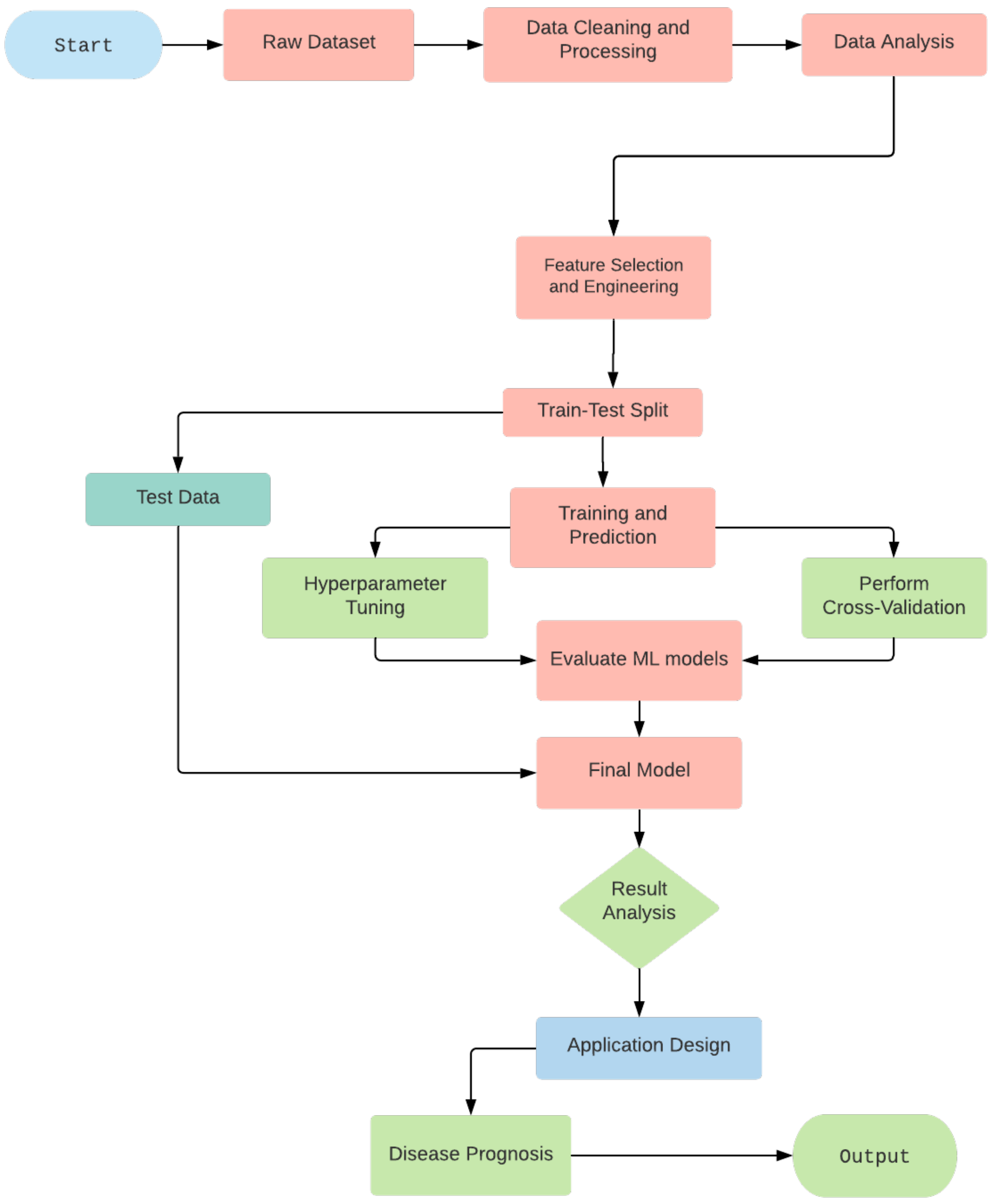


Figure 3.5: Workflow Diagram

Chapter 4

Research Methodology

4.1 Data Collection

For the earliest experiment, two different data sets have been used for training models. For the Prediction of Early Diabetes, a dataset for early stage diabetes by Islam M.M.F. et al. (2020) has been used for training the models [33]. Early Diabetes dataset carries 17 features in total namely- Age, Gender, Weakness, Visual blurring etc. and among all, the target feature here is ‘Class’ which gives the ultimate result of a patient having diabetes or not. Moreover, the feature ‘Age’ carries numeric value only and the rest of the features are having categorical values in table 4.1.

#	**Column Name**	**Count**	**Dtype**
0	Age	520	int64
1	Gender	520	object
2	Polyuria	520	object
3	Polydipsia	520	object
4	sudden weight loss	520	object
5	weakness	520	object
6	Polyphagia	520	object
7	Genital thrush	520	object
8	visual blurring	520	object
9	Itching	520	object
10	Irritability	520	object
11	delayed healing	520	object
12	partial paresis	520	object
13	muscle stiffness	520	object
14	Alopecia	520	object
15	Obesity	520	object
16	class	520	object

Table 4.1: Early Diabetes Dataset

In addition to this, we have worked on another dataset for our diabetic kidney disease (DKD) prediction model development [15] [20]. The dataset is used for training the corresponding machine learning models for predicting Diabetic Kidney Diseases. The Data set consists of 400 Instances with 25 attributes. It has 25 features like- age, bp(blood pressure), sg(sugar), rbc(red blood cell) etc. and 'class' is our target feature which defines whether a patient is having kidney disease or not. Again, among all features, 11 features are of nominal values and the rest 14 features are numeric and each feature is carrying 400 number of instances in table 4.2.

Table 4.2: DKD Dataset

#	Column Name	Count	Dtype
0	age	400	object
1	bp	400	object
2	sg	400	object
3	al	400	object
4	su	400	object
5	rbc	400	object
6	pc	400	object
7	pcc	400	object
8	ba	400	object
9	bgr	400	object
10	bu	400	object
11	sc	400	object
12	sod	400	object
13	pot	400	object
14	hemo	400	object
15	pcv	400	object
16	wbcc	400	object
17	rbcc	400	object
18	htn	400	object
19	dm	400	object
20	cad	400	object
20	cad	400	object
21	appet	400	object
22	pe	400	object
23	ane	400	object
24	class	400	object

4.2 Data Pre-Processing

DKD Dataset: The Datasets are saved as CSV format file and the csv files are read into a pandas dataframe for prepocessing. The datasets were filled with null/zero values initially. First, the ”null” values are replaced with np.nan using pandas built in `.replace()` function.

The primary target was how to deal with the missing values in the dataset. Following is the ratio of missing values in DKD Dataset in 4.3:

Table 4.3: DKD Dataset Missing Value Ratio

Column Name	**Missing Value Ratio**
age	2.25
bp	3.00
sg	11.75
al	11.50
su	12.25
rbc	38.00
pc	16.25
pcc	1.00
ba	1.00
bgr	11.00
bu	4.75
sc	4.25
sod	21.75
pot	22.00
hemo	13.00
pcv	17.75
wbcc	26.50
rbcc	32.75
htn	0.50
dm	0.50
cad	0.50
appet	0.25
pe	0.25

ane	0.25

Zero numerical and NaN values have been resolved using Pandas `DataFrame.mode()` function.

Diabetes Dataset: The Diabetes dataset has no null and zero values. So it is directly prepared for further processing. The Dataset has categorical values with numerical value in the "age" column only.

4.3 Feature Selection & Engineering

Feature Selection: The criteria for determining which features in the dataset are the most relevant is based on a variety of parameters. A feature with the highest correlation or variation with the target could be considered highly relevant. As a result, less important features are deleted to enable the model to generalize and interpret the new data.

Feature Engineering: Feature engineering is the technique of selecting and transforming variables from raw data to design a predictive model in machine learning. It helps to prepare the proper inputs from the datasets and make features to be compatible with the machine learning algorithm requirements.

To identify relationship between each features in both datasets, we have used Correlation finding technique to understand the impact of the each features on target value. Features whose value did not impact the class prediction were observed so that they might be given less priority in the final model. Correlation Matrix has been extracted using .corr() function.

Listing 4.1: Correlation Function

```
corrmat = dfn.corr()
```

Later, a correlation heatmap has been generated for better understanding of the relation between each feature on the dataset along with correlation matrix.

Listing 4.2: Correlation Heatmap

```
top_corr_features = corrmat.index
plt.figure(figsize=(14,8))
#plot heat map
g=sns.heatmap(dataset[top_corr_features]
                    .corr(),annot=True,cmap="RdYlGn")
```

Diabetes :

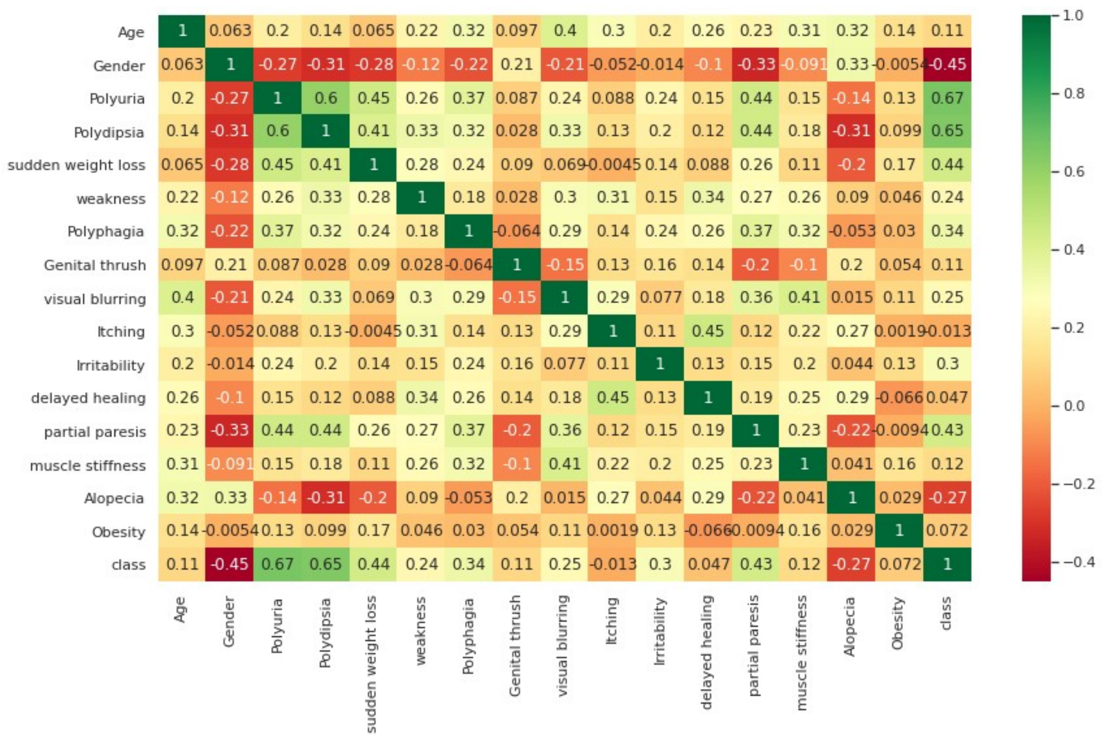


Figure 4.1: Correlation Heatmap for Diabetes Dataset

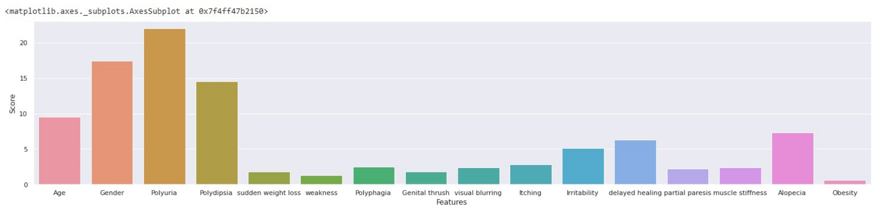
In the correlation heatmap, we can see features called ”polyuria”, ”polydipsia”, ”sudden weight loss”, ”partial paresis” have significant impact on the final predictive result. Features called ”delayed healing” and ”itching” have notable relation in between. We have put less priority on the features in our final models which correlation scores with other features are less significant.

Feature engineering and selection are set of procedures that typically require repeated attempts to perfect. We have implemented “get_feature_importance()” method from CatBoost ML Library, a gradient boosted library,to change and remodify feature selection. We may discover notable relationships that require the adding of more variables for mandatory feature engineering. Furthermore, these techniques frequently incorporate a combination of domain expertise and statistical data quality.

Listing 4.3: get_feature_importance task

```
model_fi= CatBoostClassifier(iterations=50, random_seed=42,
          logging_level='Silent')
          .fit(X1,Y1)
feature_importances = model_fi
                    .get_feature_importance(Pool(X1, label=Y1),
                     type="PredictionValuesChange")
feature_names = X1.columns
```

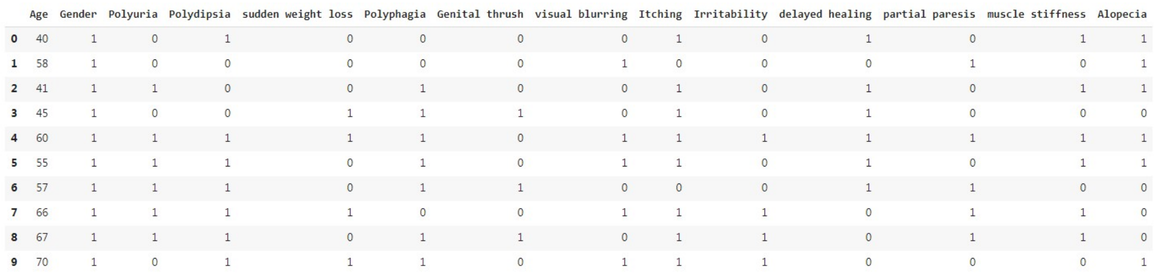
```
data_tuples = list(zip(feature_names,feature_importances))
dfne=pd.DataFrame(data_tuples, columns=['Features','Score'])
fig_dims = (25, 5)
fig, ax = plt.subplots(figsize=fig_dims)
sns.barplot(x = "Features", y = "Score", ax=ax, data=dfne)
```



```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4ff47b2150>
```

Figure 4.2: feature_importance graph for Diabetes

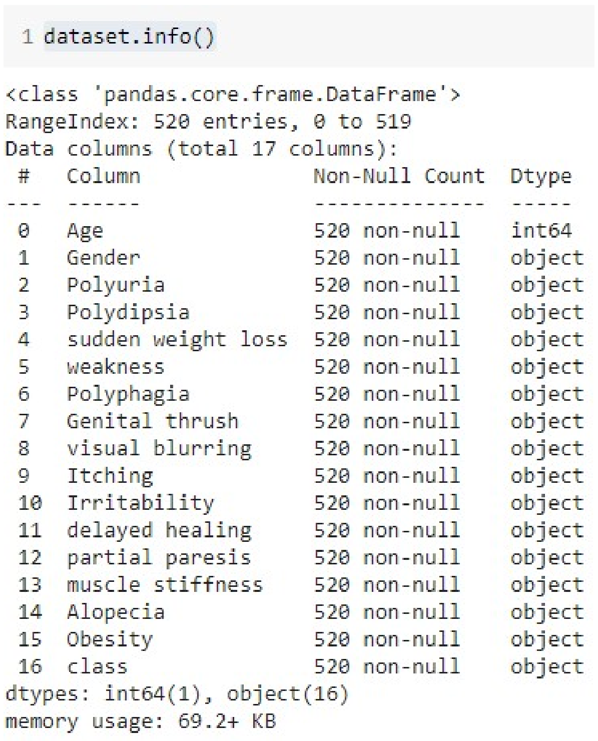
The Label encoding technique has been used for the categorical values to fit in the machine learning models. Although, both CatBoost and LGBM ML libraries can handle categorical data with parameter tuning, we have processed the categorical features using the label encoder which replace the categorical values with a numeric value between 0 and the number of features' class minus 1.



	Age	Gender	Polyuria	Polydipsia	sudden weight loss	Polyphagia	Genital thrush	visual blurring	Itching	Irritability	delayed healing	partial paresis	muscle stiffness	Alopecia
0	40	1	0	1	0	0	0	0	1	0	1	0	1	1
1	58	1	0	0	0	0	0	1	0	0	0	1	0	1
2	41	1	1	0	0	1	0	0	1	0	1	0	1	1
3	45	1	0	0	1	1	1	0	1	0	1	0	0	0
4	60	1	1	1	1	1	0	1	1	1	1	1	1	1
5	55	1	1	1	0	1	0	1	1	0	1	0	1	1
6	57	1	1	1	0	1	1	0	0	0	1	1	0	0
7	66	1	1	1	1	0	0	1	1	1	0	1	1	0
8	67	1	1	1	0	1	1	0	1	1	0	1	1	0
9	70	1	0	1	1	1	0	1	1	1	0	0	0	1

Figure 4.3: Label Encoding

Before label encoding the categorical values, we have made sure that every categorical columns have data type ”object” and numerical columns have data type ”float64” or ”int” . We have used .info() method to see the data type of each features in Diabetes dataset.



```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 520 entries, 0 to 519
Data columns (total 17 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Age                 520 non-null    int64
 1   Gender              520 non-null    object
 2   Polyuria            520 non-null    object
 3   Polydipsia          520 non-null    object
 4   sudden weight loss  520 non-null    object
 5   weakness            520 non-null    object
 6   Polyphagia          520 non-null    object
 7   Genital thrush      520 non-null    object
 8   visual blurring     520 non-null    object
 9   Itching             520 non-null    object
 10  Irritability        520 non-null    object
 11  delayed healing     520 non-null    object
 12  partial paresis     520 non-null    object
 13  muscle stiffness    520 non-null    object
 14  Alopecia            520 non-null    object
 15  Obesity             520 non-null    object
 16  class               520 non-null    object
dtypes: int64(1), object(16)
memory usage: 69.2+ KB
```

Figure 4.4: Data type of the features in diabetes dataset

We have dropped the features called ”weakness” and ”Obesity” to optimize the performance of our model and reduce the number of user inputs in the application because the features called ”weakness” and ”obesity” have no significant impact on the predicted result in Figure 4.3.

DKD Dataset:

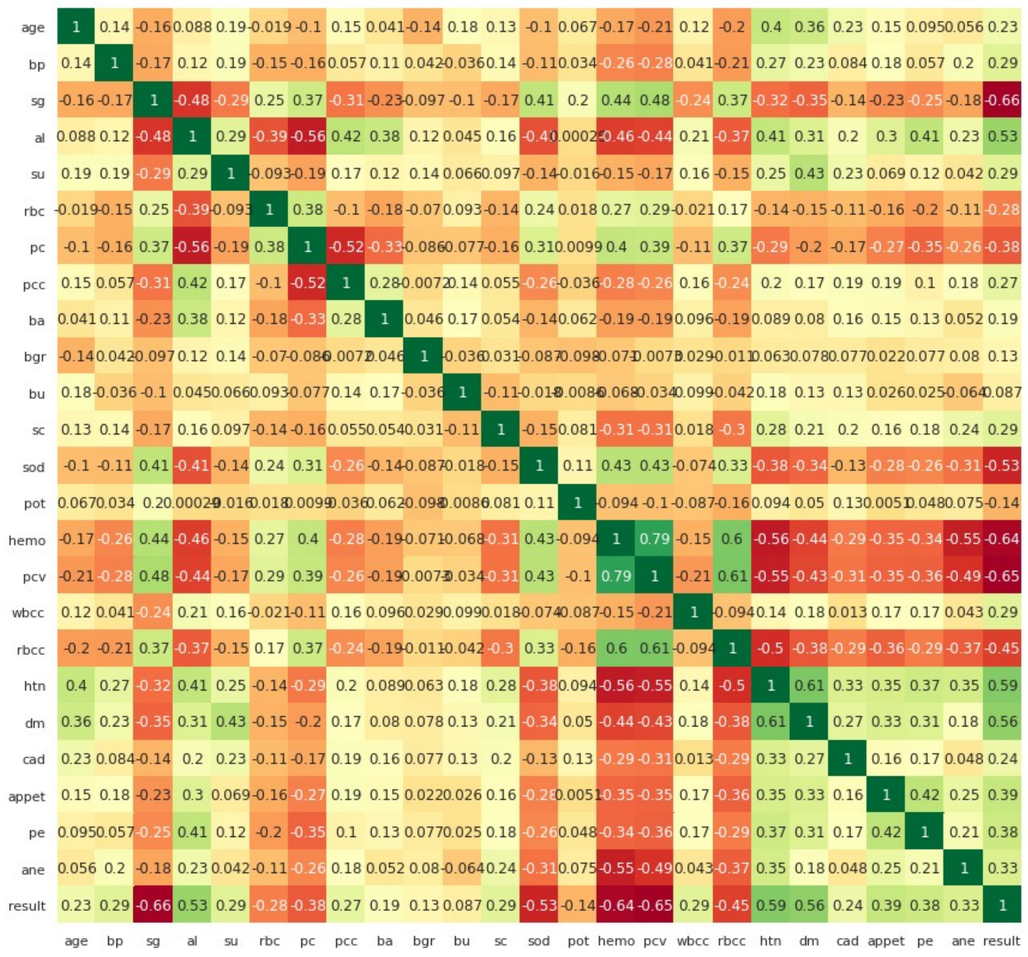


Figure 4.5: Correlation Heatmap for DKD Dataset

In the DKD Dataset, we can see the features called "htn", "dm", "appet" , "pe" , "ane" ,"al", have large impact on the predictive result which indicate the positive and negative outcome of patient's diabetic kidney disease. The features called "pcv" and "hemo" has significant correlation between themselves which is 0.79, features called "htn" and "dm" have also notable correlation among themselves, features "al", "su", "appet" and "cad" have notable correlation with the feature called "dm". We have further extracted the feature importance of the each features using "get_feature_importance()" method from CatBoost ML library. The "get_feature_importance()" method has been implemented using "PredictionValueChange" of CatBoost Library.

PredictionValuesChange :

This denotes the discrete importance values for each of the features in the dataset. For each feature, "PredictionValuesChange" reveals the average of the prediction changes if the feature values modify. The greater the value of the importance is, the more extensive changes on average in the prediction value occurs when the feature is modified [25] [32].

$$feature\_importance_F = \sum_{trees,leafs_F} (v_1 - avr)^2 . c_1 + (v_2 - avr)^2 . c_2,$$

$$avr = \frac{v_1 . c_1 + v_2 . c_2}{c_1 + c_2}, where$$

c_1,c_2 is the total weight of the object in left and right leaves consecutively. This weight is equal with the number of objects in each leaf when the weight is not mentioned in the dataset.

v_1, v_2 is the value for the formula in the left and right leaves consecutively.

The feature importance graph of DKD Dataset is shown below:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69f1c70410>
```

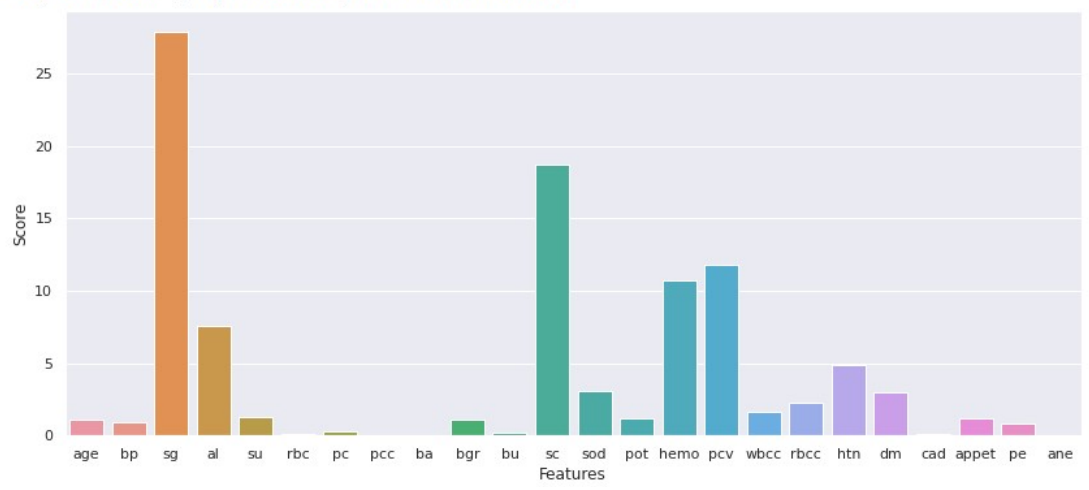
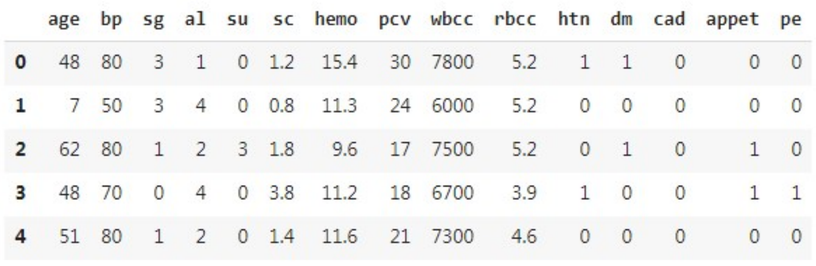


Figure 4.6: feature_importance graph for DKD

After conducting feature analysis in DKD dataset, we have dropped the features called 'rbc', 'pc', 'bgr', 'bu', 'sod', 'pot', 'ba', 'pcc', 'ane', since these features do not have any notable impact on the prediction result. However, when we will analyse the data using our machine learning models we may need to add or remove more features to optimize the overall performance of our model.

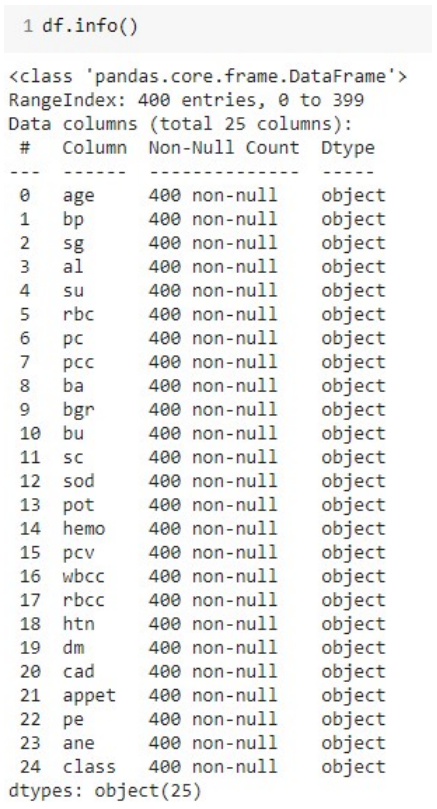
The Label encoding technique has been used for the categorical values to fit in the machine learning models. We have processed the categorical features using the label encoder which replaces the categorical values with a numeric value between 0 and the number of features' class minus 1. The numerical values are remain intact in DKD Dataset [10].



	age	bp	sg	al	su	sc	hemo	pcv	wbcc	rbcc	htn	dm	cad	appet	pe
0	48	80	3	1	0	1.2	15.4	30	7800	5.2	1	1	0	0	0
1	7	50	3	4	0	0.8	11.3	24	6000	5.2	0	0	0	0	0
2	62	80	1	2	3	1.8	9.6	17	7500	5.2	0	1	0	1	0
3	48	70	0	4	0	3.8	11.2	18	6700	3.9	1	0	0	1	1
4	51	80	1	2	0	1.4	11.6	21	7300	4.6	0	0	0	0	0

Figure 4.7: Label Encoding on DKD

Before label encoding the categorical values we have made sure that every categorical columns have data type "object" and numerical columns have data type "float64" or "int" . We have used .info() method to see the data type of each features in DKD dataset as in the following figure.



```
df.info()
```

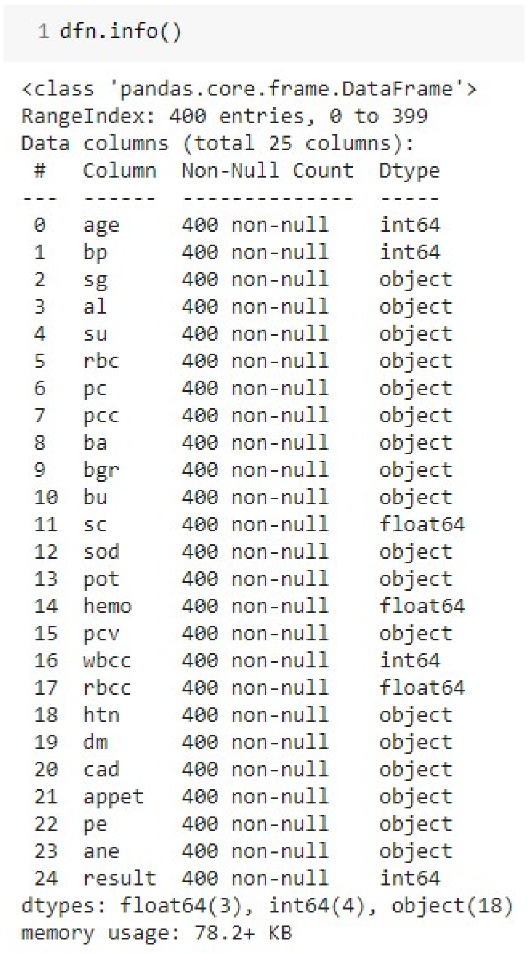
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 25 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   age     400 non-null    object
 1   bp      400 non-null    object
 2   sg      400 non-null    object
 3   al      400 non-null    object
 4   su      400 non-null    object
 5   rbc     400 non-null    object
 6   pc      400 non-null    object
 7   pcc     400 non-null    object
 8   ba      400 non-null    object
 9   bgr     400 non-null    object
 10  bu      400 non-null    object
 11  sc      400 non-null    object
 12  sod     400 non-null    object
 13  pot     400 non-null    object
 14  hemo    400 non-null    object
 15  pcv     400 non-null    object
 16  wbcc    400 non-null    object
 17  rbcc    400 non-null    object
 18  htn     400 non-null    object
 19  dm      400 non-null    object
 20  cad     400 non-null    object
 21  appet   400 non-null    object
 22  pe      400 non-null    object
 23  ane     400 non-null    object
 24  class   400 non-null    object
dtypes: object(25)
```

Figure 4.8: Data Type of DKD

In the figure 4.8, we can see, all the features of DKD Dataset are "object" datatype after the conversion. There are some features such as "age", "bp" which are needed to be treated as "float64" or "int" datatype. Therefore, we have used a method *.astype*() to convert the datatype of the specific features [34] [7].
The conversion process is below:

Listing 4.4: Datatype conversion for dkd

```
convert_dict = {
                'age'  : int,
                'bp'   : int,
                'sc'   : float,
                'hemo': float,
                'wbcc': int,
                'rbcc': float
               }

dfn = dfn.astype(convert_dict)
```



```
dfn.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 25 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   age     400 non-null    int64
 1   bp      400 non-null    int64
 2   sg      400 non-null    object
 3   al      400 non-null    object
 4   su      400 non-null    object
 5   rbc     400 non-null    object
 6   pc      400 non-null    object
 7   pcc     400 non-null    object
 8   ba      400 non-null    object
 9   bgr     400 non-null    object
 10  bu      400 non-null    object
 11  sc      400 non-null    float64
 12  sod     400 non-null    object
 13  pot     400 non-null    object
 14  hemo    400 non-null    float64
 15  pcv     400 non-null    object
 16  wbcc    400 non-null    int64
 17  rbcc    400 non-null    float64
 18  htn     400 non-null    object
 19  dm      400 non-null    object
 20  cad     400 non-null    object
 21  appet   400 non-null    object
 22  pe      400 non-null    object
 23  ane     400 non-null    object
 24  result  400 non-null    int64
dtypes: float64(3), int64(4), object(18)
memory usage: 78.2+ KB
```

Figure 4.9: Data Type of DKD after conversion

In the figure 4.6, the feature called ”bp” which is Blood Pressure ($diastolic\_value$ 60 <= 80) shows less impact on final result, so we have tested the changes of the value in this feature along with the value in ”age” using scatter plot and try to the visualize the relation between those two features in figure 4.10 .

In the figure 4.10, The blood pressure ($diastolic\_value$ 60 <= 80) is likely to be increased with the growth of values of ”age”. In the figure, we have found that, most instances in the dataset are aged between 40 to approximately 75 and high diastolic blood pressure is also measured between the age of 40 to 75.

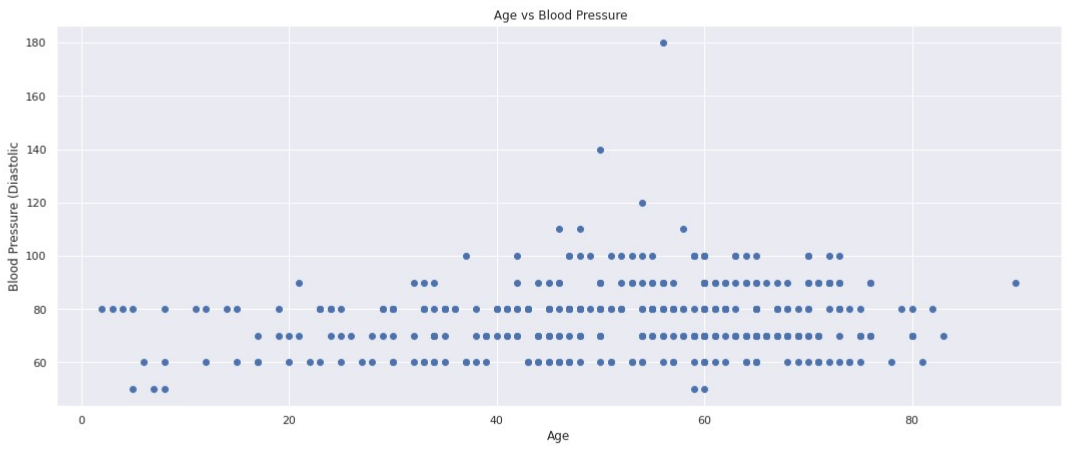


Figure 4.10: Age vs Blood Pressure

4.4 Train-Test Split

The independent features have been put into ”X” and the dependent column have been put as Y= ”Class”, then it has been divided for training and testing the models. Train-Test split was conducted for this purpose. The data has been divided into 10% for test set and 90% for the training set because of small instances in Diabetes dataset.

For dkd dataset, the target column is ”Y=result” and the other independent features are stored in X. Then X and Y has been split into training and test set for the models and Train-Test split has been conducted by dividing the dataset into 20% for test set and 80% for train set. To do this, the method *.train_test_split*() from *model_selection* is used which is provided by *sklearn* machine learning library. The training set is used to train our model. And accuracy evaluation has been done using the test set, which is unseen data for the trained model[10].

4.5 Data Analysis

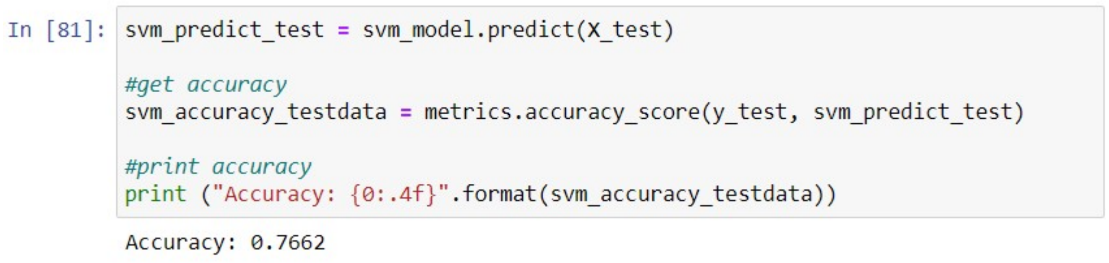
In the initial experiment, the Logistics Regression, Naive Bayes and Support Vector Machine (SVM) Classification algorithm have been used to train the machine learning models using the two of our datasets. Later, Ensemble Methods like CatBoost, XGBoost, LightGBM and also Artificial Neural Network (ANN) algorithm have been used for training and prediction to achieve higher efficiency and accuracy.

Firstly, we have used Logistics Regression Algorithm to train the model using Diabetes and Diabetic Kidney Disease datasets. The both datasets were split into two parts, Training set and Test set. The Test and Train set ratio is 20:80. From initial Logistics Regression Model we have been able to achieve 93.27% accuracy on Early Diabetes and 95.50% accuracy on Diabetic Kidney Disease (DKD).

Further we have used the OneHotEncoding Method to encode the categorical values in early diabetes dataset and using Logistics Regression we were able to increase our accuracy to 95.192%.
In our Support Vector Machine Model we have been able to achieve 76.62% accuracy on Diabetes dataset by using the parameters, random state=42 and kernel='rbf'

Radial Basis Function Kernal for SVM Classification:

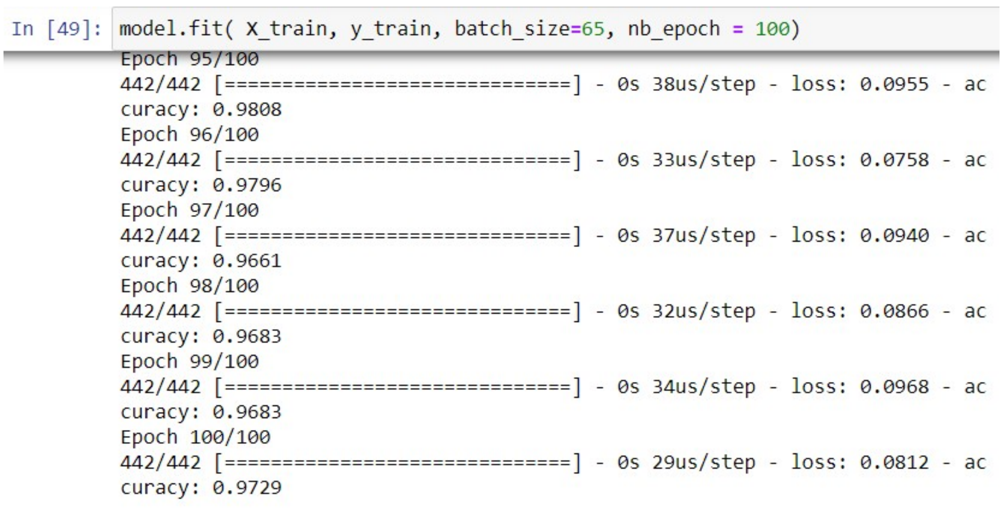
$$K(x, x') = exp[\frac{-1}{2} \parallel x - x' \parallel]$$



```
In [81]: svm_predict_test = svm_model.predict(X_test)

         #get accuracy
         svm_accuracy_testdata = metrics.accuracy_score(y_test, svm_predict_test)

         #print accuracy
         print ("Accuracy: {0:.4f}".format(svm_accuracy_testdata))

         Accuracy: 0.7662
```

Figure 4.11: SVM Model

By implementing Artificial Neural Network using Keras on the Diabetes Dataset, we have been able to achieve 94.87% accuracy. The Neural Network heve been implemented using sigmoid activation function on the output layer and relu activation function on the input layer. Input Dimension was set to 16. The Model was trained using batch size 16 and Epoch 100.



```
In [49]: model.fit( X_train, y_train, batch_size=65, nb_epoch = 100)
         Epoch 95/100
         442/442 [==============================] - 0s 38us/step - loss: 0.0955 - ac
         curacy: 0.9808
         Epoch 96/100
         442/442 [==============================] - 0s 33us/step - loss: 0.0758 - ac
         curacy: 0.9796
         Epoch 97/100
         442/442 [==============================] - 0s 37us/step - loss: 0.0940 - ac
         curacy: 0.9661
         Epoch 98/100
         442/442 [==============================] - 0s 32us/step - loss: 0.0866 - ac
         curacy: 0.9683
         Epoch 99/100
         442/442 [==============================] - 0s 34us/step - loss: 0.0968 - ac
         curacy: 0.9683
         Epoch 100/100
         442/442 [==============================] - 0s 29us/step - loss: 0.0812 - ac
         curacy: 0.9729
```

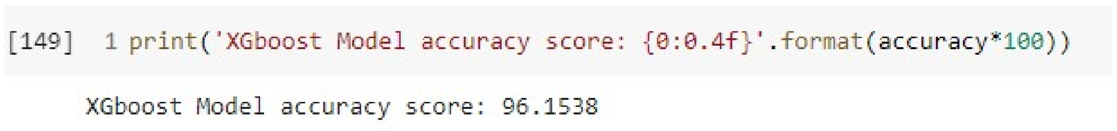
Figure 4.12: Artificial Neural Network

Later, the XGBoost model has been used to predict the target result for Diabetes. After implementing XGBoost algorithm in our model with the following parameter in Listing 4.5.

Listing 4.5: XGBoost class

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0,
              learning_rate=0.1, max_delta_step=0, max_depth=3,
              min_child_weight=1, missing=None, n_estimators=100,
              n_jobs=1,nthread=None, objective='binary:logistic',
              random_state=0,reg_alpha=0, reg_lambda=1,
              scale_pos_weight=1, seed=None,
              silent=None, subsample=1, verbosity=1)
```

The XGBoost showed 96.15% accuracy ,



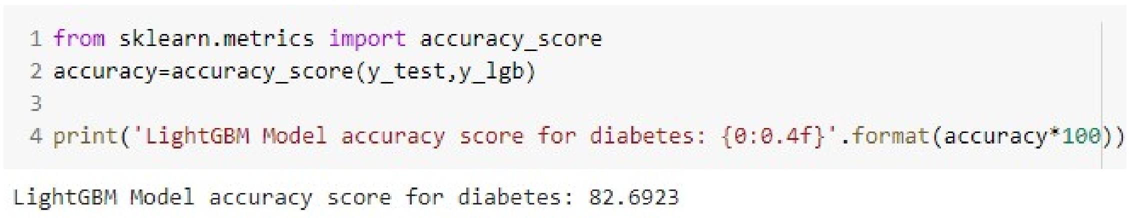
```
[149]  print('XGboost Model accuracy score: {0:0.4f}'.format(accuracy*100))

    XGboost Model accuracy score: 96.1538
```

Figure 4.13: XGBoost Accuracy Score in Diabetes Data

Another gradient boosting method called LightGBM has been used in both Diabetes and DKD dataset to understand the performance of the datasets on different approaches, that ensured the performance of the final models with tested accuracy. LightGBM algorithm in our model is initially used with the following parameters in Listing 4.6.

```
LGBMClassifier(boosting_type='gbdt', class_weight=None,
               colsample_bytree=1.0,importance_type='split',
               learning_rate=0.01, max_depth=5,
               min_child_samples=20, min_child_weight=0.001,
               min_split_gain=0.0,n_estimators=100,
               n_jobs=-1,
               num_leaves=16, objective=None,
               random_state=50, reg_alpha=0.0,
               reg_lambda=0.0, silent=True,
               subsample=1.0, subsample_for_bin=200000,
               subsample_freq=0)
```

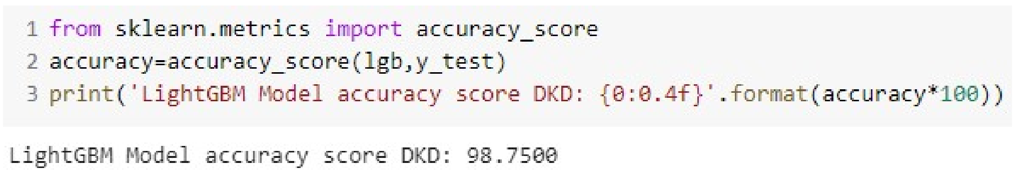
The LightGBM have showed 82.69% accuracy on Diabetes Prediction and The LightGBM scored 98.75% on DKD Prediction with the learning_rate=0.01 and boosting_type='gbdt' and other default parameters.



```
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(y_test,y_lgb)

print('LightGBM Model accuracy score for diabetes: {0:0.4f}'.format(accuracy*100))
```

```
LightGBM Model accuracy score for diabetes: 82.6923
```

Figure 4.14: Diabetes LightGBM



```
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(lgb,y_test)
print('LightGBM Model accuracy score DKD: {0:0.4f}'.format(accuracy*100))
```

```
LightGBM Model accuracy score DKD: 98.7500
```

Figure 4.15: DKD LightGBM

Chapter 5

Model Implementation and Optimization

5.1 Model Optimization

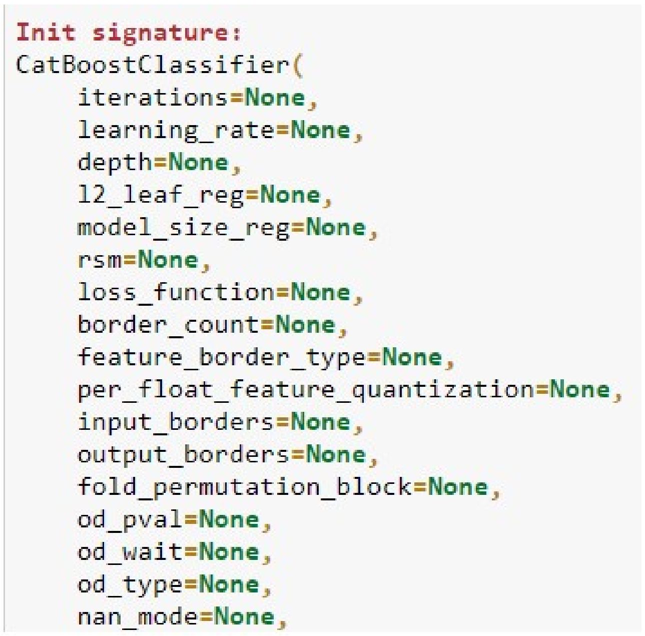
5.1.1 Hyperparameter Tuning

A parameter is a value learned during the training of a machine learning (ML) model, while a hyperparameter is a value set before the training of an ML model. These values then govern the learning process. Parameters are changed by the learning process while hyperparameters remain constant.

Parameters are assimilated automatically while hyperparameters have to be set manually. Different machine learning models have different hyperparameters, and tuning the right one for output is essential. Consequently, depending on the dataset, the optimal hyperparameters can vary. There are several approaches to hyperparameter tuning, such as:

1. Manual Tuning
2. Grid Search
3. Randomized Search

In this work, the Manual Tuning approach to hyperparameter tuning has been followed. CatBoost has a dynamic parameter tuning framework that can be modified for various tasks. The list of several CatBoost Parameters are given below:



```
Init signature:
CatBoostClassifier(
    iterations=None,
    learning_rate=None,
    depth=None,
    l2_leaf_reg=None,
    model_size_reg=None,
    rsm=None,
    loss_function=None,
    border_count=None,
    feature_border_type=None,
    per_float_feature_quantization=None,
    input_borders=None,
    output_borders=None,
    fold_permutation_block=None,
    od_pval=None,
    od_wait=None,
    od_type=None,
    nan_mode=None,
```

Figure 5.1: CatBoost Parameters

We have modified several parameters in CatBoost to optimize the performance of the models as well as to avoid overfitting in the models.

```
model = CatBoostClassifier(learning_rate=0.001,
                           eval_metric='AUC',
                           od_type='IncToDec',
                           od_wait=200,
                           od_pval=0.001,
                           depth=10,
                           l2_leaf_reg=9)
```

learning_rate: This parameter is used to reduce the gradient step size and has an impact on the overall training time. When the value is smaller more training iterations are necessary. The learning rate is automatically determined based on the properties of the datasets and the number of iterations.
Depending on the overfitting findings, there are a few options for changing the learning rate:

1. On the latter iterations of training, there is no overfitting (the training does not converge) - enhance the rate of learning.
2. Overfitting is recognized, which slows down learning.

If overfitting happens, CatBoost will end the training before the training parameters decide it. It can, for example, be stopped before a certain number of trees have been installed. This alternative can be found in the initial parameters. Here we use "$od\_type : IntToDec$", "$od\_pval : 0.001$" and "$od\_wait : 200$" for overfit detection in models. The type of overfitting detector to use:

IncToDec: CatBoost examines the resulting loss change on the validation dataset before creating each new tree. The overfit detector is triggered if the Threshold value set in the starting parameters is larger than the Current PValue.

$$CurrentPValue :$$

$$CurrentPValue < Threshold$$

How $CurrentPValue$ is determined from a set of values to maximize the metrix $score[i]$:

$$ExpectedInc\ is\ calculated :$$

$$ExpectedInc = max_{i_1 < i_2 < i} 0.99^{i-i_1}.(score[i_2] - score[i_1])$$

$$CurrentPValue\ is\ calculated :$$

$$CurrentPValue = exp(\frac{-0.5}{x})$$

Inter: CatBoost evaluates the number of iterations since the last iteration with the best loss function value before starting a new tree. If the number of iterations exceeds the value set in the training parameters the model is said to be overfitted.

eval_metric: This metric is used to detect overfitting and choose the best model. We have used "AUC" as "eval_metric" in our models. In our binary classification models, AUC is calculated using the following formula:

$$AUC = \frac{\sum_{t_i=0,t_j=1} I(a_i, aj) w_i w_j}{\sum_{t_i=0,t_j=1} w_i w_j}$$

L2_leaf_reg: It refers to the cost function's L2 regularization term coefficient. Any positive number will suffice. We have used the default value 3.0 for diabetic model and 9.0 for DKD model.

Depth: It refers to the tree's depth. The supported range values are determined by the processing unit type and loss function type. We use depth=10 for DKD and depth=14 to optimize the performance of the models which is evaluated by K-FOLD Cross Validation and AUC. The value can be set to any integer up to 16 on the CPU.

The default values of parameters of LightGBM are as follows:

```
LGBMClassifier(boosting_type='gbdt', class_weight=None,
               colsample_bytree=1.0,importance_type='split',
               learning_rate=0.01, max_depth=-1,
               min_child_samples=20, min_child_weight=0.001,
               min_split_gain=0.0,n_estimators=100,
               n_jobs=-1, num_leaves=31,
               objective=None,random_state=None,
               reg_alpha=0.0, reg_lambda=0.0,
               silent=True,
               subsample=1.0, subsample_for_bin=200000,
               subsample_freq=0)
```

boosting_type : With LightGBM, we are able to run distinctive sorts of Gradient Boosting strategies. GBDT, Dart, and GOSS are the methods which can be indicated with the "boosting" parameters. GBDT is the traditional Gradient Boosting Decision Tree. These days, GBDT is broadly utilized for its accuracy, proficiency, and stability. It depends on three key principles:

1. Weak Learners (Decision Trees)
2. Gradient Optimization
3. Boosting Procedure.

So within the GBDT strategy, there are a huge number of decision trees. Those trees are constructed step-by-step:

1. First among the trees discovers how to adjust the dependable(target) variable.
2. The residual (difference) between the main tree's expectations and the ground truth is accommodated by the second tree.
3. The third tree figures out how to accommodate the residuals of the moment tree, and so on.

DART boosting method stands for Dropouts meet Multiple Additive Regression Trees which utilize the dropout to solve the overfitting regression trees, taking advantage of the dropout settings in deep neural networks. Specifically, GBDT struggles from over-specialization, which means that trees added later in the iteration have a limited impact on the prediction of a small number of occurrences and have no influence on the remaining instances. Dropout makes it harder for the trees to focus on those few samples in following iterations, which increases performance. [14].

Gradient-based One-Side Sampling (LGBM GOSS) is a revolutionary examining approach that downsamples occurrences depending on gradients. As we know, instances with small gradients are successfully trained (low training error), while those with larger gradients are poorly trained. The conventional GBDT is reliable, however it is too slow for large datasets.

In this work, we have selected *boosting_type* : *dart* for its capability to deal with overfitting as our datasets are not that large and the other two methods might lead to overfitting [14].

Feature_fraction: This parameter indicates the fraction of features to be considered for each iteration. LightGBM deals with column sampling. On each iteration, it will choose a subset of features at random (tree). This function can be utilized to increase training speed as well as to combat overfitting. The default value for this feature is 1 but as we have the chance of overfitting of the model we have set it to 0.8 which shows that 80% of the features would be selected by LightGBM before training each tree.

learning_rate: This parameter defines the boosting learning rate which decides the impact of each tree on the ultimate outcome. GBM works by beginning with a starting estimate which is updated utilizing the output of each tree. The default value is 0.1 for LightGBM. We have set it to 0.05 for this work as there is a chance of overfitting for higher value of learning rate and also a chance of not gaining the better optimum result.

max_depth: This indicates the maximum depth to which each tree will be constructed. A single tree will halt splitting when there are no more parts that fulfill the min_rows parameter, in case it comes to max_depth, or in case there are no splits that fulfill this min_split_improvement parameter. The default value of this parameter is -1 which indicates it has no limit. But the larger value of this parameter can lead to overfitting of the model to the train set. The best value for the num_leaves parameter, model performance, and training duration will all be affected by this parameter. So we have set the value of max_depth to 5 so that deep trees are not grown and overfitting does not occur for the model as the data is small.

Metric: It is an important parameter because it signals model loss. Because it is classification-threshold invariant, we chose the 'AUC' measure for this technique. It assesses the accuracy of the models' estimations regardless of the classification level used.
num_leaves: This is a crucial parameter for controlling the tree model's complexity.
num_leaves might theoretically be set to $2^{(max\_depth)}$ to get the same amount of leaves as a depth-wise tree. In practice, though, this simple conversion is not ideal. The reason for this is that for a given number of leaves, a leaf-wise tree is consistently significantly more profound than a depth-wise tree. Overfitting can occur when depth is unconstrained. As a result, while tuning the num_leaves, it is preferable to make it smaller than $2^{(max\_depth)}$.

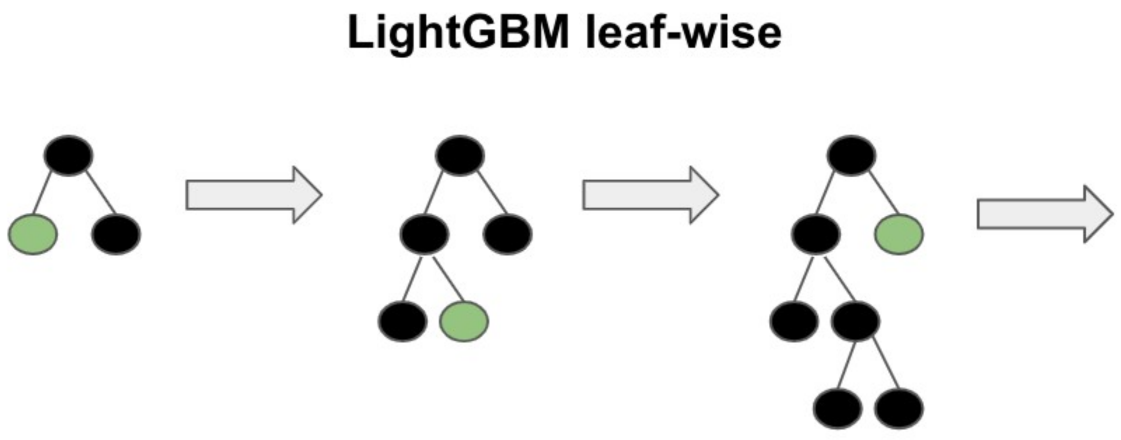


Figure 5.2: LGBM leaf-wise

Finally, as our problem is a binary classification problem we have set the parameter 'objective' to binary which stays as regression by default. These are the parameters we have tuned for our work to get greater accuracy dealing with over-fitting

5.2 Evaluating Machine Learning Models

For Model Evaluation, we have used 10- Fold Cross Validation in all the models. In 10- Fold Cross Validation, Logistic Regression scores 91.84% and Aritificial Neural Network scores 94.87% on Early Diabetes Prediction.

Logistic Regression cross Validation scores 98% in diabetic kidney disease prediction. We have also measured the precision, Recall and F1 - Score of each models given in table 5.1.
In the given evaluation table 5.1, we can see XGBoost scores 96.15% accuracy which outperforms the Artificial Intelligence, Logistics Regression, Support Vector Machine and LightGBM in Diabetes Prediction Model.

Dataset	Algorithm	Accuracy	Precision	Recall	F1-Score
Early Diabetes	LR	95.19%	95.12%	94.68%	94.8 %
	SVM	76.62%	85%	78%	83%
	ANN	94.77%	85%	78%	83%
	XGBoost	96.15%	96.13%	96.13%	96.13%
	LightGBM	82.69%	85.63%	81.54%	81.88%
Diabetic Kidney Disease	LR	95.50%	93.66%	93.66%	94.50 %
	LightGBM	98.75%	97.22%	97.82%	97.46%

Table 5.1: Model Evaluation

Logistics Regression also shows promising performances in both datasets, 95.19% acccuracy in diabetes model and 95.50% accuracy in DKD model.
In our dataset, there are mostly categorical features. Artificial Neural Network also shows good accuracy in Early Diabetes prediction. SVM scores lowest in Early Diabates prediction and LightGBM shows highest accuracy of 98.75% in dkd prediction model.

After analyzing the table, the CatBoost, which is comparatively new and a state-of-the-art open-source GBDT library, has been implemented on our datasets to optimize the performance, for gaining higher accuracy in prediction also because as it is faster than XGBoost [32].

CatBoost: Catboost has only been around for a year and is already posing a threat to XGBoost, LightGBM, and H2O. CatBoost is a gradient boosting decision tree-based machine learning approach (GBDT). Gradient boosting is a machine learning approach for dealing with issues with many features, noisy data, and complex dependencies. CatBoost has the following advantages over other

GBDT algorithms. To begin with, categorical characteristics are where this algorithm thrives. Categorical features can be replaced with average label values using the traditional GBDT approach. Features, on the whole, have larger details than labels. As a result, we utilize average label value to aggressively represent features for which a conditional shift will occur.

CatBoost, on the other hand, combines various categorization features. It integrates all categorical qualities and their combinations in the current tree with all categorical features in the dataset using a greedy strategy. Finally, CatBoost has the ability to correct for gradient bias. The gradient of the previous learner is used to instruct each learner in each cycle of GBDT. The output is the sum of all learners' categorized results, which causes the final trained model to be overfitted [25].

CatBoost uses a revolutionary technique known as ordered boosting to tackle this problem. This method can help the model generalize better by overcoming gradient bias-induced prediction shift. CatBoost trains a different model "M" for each sample "X" in order to achieve unbiased gradient estimation. Model "M" is trained using a training set that excludes sample "X." It use "M" to obtain a sample gradient estimation. This gradient will also be utilized to train the base learner in the final model [25] [32].

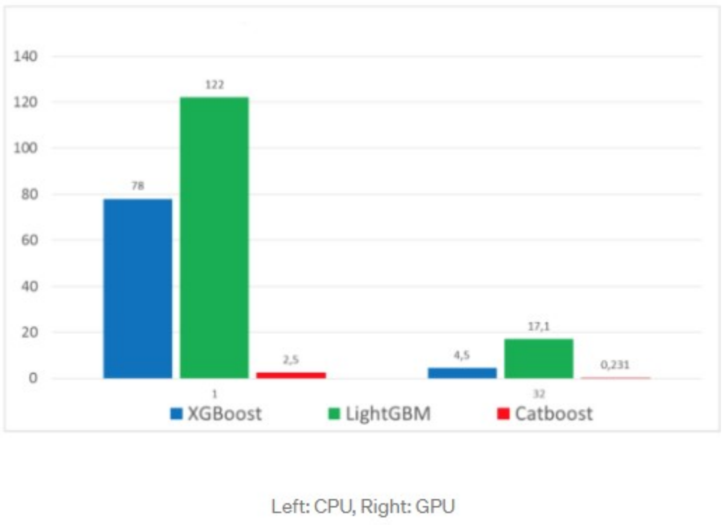


Figure 5.3: Prediction Performance CatBoost: Left: CPU, Right: GPU

In Early Diabetes dataset, we are able to achieve 96.15% accuracy to predict the early diabetes from the given dataset. The parameters with $CatBoostClassifier()$ is used in our model for early diabetes prediction as follows:

```
model = CatBoostClassifier(l2_leaf_reg=3.0,
                           learning_rate=0.0001,
                           eval_metric='AUC',
                           od_type='IncToDec',
                           od_wait=200,
                           od_pval=0.001,
                           depth=14,
                           )
```

Then, the model has been evaluated using different methods to justify the outcome using the given dataset. A test dataset is used to test the performance of the model. The true positive result is 27, true negative result is 23 and the false positive result is 1 and false negative result is 1. Confusion Matrix is provided below:

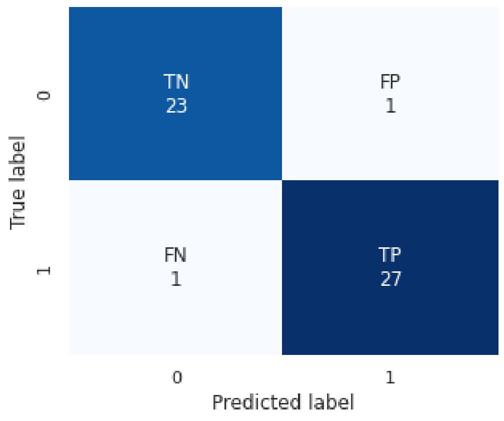


Figure 5.4: Confusion matrix for early diabetes model

In the dkd dataset, we are able to achieve 97.5% accuracy to predict Diabetic Kidney disease from the given dataset.A test dataset has been used to test the performance of the model for DKD prediction. The true positive result is 44, true negative result is 34 and the false positive result is 0 and false negative result is 2. Confusion matrix visualization is given below:

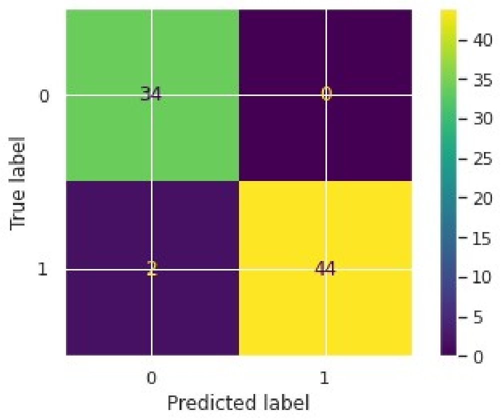


Figure 5.5: Confusion matrix for dkd model

From the context of health and medical research, **Sensitivity** refers to the probability of a test to trace a state in a patient if it is really present. A test with a low sensitivity is considered too cautious when looking for a positive result, which means it may miss a disease in a sick individual. A high sensitivity test is less likely to result in a false negative. A positive is a positive in a test with a high sensitivity.

$$Sensitivity/Recall = \frac{TP}{TP + FN}$$

The specificity of a test refers to its ability to detect the existence of an illness in someone who doesn't have it. A true negative is a negative result in a test with high specificity. A test with a low specificity is able to seek a positive result even if one does not exist, leading to a high number of false positives.

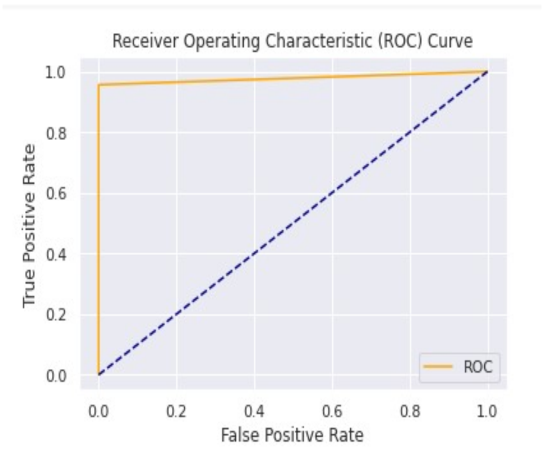
$$Specificity = \frac{TN}{TN + FP}$$

In diabetes model, we got 95.83% Specificity and 96.42% Sensitivity.

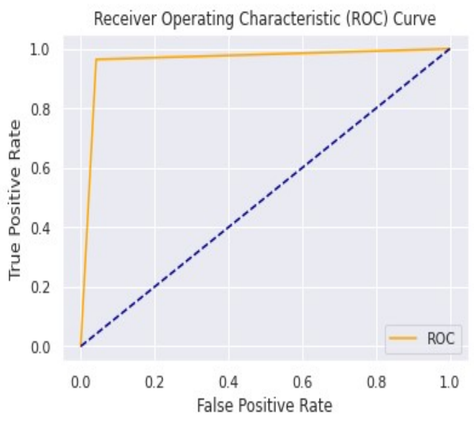
Generally, a medical test with a Sensitivity and Specificity of around 90% is considered to have good diagnostic performance. So, our model is performing well in diabetes dataset.

For DKD, the specificity is 100.0% which means no healthy individuals are identified as infected. In medical test, False Positive Rate (FPR) should be as low as possible cause False Negative results can be further analyzed based on the patient symptoms. False positive result creates problems for patient and doctors as well. The sensitivity is 95.65% in DKD. So, the model is performing well in DKD dataset also.

TPR is plotted against FPR on the y-axis, while FPR is plotted on the x-axis in the ROC curve.



(a) DKD ROC



(b) Diabetes ROC

Figure 5.6: ROC Curve for the models

ROC Curve scores for Diabetes is 0.9613 and for DKD is 0.9782 using Catboost. The ROC curve or Area Under the Curve (AUC), is a performance measure for classification problems at different threshold range. AUC represents the degree or measure of separability, whereas ROC is a probability curve.

It shows how precisely the model can distinguish between classes. The AUC measures how effectively the model correctly predicts 0s as 0s and 1s as 1s. By analogy, the higher the AUC, the more accurately the model distinguishes among people who have the condition and those who do not.

In Early Diabetes, using LightGBM with hyperparameter tuning, we are able to achieve 90.3846% accuracy as given in the confusion matrix in Figure 5.6 :

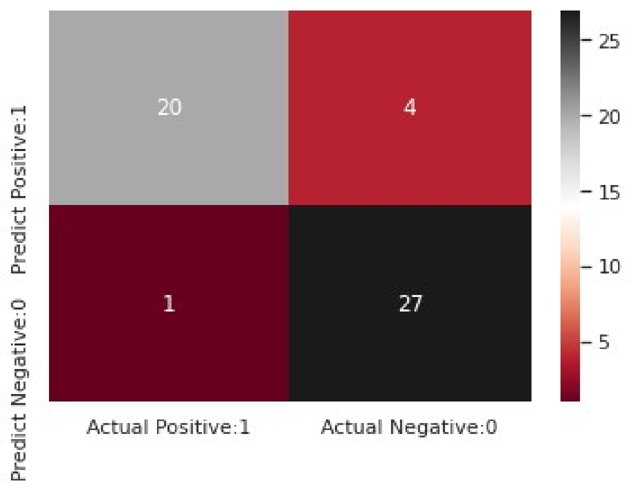


Figure 5.7: Confusion matrix for LGBM with hyperparameter tuning

True Negatives(TN) = 20

True Positives (TP) = 27

False Positives(FP) = 4

False Negatives(FN) = 1

The False Positive result is higher than previous model using Catboost as mentioned before. False Positive result need to be as low as possible in any medical test and prediction. The Sensitivity is 96.425% and the Specificity is 83.33%. In this model, the specificity is lower than previous model where we got specificity of 95.83% [10].

In DKD, using LightGBM with hyperparameter tuning, the accuracy remains unchanged with the previous LGBM model in table 5.1 without hyperparameter tuning.

However, the precision 98.57%, recall 98.91%, f1-score 98.72% has been changed and got better than previous LGBM model in table 5.1. The confusion matrix for this model using DKD dataset given in figure 5.7.

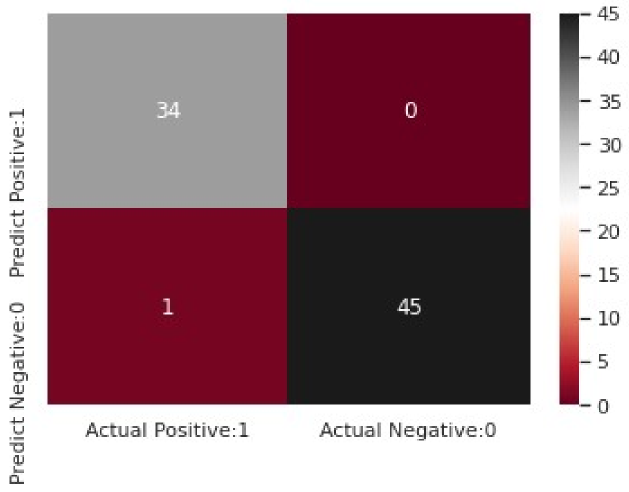


Figure 5.8: hyperparamter tuned LGBM model confusion matrix dkd

True Negatives(TN) = 34
True Positives (TP) = 45
False Positives(FP) = 0
False Negatives(FN) = 1

In this model, the overall result is slightly better than previous catboost model for prediction of DKD in patients. The Sensitivity is 97.82% and the Specificity is 100% to predict the DKD in patient using given dataset.

Chapter 6

Result Analysis

Finally, LightGBM model has been implemented on DKD dataset and the Catboost model has been implemented on Early Diabetes dataset. The models have been tested using 10-fold Cross Validation. Cross-validation is a quantization technique for testing machine learning models on a short dataset. The procedure defines how many groups should be created from a given data sample. As a result, k-fold cross-validation is a common name for the procedure. In the models reference, it can be exchanged for k. For instance, k=10 for 10-fold cross-validation. we have used the most recent *cross_val_score* method from sklearn library to check our accuracy on our diabetes prediction and DKD prediction models. The cross validation scores are very promising for both models: LightGBM on DKD and CatBoost on Diabetes[10].

Data	**ML Model**	**Accuracy**	**Specificity**	**Sensitivity**
Early Diabetes	CatBoost	96.15%	95.83%	96.42%
DKD	LightGBM	98.75%	100%	97.82%

Table 6.1: Final Result Analysis

Chapter 7

Application Design

7.1 Model Selection

Finally, The CatBoost model has been selected to use in the backend of our disease prediction application for Diabetes and LightGBM model has been selected to use for Diabetic Kidney Disease Prediction. The CatBoost model has shown promising results in both datasets, DKD and Diabetes. Eventually, LightGBM shows optimum result in DKD prediction and Catboost model has selected for early diabetes prediction.

7.2 Application Design

The figure 7.1 shows the workflow diagram of our proposed prediction application. The web application has been designed using HTML,CSS, JS and Django. In home page, two dedicated buttons for DKD and Diabetes Prediction were added. The users can give the inputs based on their physiological symptoms and diagnostic reports. The inputs then trigger the training function to train the model using the provided data from the database and the user input data is passed to the prediction function to predict the result as ”Positive” and ”Negative”.

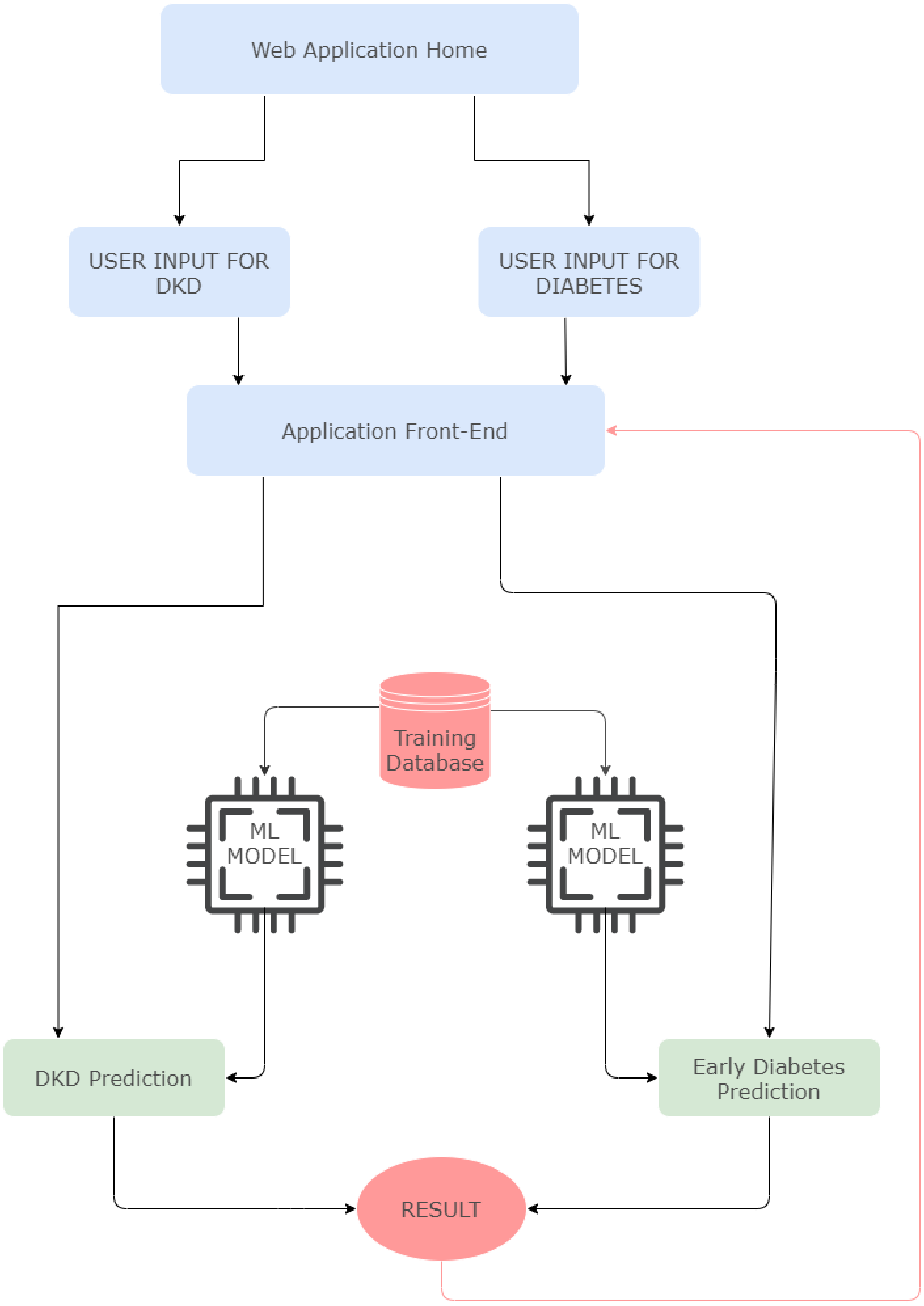


Figure 7.1: Web Application Workflow Diagram

Chapter 8

Conclusion and Future Work

8.1 Future Work

There is a scope for improvement in every research work and so is our research work which also has some room for development in several areas. From the very beginning, our plan was to collect data of Bangladeshi people which was not possible due to the pandemic. So, in the future our main objective is to collect a dataset of the people of Bangladesh. Then try to work with the hospitals in Bangladesh to gain a better perspective on data, such as to get more access to data and get more ideas about all the features involved in the predictions. Afterwards, our plan is to collect a much larger dataset so that we can come up with much more efficient and accurate predictions. Since, the main logic of machine learning and ensemble models is to predict correctly and efficiently based on larger datasets. Lastly, our motive is to try using our system in much more real life scenarios. We will try to test and validate the prediction application on much more reliable data, such as early diabetes in case of Bangladeshi people or in case of people of a specific region based on whichever dataset is available. We hope that our research will enable us to bring new changes in the medical sector, and hopefully change many lives for the better in the future.

8.2 Conclusion

Diabetes is becoming a global problem that affects both industrialized and underdeveloped countries. As a result, early diabetes detection and prediction of renal disease are critical for preventing renal failure. Early detection of diabetes and diabetic kidney disease will reduce the cost, time, and effort required to identify both diseases separately. It will help the doctors to detect diabetes early in the patients and also if they are at risk of being attacked by diabetic kidney disease and thus provide medication as thus. It will also decrease the risks of the patients being undiagnosed for any of the diseases and thus receive medication and treatment earlier than being affected much. This prediction system will be beneficial for the doctors and patients and also reduce the hassle of patients conducting test individually for both conditions. In this way both doctors and patients will have less hassle of conducting tests individually for two diseases and detection will be easier. Prediction will be made earlier and thus proper and effective treatment and medication can be started earlier so that the consequences do not become worse with the days proceeding. Our system will hopefully open more scopes of faster and more efficient detection of Diabetes along with Diabetic Kidney Disease. The system could lead to more effective and precise intervention which could motivate people to seek medical specialist consultation in time and make aware people about their health condition.

Bibliography

[1] D. H. Moore II, "Classification and regression trees, by leo breiman, jerome h. friedman, richard a. olshen, and charles j. stone. brooks/cole publishing, monterey, 1984,358 pages, $27.95," *Cytometry*, vol. 8, no. 5, pp. 534–535, 1987. DOI: https://doi.org/10.1002/cyto.990080516. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cyto.990080516. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cyto.990080516.

[2] S. R. Garner *et al.*, "Weka: The waikato environment for knowledge analysis," in *Proceedings of the New Zealand computer science research students conference*, 1995, pp. 57–64.

[3] J. H. Friedman, "Greedy function approximation: A gradient boosting machine.," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001. DOI: 10.1214/aos/1013203451. [Online]. Available: https://doi.org/10.1214/aos/1013203451.

[4] S. I.-J. Chien and C. M. Kuchipudi, "Dynamic travel time prediction with real-time and historic data," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 608–616, 2003. DOI: 10.1061/(asce)0733-947x(2003)129:6(608).

[5] Q. Yang, M. J. Khoury, J. Friedman, J. Little, and W. D. Flanders, "How many genes underlie the occurrence of common complex diseases in the population?" *International journal of epidemiology*, vol. 34, no. 5, pp. 1129–1137, 2005.

[6] M. Schmid and T. Hothorn, "Flexible boosting of accelerated failure time models," *BMC bioinformatics*, vol. 9, p. 269, Feb. 2008. DOI: 10.1186/1471-2105-9-269.

[7] W. McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, S. van der Walt and J. Millman, Eds., 2010, pp. 56–61. DOI: 10.25080/Majora-92bf1922-00a.

[8] M. Schmid, T. Hothorn, K. O. Maloney, D. E. Weller, and S. Potapov, "Geoadditive regression modeling of stream biological condition," *Environmental and Ecological Statistics*, vol. 18, no. 4, pp. 709–733, 2010. DOI: 10.1007/s10651-010-0158-4.

[9] W. Yu, T. Liu, R. Valdez, M. Gwinn, and M. J. Khoury, "Application of support vector machine modeling for prediction of common diseases: The case of diabetes and pre-diabetes," *BMC Medical Informatics and Decision Making*, vol. 10, no. 1, 2010. DOI: 10.1186/1472-6947-10-16.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[11] R. Arora, "Comparative analysis of classification algorithms on different datasets using weka," *International Journal of Computer Applications*, vol. 54, no. 13, 2012.

[12] C. Becker, R. Rigamonti, V. Lepetit, and P. Fua, "Supervised feature learning for curvilinear structure segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, K. Mori, I. Sakuma, Y. Sato, C. Barillot, and N. Navab, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 526–533, ISBN: 978-3-642-40811-3.

[13] R. K. Leung, Y. Wang, R. C. Ma, A. O. Luk, V. Lam, M. Ng, W. Y. So, S. K. Tsui, and J. C. Chan, "Using a multi-staged strategy based on machine learning and mathematical modeling to predict genotype-phenotype risk patterns in diabetic kidney disease: A prospective case–control cohort analysis," *BMC nephrology*, vol. 14, no. 1, p. 162, 2013.

[14] K. V. Rashmi and R. Gilad-Bachrach, *Dart: Dropouts meet multiple additive regression trees*, 2015. arXiv: 1505.01866 `[cs.LG]`.

[15] L. J. Rubini, "Ucimachinelearningrepository [http://archive. ics. uci. edu/ml/-datasets/chronic_kidney_disease]. karaikudi," *TamilNadu: Algappa University, Department of Computer Science and Engineering*, 2015.

[16] M. C. Thomas, M. Brownlee, K. Susztak, K. Sharma, K. A. Jandeleit-Dahm, S. Zoungas, P. Rossing, P.-H. Groop, and M. E. Cooper, "Diabetic kidney disease," *Nature Reviews Disease Primers*, vol. 1, no. 1, pp. 1–20, 2015.

[17] N. K. F. Inc. (2016). "Diabetes and chronic kidney disease," [Online]. Available: https://www.kidney.org/news/newsroom/factsheets/Diabetes-And-CKD (visited on 09/10/2020).

[18] U. M. Learning. (2016). "Pima indians diabetes database," [Online]. Available: https://www.kaggle.com/uciml/pima-indians-diabetes-database (visited on 06/10/2020).

[19] S. Narula, K. Shameer, A. M. S. Omar, J. T. Dudley, and P. P. Sengupta, "Machine-learning algorithms to automate morphological and functional assessments in 2d echocardiography," *Journal of the American College of Cardiology*, vol. 68, no. 21, pp. 2287–2295, 2016.

[20] D. Dua and C. Graff, *UCI machine learning repository*, 2017. [Online]. Available: http://archive.ics.uci.edu/ml (visited on 06/01/2021).

[21] I. S. Boon, T. Au Yong, and C. S. Boon, "Assessing the role of artificial intelligence (ai) in clinical oncology: Utility of machine learning in radiotherapy target volume delineation," *Medicines*, vol. 5, no. 4, p. 131, 2018.

[22] J. Bouaziz, R. Mashiach, S. Cohen, A. Kedem, A. Baron, M. Zajicek, I. Feldman, D. Seidman, and D. Soriano, "How artificial intelligence can improve our understanding of the genes associated with endometriosis: Natural language processing of the pubmed database," *BioMed research international*, vol. 2018, 2018.

[23] J. Catanzarite. (2018). "The naive bayes classifier," [Online]. Available: https://towardsdatascience.com/the-naive-bayes-classifier-e92ea9f47523 (visited on 08/01/2020).

[24] D. Deshpande, J. G. Pasipanodya, S. G. Mpagama, P. Bendet, S. Srivastava, T. Koeuth, P. S. Lee, S. M. Bhavnani, P. G. Ambrose, G. Thwaites, *et al.*, "Levofloxacin pharmacokinetics/pharmacodynamics, dosing, susceptibility breakpoints, and artificial intelligence in the treatment of multidrug-resistant tuberculosis," *Clinical Infectious Diseases*, vol. 67, no. suppl_3, S293–S302, 2018.

[25] T. Peretz, "Mastering the new generation of gradient boosting," 2018. [Online]. Available: https://towardsdatascience.com/https-medium-com-talperetz24-mastering-the-new-generation-of-gradient-boosting-db04062a7ea2 (visited on 06/01/2021).

[26] K. P. Shung. (2018). "Accuracy, precision, recall or f1?" [Online]. Available: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9 (visited on 08/01/2020).

[27] D. Sisodia and D. S. Sisodia, "Prediction of diabetes using classification algorithms," *Procedia computer science*, vol. 132, pp. 1578–1585, 2018.

[28] G. Urban, P. Tripathi, T. Alkayali, M. Mittal, F. Jalali, W. Karnes, and P. Baldi, "Deep learning localizes and identifies polyps in real time with 96% accuracy in screening colonoscopy," *Gastroenterology*, vol. 155, no. 4, pp. 1069–1078, 2018.

[29] I. D. ATLAS. (2019). "Diabetes facts & figures," [Online]. Available: https://www.idf.org/aboutdiabetes/what-is-diabetes/facts-figures.html (visited on 09/10/2020).

[30] M. Makino, R. Yoshimoto, M. Ono, T. Itoko, T. Katsuki, A. Koseki, M. Kudo, K. Haida, J. Kuroda, R. Yanagiya, *et al.*, "Artificial intelligence predicts the progression of diabetic kidney disease using big data machine learning," *Scientific reports*, vol. 9, no. 1, pp. 1–9, 2019.

[31] A. Mohiuddin. (2019). "Diabetes fact: Bangladesh perspective," [Online]. Available: http://www.ghrnet.org/index.php/ijdr/article/view/2457/2836 (visited on 06/01/2021).

[32] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, *Catboost: Unbiased boosting with categorical features*, 2019. arXiv: 1706.09516 `[cs.LG]`.

[33] e. a. Islam MM Faniqul, "Likelihood prediction of diabetes at early stage using data mining techniques," *Computer Vision and Machine Intelligence in Medical Image Analysis*, pp. 113–125, 2020.

[34] T. pandas development team, *Pandas-dev/pandas: Pandas*, version latest, Feb. 2020. DOI: 10.5281/zenodo.3509134. [Online]. Available: https://doi.org/10.5281/zenodo.3509134.